Computer Networks 150 (2019) 1–14

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

A model for integrating heterogeneous sensory data in IoT systems

Siyao Cheng[a], Yingshu Li[b,*], Zhi Tian[c], Wei Cheng[d], Xiuzhen Cheng[e]

[a] School of Computer Science and Technology, Harbin Institute of Technology, 92 West Dazhi Street, Nan Gang District, Harbin, 150001, Heilongjiang, China
[b] Department of Computer Science, Georgia State University, 25 Park Place, Suite 753, Atlanta, GA 30303, USA
[c] Department of Electrical and Computer Engineering, George Mason University, 4400 University Drive, MS 1G5 Fairfax, Virginia, 22030, USA
[d] Department of Computer Science, University of Washington Tacoma, 1900 Commerce StreetTacoma, Washington, 98402 USA
[e] Department of Computer Science, George Washington University, 800 22nd St. NW, Suite 4000, Washington DC, 20052 USA

ARTICLE INFO

Article history:
Received 23 May 2018
Revised 8 October 2018
Accepted 29 November 2018
Available online 13 December 2018

Keywords:
Heterogeneous
Multi-Modal sensory data
Internet of things
Integration

ABSTRACT

With the development of Internet of Things (IoT), heterogeneous sensory data appears everywhere in our lives. Unlike traditional sensory data, heterogeneous sensory data often involves variety modalities of data in one set, so that it is called as the *multi-modal sensory data* in this paper. The appearance of such data making it possible to monitor more complicated objects and improve monitoring accuracy. However, due to lack of integration model for multi-modal sensory data, most of the existing sensory data management algorithms only consider single modal sensory data, resulting in insufficient utilization of sensory data. Thus, we propose a model for integrating the heterogeneous sensory data generated in a IoT system based on Hidden Markov Process in the paper. The distributed algorithm for constructing such a model is then presented. The integration model can be applied to many applications, while we take the cooperative event detection as an example for illustration. The extensive theoretical analysis and experimental results show that all the proposed algorithms are efficient and effective .

1. Introduction

With the rapid development of sensing techniques, embody systems and cross-technology communication [1–3], various sensors are always involved in a IoT system or even in a single device. For example, the current smart phones are equipped with several different sensors, such as accelerometer, digital compass, gyroscope, GPS, microphone and camera [4]. An intelligent traffic monitoring system could involve many flow monitoring sensors, such as electronic eyes, GPS devices and intelligent traffic lights. A smart home application always contains the RFIDs for locating some objects, the sensors for sampling the temperature, humidity, light intensity, air flow and so on in the environment, the smart bracelet for obtaining the healthy information of monitoring people, the cameras and acoustic sensors for catching the abnormal informations and guaranteeing the safety of house *etc*.

Unlike the traditional sensor networks, the sensory data sampled by the current IoT system not only have big volume [5,6] but also involved diverse modalities. In the aforementioned example, a crowdsourcing task running in a smart phone may use the accelerometer, microphone and camera to collect sensory data simultaneously, while the sensory data sampled by them are vector data, audio data and video data, respectively. Similarly, an intelligent traffic system also generates scalar data, vector data and video data simultaneously. Meanwhile, in a forest ecology monitoring system, temperature and humidity are presented as scalar data, wind velocity and direction are presented as vector data, and pictures of plants and videos of animals are presented as multimedia data. Furthermore, in a smart home application, the dataset includes the scalar data such as temperature, humidity *.etc*, the vector data, such as the movement information of monitoring persons, and the multimedia data, such as the data sampled by the camera and acoustic sensors. We notice that the data set generated by the above IoT systems refer to multiple modalities, and we call such heterogeneous data set as *multi-modal sensory data set*.

The appearance of such multi-modal sensory data provide abundant information and great opportunities to reveal the mysterious physical world, and it also brings many benefits for current IoT system. Firstly, the multi-modal sensory data supply plenty of semantics information comparing with the traditional sensory data. Since each modality of sensory data give some new information about the monitoring objects, and thus, the multi-modal sensory data breaks the limitation of the single-modal sensory data, and make it possible for multi-preceptive observation and analysis. Secondly, more complexity objects could be monitored by current IoT system with the help of multi-modal sensory data. Obviously, more detailed and comprehensive information are required

* Corresponding author.
E-mail address: yili@gsu.edu (Y. Li).

https://doi.org/10.1016/j.comnet.2018.11.032
1389-1286/© 2018 Elsevier B.V. All rights reserved.

when the monitoring objects are complex, and the multi-modal sensory data set meets such requirements since it provides abundant semantics information. Thirdly, the multi-modal sensory data improve the utilization of system and shorten the latency of discovering the abnormal information. Since the sensory data of different modalities are related with each other, the system utilization rate will be further promoted if we sufficiently take advantage of such relationship. Meanwhile, the abnormal event could be detected in time with the help of different modalities of sensory data, so that it will save lots of time for event detection.

Based on the above discussion, the multi-modal sensory data are quite useful for current IoT system, and they will be ubiquitous for us since the monitoring objects of current IoT systems become more and more complex. However, it also brings many challenges on how to manage and make maximum utilization of these data. Although there are a great number of distributed sensory data management algorithms in traditional sensor networks, including data acquisition algorithms [7], data collection algorithm [8], data mining and modeling algorithms [9,10], data transmission scheduling algorithms [11,12], and query processing algorithms [13,14] *.etc*, but most of them are only suitable for dealing with scalar data and cannot process more complicated data. Some of the works, such as [15–17], investigate how to deal with multimedia data in WSNs, but they only consider one modality of sensory data and cannot deal with multi-modal sensory data. Besides, the multi-modal sensory data are also quite different from the traditional heterogeneous data that have been studied because most of them, as discussed in [18–20], only consider the data with different structures, while they still share the same modality.

To the best of our knowledge, we are the first one to consider the problem of dealing with the multi-modal sensory data. Then, the first problem is coming: can we process sensory data one modality by one modality separately without fusing them together? Unfortunately, the answer is no. Since most of the current monitored objects become more complicated than they used to be, one or two modalities of sensory data cannot describe them accurately. For example, to discover the variation of forest ecology, temperature, humidity, wind velocity and direction, video and image data should be managed simultaneously. To recognize human activities by smart phones, the sensory data sampled by the accelerometer, digital compass, gyroscope, GPS, microphone and camera should all be taken into account. Moreover, fusing computation on the multi-modal sensory data also improves the observation accuracy. For example, in a fire monitoring system, it will catch the threat of fire as early as possible if temperature, light, video and audio data are considered together.

Due to the above reasons, a group of fusing computation algorithms on multi-modal sensory data are desired for current IoT systems. However, it is quite challenging to simultaneously deal with even two modalities of sensory data as their representations are quite diverse. To make the fusing computations to be possible, a model of integrating the multi-modal sensory data is highly expected.

In this paper, we construct such a model according to the Hidden Markov Process [21]. The model firstly projects each sensory data stream collected by a sensor node into a sequence of states. Thus, the fusing computations can be executed on states instead of on the raw sensory data. To the best of our knowledge, it is the first model to consider the problem of how to integrate the multi-modal sensory data in IoT systems. Such a model projection process makes the fusing computation on multi-model sensory data to be possible and can be applied to many applications. For example, discovering the relationship between different models of sensory data, backtracking the reason of certain phenomenons, mining the pattern of frequent observations, detecting the events cooperatively, *etc*. Furthermore, this model can provide insights for the sensor deployment strategy to cover the events, the system control method to avoid disasters, *etc*, thus, it is valuable for the current IoT systems.

Finally, the cooperative event detection is taken as an example to show how to use our model to support the fusing computations on multi-modal sensory data because the event detection is one of most important operations in IoT systems. Other fusing computation operations will be discussed in our future works due to the space limitation. In summary, the main contributions of our paper are summarized as follows.

(1) The definitions of *multi-modal sensory data* and *the problem of fusing computation on multi-modal sensory data* are firstly proposed.

(2) A model for integrating the multi-modal sensory data generated in a IoT system is provided. The algorithm for learning such a model according to the training data is given.

(3) A distributed algorithm for detecting the events cooperatively is presented based on the above model.

(4) The real system experiments were carried out. The extensive experimental results verify the efficiency and effectiveness of all the proposed algorithms.

The rest of the paper is organized as follows. Section 2 provides the problem definition. Section 3 discusses how to construct the model for integrating the multiple modal sensory data. Section 4 proposes a distributed cooperative event detection algorithm. Section 5 presents the experimental results. Section 6 surveys the related works and Section 7 concludes the paper.

2. Problem definition

Assume that there are n sensor nodes in a IoT system, indexed by $\{1, 2, \cdots, n\}$. Similar to traditional sensor networks, each sensor node i samples a sensory data stream from the monitored physical world. Let D_i denote the sensory data stream sampled by sensor node i, and $d_{it} \in D_i$ denotes the snapshot value sampled by sensor node i at time t. As mentioned in Section 1, the type of d_{it} depends on D_i, *i.e.* d_{it} does not have to be a single value. For example, d_{it} is a frame of image if D_i is a video stream, d_{it} is a scalar value if D_i is a scalar data stream, d_{it} is a vector if D_i is a vector data stream, *etc*. Furthermore, the clocks of all the sensor nodes in the system are synchronized according to some well established techniques [22].

Let $f_i (1 \le i \le n)$ be the sampling frequency of sensor node i. In a given time window $[T_s, T_f]$, the sensory data of sensor node i can be regarded as a set of m snapshots, *i.e.*, $D_i(T_s, T_f) = \{d_{it_1}, d_{it_2}, \ldots, d_{it_m}\}$, where $t_1 = T_s$, $m = |T_f - T_s| \times f_i$, and $t_{r+1} - t_r = 1/f_i$ for any $1 \le r \le m-1$.

Since the sampling frequency of a sensor node could be large, the consecutive snapshots from a sensor may be very similar with each other. Thus, we use *observation* to denote a set of consecutive snapshots which have little variation. The formal definition of *observation* is given as follows.

Definition 1 (Observation)**.** An observation of sensor i, denoted by o_{il}, satisfies that o_{il} is a set of consecutive snapshots, where l is an integer to identify the serial number of observations in D_i. Thus, $o_{il} = \{d_{it_{l_1}}, d_{it_{l_2}}, \ldots, d_{it_{l_k}}\}$, where $t_{l_1} < \ldots < t_{l_k}$ and $t_{l_{j+1}} - t_{l_j} = 1/f_i$ for $\forall 1 \le j \le k-1$.

Therefore, a sensory data stream in any given time window $[T_s, T_f]$ can be divided into a set of observations, *i.e.*, $D_i(T_s, T_f) = o_{i1} \bigcup o_{i2} \bigcup \cdots \bigcup o_{ir}$, where o_{il} is disjoint with o_{ij} in temporal space for any $1 \le l \ne j \le r$.

Apparently, the number of the observations collected by a sensor node are determined by the variation of the monitored process or event. Since the variation of a process or event always follows certain laws, the number of the observations collected by a sensor node is limited. Let $m_i^{(o)}$ be the number of all the possible observations collected by sensor node i. We assume that the training data of each sensor node $i(1 \le i \le n)$ is large enough and can cover all the observations. The case that the training data are insufficient will be considered in our future work due to space limitation.

Meanwhile, we found that a process or event is always reflected by a series of states in most applications. For example, in a fire detection system, there are three states, representing normal, risk and fire respectively. Thus, we use $S_1, S_2, \ldots, S_k$ to denote the states of our system. As the monitoring processes of IoT systems are complicates, we regard $S_1, S_2, \ldots, S_k$ as hidden states.

Obviously, there exists a certain relationship between a hidden state and an observation. Meanwhile, two different states are related to each other. In most monitoring systems, the Markov property is guaranteed [23,24], *i.e.* a current state is only determined by the previous one, and the current observation only depends on the current state. Thus, we can construct the integration model of the multi-modal sensory data based on the Hidden Markov Process [21].

According to the above analysis, let $\mathcal{F}$ be the *integration model of multi-modal sensory data*. $\mathcal{F}$ uses an $m_i^{(o)} \times k$ matrix, B_i $(= [b_{pq}]_{1 \le p \le m_i^{(o)}, 1 \le q \le k})$, to describe the relationship between states and observations of sensor node $i(1 \le i \le n)$, and uses a $k \times k$ matrix, A $(= [a_{ij}]_{k \times k})$, to represent relationship among states, and $B_i(1 \le i \le n)$ and A satisfies that: 1). $A(p, q) = \Pr\{z_t = S_q | z_{t-1} = S_p\}$ for any $1 \le p$ and $q \le k$; 2). $B_i(p, r) = \Pr\{x_t = o_{ir} | z_t = S_p\}$ for any $1 \le r \le m_i^{(o)}$ and $1 \le p \le k$. where $\Pr\{X\}$ denotes the probability of random event X, x_t and z_t are random variables, t denotes the current time slot and $t-1$ denotes the previous time slot exactly before t. That is, A and $B_i(1 \le i \le n)$ are the *transition probability matrix* and *emission probability matrix* of $\mathcal{F}$, respectively.

From the above analysis, the model $\mathcal{F}$ will project the sensory data streams with different modalities into the sequences of states firstly and then all the computations are implemented on states instead of the raw sensory data Therefore, $\mathcal{F}$ needs to be constructed firstly. The problem of learning $\mathcal{F}$ based on the training data is defined as follows.

Input:

(1) Data streams in a long time window $[T_s, T_f]$, $\{D_i(T_s, T_f) | 1 \le i \le n\}$;

(2) The hidden states $\{S_1, S_2, \ldots, S_k\}$.

Output:

(1) The observation sets and observation sequences of n sensor nodes;

(2) The transition and emission probability matrices, A, $B_i(1 \le i \le n)$. □

Finally, the problem of cooperative event detection is took as an example to show how to use $\mathcal{F}$, which is defined as follows.

Input:

1. Current time window $[T_s^{(c)}, T_f^{(c)}]$;
2. Sensory data streams from n sensor nodes in $[T_s^{(c)}, T_f^{(c)}]$, *i.e.* $\{D_i(T_s^{(c)}, T_f^{(c)}) | 1 \le i \le n\}$;
3. $\mathcal{F} = \{A, B_1, B_2, \ldots, B_n\}$.

Output: The probability of the event being happens. □

The symbols that used in the paper is summarized in Table 1.

3. Integration model learning algorithm

Two sub problems need to be solved in order to learn model $\mathcal{F}$ according to the training data sets:

1) How to retrieve the observation set and observation sequence from a continuous sensory data stream?

2) How to learn the *transition probability matrix* A and the *emission probability matrices* $\{B_i | 1 \le i \le n\}$ according to the observations?

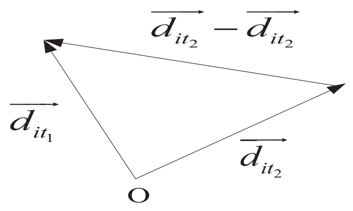

$\overrightarrow{d_{it_2}} - \overrightarrow{d_{it_1}}$
$\overrightarrow{d_{it_1}}$
$\overrightarrow{d_{it_2}}$
O

Fig. 1. The distance between two snapshots.

The following two subsections provide the solutions to the above problems. Considering the distributed properties of IoT systems, all the algorithm proposed in the rest sections are also distributed, so that the data processing abilities of each sensor node are utilized sufficiently comparing with the centralized algorithms. Meanwhile, lots of energy will be saved if we adopt the distributed algorithms in a IoT system since fewer data are required to be transmitted in the network comparing with the centralized ones.

3.1. Observation determination algorithm

The observations can be determined by each sensor node locally according to its training data set.

3.1.1. The simple method

According to Section 2, the training data set of sensor node i is denoted by $D_i(T_s, T_f)$. If the corresponding states of each sensory data stream are available, the method for determining the observations is trivial.

Suppose that $(S_1^{(i)}, S_2^{(i)}, \ldots, S_{m_i}^{(i)})$ denotes the state sequence corresponding to $D_i(T_s, T_f)$, and $t_1, t_2, \ldots, t_{m_i}$ is the time sequence of state changing, *i.e.* $t_q(2 \le q \le m_i)$ is the time instance at which the state changes from $S_{q-1}^{(i)}$ to $S_q^{(i)}$, where $t_1 = T_s$. Therefore, $t_1, t_2, \ldots, t_{m_i}$ divide the data stream $D_i(T_s, T_f)$ into m_i parts. We use $o_{i1}, o_{i2}, \ldots, o_{im_i}$ to denote these parts, then o_{ir} can be regarded as an observation for all $1 \le r \le m_i$. Therefore, the original observation set can be determined by $O_i = \{o_{i1}, o_{i2}, \ldots, o_{im_i}\}$, and the original observation sequence satisfies $\overrightarrow{O_i} = (o_{i1}, o_{i2}, \ldots, o_{im_i})$.

Apparently, duplicate observations in set O_i, which are regarded as redundant information, should be removed in order to save space and time costs for constructing model $\mathcal{F}$. To reduce the redundant observations, the similarity of two observations needs to be evaluated. Since an observation contains a group of snapshots as shown in Definition 1, the distance between any two snapshots is required firstly.

Let d_{it_1} and d_{it_2} denote two snapshots sampled by sensor i, and $Dis(d_{it_1}, d_{it_2})$ be the distance between d_{it_1} and d_{it_2}. Then, $Dis(d_{it_1}, d_{it_2}) = |d_{it_1} - d_{it_2}|$ if the sensory data sampled by sensor i is scalar data, $Dis(d_{it_1}, d_{it_2}) = ||\overrightarrow{d_{it_1}} - \overrightarrow{d_{it_2}}||_2$ if the sensory data sampled by sensor i is vector data, as shown in Fig. 1, or $Dis(d_{it_1}, d_{it_2})$ can be determined by the Euclidean distance between two images [25] if the sensory data sampled by i is video data.

Based on the distance between two snapshots, the distance between two observations is defined as follows.

Definition 2 (Distance between two Observations)**.** Let o_{iq} and o_{ir} be two observations of sensor i. The minimum weighted edit distance, denoted by $ED(o_{iq}, o_{ir})$, is used to denote the distance between o_{iq} and o_{ir}, where

1. the weight of modifying d_{it_1} to d_{it_2} equals $Dis(d_{it_1}, d_{it_2})$;
2. the weight of deleting and inserting d_{it_1} equals $Dis(d_{it_1}, \mathbf{0})$;

for any $d_{it_1} \in o_{iq}$ and $d_{it_2} \in o_{ir}$.

Table 1
Symbol list.

Symbol	Description
$i(1 \leq i \leq n)$	ID of Sensor Node
D_i	Sensory data stream sampled by sensor i
d_{it}	The snapshot value sampled by sensor i at time t
$[T_s, T_f]$	The given time window
o_{il}	An observation of sensor i
$m_i^{(o)}$	Number of all the possible observations collected by i
$\mathcal{F}$	Integration model of multi-modal sensory data
$S_1, S_2, \ldots, S_k$	The hidden states
$A\ (=[a_{ij}]_{k\times k})$	Transition probability matrix
$B_i\ (=[b_{pq}]_{1\leq p\leq m_i^{(o)}, 1\leq q\leq k})$	Emission probability matrix of Sensor i
$O_i = \{o_{i1}, o_{i2}, \ldots, o_{im_i}\}$	Original observation set corresponding to $D_i(T_s, T_f)$
$\overrightarrow{O_i} = (o_{i1}, o_{i2}, \ldots, o_{im_i})$	Original observation sequence corresponding to $D_i(T_s, T_f)$
$\overrightarrow{S^{(i)}} = (S_1^{(i)}, S_2^{(i)}, \ldots, S_{m_i}^{(i)})$	State sequence corresponding to $D_i(T_s, T_f)$
$Dis(d_{it_1}, d_{it_2})$	Distance between two observations

Based on Definition 2, the minimum weighted edit distance between any two observations are calculated firstly, and the observations are merged together if their distance is smaller than b, where b is a threshold specified by users and the Needleman-Wunsch algorithm in [26] is used to calculate the minimum weight edit distance. The detail algorithm is given is shown in Algorithm 4 in the appendix.

To determine the original observation set, we only need a sequentially scan according to the corresponding states, so the computation cost is $O(m_i)$.

To reduce the redundant observations, the average computation cost is equal to $O((m_i)^2 l_{avg}^2)$ since we need to calculate the minimum weight edit distance of $m_i(m_i-1)$ pairs of observations, and the average computation complexity for calculating the minimum weight edit distance of one pair is $O(l_{avg}^2)$ according to [26], where l_{avg} denotes the average number of snapshots contained by an observation.

3.1.2. Similarity based method

The method introduced above is efficient and has high accuracy. However, in some applications, the corresponding states of each sensory data stream are hard to be obtained even for the training data set, because these states are hidden and cannot be observed directly. Therefore, we introduce another similarity based method for this case.

Since we do not have any additional information except $D_i(T_s, T_f)$, one feasible way to determine the observation is based on the similarity between each pair of snapshots. Before introducing the algorithm, we first give the definition of a division and the inductive distance of a division for clarity.

Definition 3 (Division)**.** $\{o_{i1}, o_{i2}, \ldots, o_{il}\}$ is a division of $D_i(T_s, T_f)$ *iff.*

1. $o_{i1}, \ldots, o_{il}$ are observations that satisfy Definition 1;
2. $o_{i1} \bigcup o_{i2} \bigcup \ldots \bigcup o_{il} = D_i(T_s, T_f)$ and o_{ix} and o_{iy} are disjoint in temporal space for any $1 \leq x \neq y \leq l$. □

Definition 4 (The length and inductive distance of a division)**.** Let $\{o_{i1}, o_{i2}, \ldots, o_{il}\}$ be a division of $D_i(T_s, T_f)$. The length of division $\{o_{i1}, o_{i2}, \ldots, o_{il}\}$ is equal to l and the inductive distance of the division $\{o_{i1}, o_{i2}, \ldots, o_{il}\}$, denoted by $ID(o_{i1}, o_{i2}, \ldots, o_{il})$, satisfies that

$ID(o_{i1}, o_{i2}, \ldots, o_{il}) = \max\{Dis(d_{it_1}, d_{it_2}) \mid d_{it_1}, d_{it_1} \in o_{ix} \bigwedge 1 \leq x \leq m\}$. □

Next, we consider two cases for observation determination, and the users can select one according to the applications.

Case 1. According to the algorithm in the above section, it requires to compare each pair of the observations in the original observation sequence to identify the redundant ones. Therefore, the length of the original observation sequence determined by a division should be as small as possible so that fewer comparisons are needed. Due to such intuition, we required to find the division of $D_i(T_s, T_f)$ whose length is minimized on condition that inductive distance is no more than b_1, where b_1 is a given threshold. The formal definition of such problem is defined as follows.

$$Min|\overrightarrow{O_i}|$$

$$s.t.\ Dis(d_{i.t_{l_j}}, d_{i.t_{l_r}}) \leq b_1 \text{ for any } o_{ip} \in \overrightarrow{O_i} \text{ and } d_{i.t_{l_j}}, d_{i.t_{l_r}} \in o_{ip}.$$

Such a problem can be solved by a greedy algorithm, which consists of fours steps. First, let $l=1$. Then, scan $D_i(T_s, T_f)$ sequentially, and insert the snapshots to observation o_{il} until the distance between the new coming snapshot with any one snapshot in o_{il} being larger than b_1. Third, let $l=l+1$ and repeat the second step until we reach the end of $D_i(T_s, T_f)$. Finally, call the algorithm in Section 3.1.1 to remove the redundant observations in the observation set and replace them in the observation sequence.

The detail algorithm is given in Algorithm 5 in the appendix.

Case 2. In some applications, the length of an observation should not be too large in order to catch every variance of the monitoring object accurately. On the other hand, the length of an observation should not be too small as well since the corresponding states of the monitored objects usually are limit according to the analysis in Section 2. Due to such reasons, a set of consecutive snapshots is regarded to contain multiple observations and should be divided recursively if its size is larger than b_2, however, it is regarded as an observation and cannot be divided again if its size is smaller than or equal to b_2, where b_2 is a given threshold. Under such an assumption, the inductive distance of the division is required to be minimized. Specifically, the problem of determining the required division is formalized as follows.

Min $\max\{Dis(d_{it_1}, d_{it_2}) \mid d_{it_1}, d_{it_1} \in o_{il} \bigwedge o_{il} \in O_i\}$ such that for each $o_{il} \in O_i$,

1. o_{il} is an observation and satisfies Definition 1;
2. o_{il} and o_{iw} are disjoint and $\bigcup_{o_{il} \in O_i} o_{il} = D_i(T_s, T_f)$, where o_{iw} is any other observation in O_i;
3. $|o_{il}| \leq b_2$, $|o_{il}| + |o_{i(l+1)}| > b_2$, $|o_{i(l-1)}| + |o_{il}| > b_2$, where $o_{i(l-1)}$, o_{il} and $o_{i(l+1)}$ are arbitrary three consecutive observations in O_i.

Such a problem can be solved by a dynamic programming method. Let $\alpha[q, r]$ denote the subset of $D_i(T_s, T_f)$ that contains the q-th, $(q+1)$-th, $(q+2)$-th,... ,r-th snapshots in $D_i(T_s, T_f)$, where $1 \leq q < r \leq |D_i(T_s, T_f)|$. The optimal division of $\alpha[q, r]$ is defined as follows.

Definition 5 (Optimal Division)**.** $\{o_{il_1}, o_{il_2}, \ldots, o_{il_v}\}$ is the optimal division of $\alpha[q, r]$ if and only if

1. $\{o_{il_1}, o_{il_2}, \ldots, o_{il_v}\}$ is the division of $\alpha[q, r]$ that satisfies Definition 2;

2. $|o_{il_x}| \le b_2$, $|o_{il_x}| + |o_{il_{x+1}}| > b_2$, $|o_{il_{x-1}}| + |o_{ix}| > b_2$, where $1 \le x \le v$
3. for any other division of $\alpha[q, r]$, $\{o'_{il_1}, o'_{il_2}, \ldots, o'_{il'_v}\}$, which satisfies condition (1) and (2), we have the following Formula (1).

$$\max\{Dis(d_{it_1}, d_{it_2}) \mid d_{it_1}, d_{it_2} \in o_{il_x} \wedge 1 \le x \le v\} \le \max\{Dis(d_{it_1}, d_{it_2}) \mid d_{it_1}, d_{it_2} \in o'_{il_x} \wedge o'_{il_x} \in \{o'_{il_1}, \ldots, o'_{il'_v}\}\} \tag{1}$$

Let $ID[q, r]$ denote the inductive distance of the optimal division of $\alpha[p, r]$, *i.e.*, $ID[q, r] = \max\{Dis(d_{it_1}, d_{it_2}) \mid d_{it_1}, d_{it_2} \in o_{il_x} \wedge 1 \le x \le v\}$. Therefore, the following dynamic programming function is obtained.

$$ID[q, r] = \begin{cases} \max\limits_{q \le k \le r}\{ID[q, k], ID[k, r]\} & \text{if } |r - q| > b_2 \\ \max\{Dis(d_{it_1}, d_{it_2}) \mid d_{it_1}, d_{it_2} \in \alpha[q, r]\} & \text{Otherwise} \end{cases} \tag{2}$$

The dynamic programming function needs to be solved so that the original observation set and sequence, O_i and $\overrightarrow{O_i}$, can be determined. Finally, The algorithm given in Section 3.1.1 will be used to reduce the redundant information in O_i and $\overrightarrow{O_i}$. Since the length of any observation is bounded (less than b_2), the computation cost of removing the redundant observations is also controllable.

3.2. The algorithms of determining transition and emission probability matrices

Let $S_1, S_2, \ldots, S_k$ denote hidden states, and $\overrightarrow{O_1}, \overrightarrow{O_2}, \ldots, \overrightarrow{O_n}$ be the observation sequences retrieved from n sensor nodes by the method in Section 3.1. Then, the remaining problem for constructing model $\mathcal{F}$ is to determine the transition probability matrix A and the emission matrices $\{B_i | 1 \le i \le n\}$. Similar to Section 3.1, there are two cases that need to be considered.

3.2.1. The maximum likelihood based algorithm

First, if the corresponding states of each sensory data stream are available, it is easy to determine the transition and emission probability metrics.

Suppose $\overrightarrow{S^{(i)}} = (S_1^{(i)}, S_2^{(i)}, \ldots, S_{m_i}^{(i)})$ denote the state sequence corresponding to $D_i(T_s, T_f)$, and $\overrightarrow{O_i} = (x_1, x_2, \ldots, x_{m_i})$ denote the observation sequence identified by the algorithm in Section 3.1. Therefore, for each sensor node $i (1 \le i \le n)$, the problem of determining the local transition and emission probability matrices, A_i and B_i, can be formalized as follows according to the maximum likelihood estimation [27].

$$A_i, B_i = \arg\max\nolimits_{A,B} \Pr(\overrightarrow{S^{(i)}}, \overrightarrow{O_i} | A, B) \tag{3}$$

such that

1. $A(p, q) \ge 0$ and $\sum_{q=1}^{k} A(p, q) = 1$ for all $p, q \in [1, k]$;
2. $B(p, v) > 0$ and $\sum_{v=1}^{m_i^{(o)}} B(p, v) = 1$ for all $p \in [1, k]$ and $1 \le v \le m_i^{(o)}$.

where $m_i^{(o)} = |O_i|$ denotes the number of the observations in the observation set, and $M(p, q)$ is the element in the p-th row and q-th column of matrix M.

Theorem 1. *A_i and B_i are the solution of the problem given in Formula (3) if* $A_i(p, q) = \frac{\sum_{t=1}^{m_i} I(S_t^{(i)} = S_q \wedge S_{t-1}^{(i)} = S_p)}{\sum_{t=1}^{m_i} I(S_{t-1}^{(i)} = S_p)}$ *and* $B_i(q, v) = \frac{\sum_{t=1}^{m_i} I(S_t^{(i)} = S_q \wedge x_t = o_{iv})}{\sum_{t=1}^{m_i} I(S_t^{(i)} = S_q)}$ *for all* $1 \le q,\ p \le k$ *and* $1 \le v \le m_i^{(o)}$, *where* $I(X)$ *is an indicate function, i.e.* $I(X) = 1$ *if random event* X *is true, otherwise* $I(X) = 0$. □

The proof of Theorem 1 is given in the appendix. Based on Theorem 1, the local transition and emission probability matrices can be determined by each sensor node itself. For each local emission probability matrix, it can be stored locally and does not need to be transmitted to the sink since it only describes the relationship between the hidden states and the sensor's own observations. However, for each local transition probability matrix, it needs to be transmitted to the sink as a global transition probability matrix is required to integrate the multi-modal sensory data from different sensor nodes.

Let A_i denote the local transition probability matrix obtained by sensor node $i (1 \le i \le n)$. The global transition probability matrix (A) can be constructed by $A(p, q) = \frac{\sum_{i=1}^{n} A_i(p.q)}{\sum_{i=1}^{n} \sum_{j=1}^{k} A_i(p,k)}$. *Since* $\sum_{j=1}^{k} A_i(p, k) = 1$ *for all* $1 \le i \le n$, $A(p, q) = (\sum_{i=1}^{n} A_i(p.q))/n$.

The algorithm of determining the transition and emission probability matrices is given in Algorithm 6in Appendix. The communication cost of the algorithm is $O(k^2)$ since the local transition probability matrix needs to be transmitted and aggregated along the spanning tree towards the sink. The computation complexity is $O(\max\{k^2 m_i, k m_i^{(o)} m_i\})$ since the appearance times of each pair of two states and each pair of a state and an observation need to be counted.

3.2.2. The EM algorithm

When the corresponding states of each data stream are unknown, we will construct the transition and emission probability matrices distributely based on the EM algorithm.

Let $\overrightarrow{O_i} = (x_1, x_2, \ldots, x_{m_i})$ be the observation sequence obtained by sensor node $i (1 \le i \le n)$ during $[T_s, T_f]$ according to the method in Section 3.1. Since the EM algorithm is an iteration method, let $A_i^{(r)}$ and $B_i^{(r)}$ $(1 \le i \le n)$ denote the local transition and emission probability matrices after r iterations, where $A_i^{(0)}$ and $B_i^{(0)}$ are the initial matrices.

Let $\overrightarrow{z_i} = (z_{i1}, z_{i2}, \ldots, z_{im_i})$ be the random vectors to denote the sequence of states corresponding to $\overrightarrow{O_i}$. The aim of the EM algorithm is to maximum the expected value of the log-likelihood function, which is given as follows

$$\begin{aligned} Q(A, B; A_i^{(r-1)}, B_i^{(r-1)}) &= E\left[\log \Pr(\overrightarrow{z_i}, \overrightarrow{O_i} | A, B) \middle| \overrightarrow{O_i}, A_i^{(r-1)}, B_i^{(r-1)}\right] \\ &= \sum_{\overrightarrow{z_i} \in S^{m_i}} \log \Pr(\overrightarrow{z_i}, \overrightarrow{O_i} | A, B) \Pr(\overrightarrow{z_i} | \overrightarrow{O_i}, A_i^{(r-1)}, B_i^{(r-1)}) \end{aligned}$$

where m_i is the length of sequence $\overrightarrow{O_i}$, and $m_i \ge |O_i| = m_i^{(o)}$ since there may exist some redundant observations in $\overrightarrow{O_i}$.

Let $\varphi(\overrightarrow{z_i}) = \Pr(\overrightarrow{z_i} | \overrightarrow{O_i}, A_i^{(r-1)}, B_i^{(r-1)})$, then $Q(A, B; A_i^{(r-1)}, B_i^{(r-1)}) = \sum_{\overrightarrow{z_i} \in S^{m_i}} \log \Pr(\overrightarrow{z_i}, \overrightarrow{O_i} | A, B) \varphi(\overrightarrow{z_i})$. Thus, the problem of determining $A_i^{(r)}$ and $B_i^{(r)}$ iteratively can be formalized as

$$\begin{aligned} A_i^{(r)}, B_i^{(r)} &= \arg\max_{A,B} Q(A, B; A_i^{(r-1)}, B_i^{(r-1)}) \\ &= \arg\max\nolimits_{A,B} \sum\nolimits_{\overrightarrow{z_i} \in S^{m_p}} \log \Pr(\overrightarrow{z_i}, \overrightarrow{O_i} | A, B) \varphi(\overrightarrow{z_i}) \end{aligned} \tag{4}$$

such that

1. $A(p, q) \ge 0$ and $\sum_{q=1}^{k} A(p, q) = 1$ for all $p, q \in [1, k]$;
2. $B(p, v) > 0$ and $\sum_{v=1}^{m_i^{(o)}} B(p, v) = 1$ for all $p \in [1, k]$ and $1 \le v \le m_i^{(o)}$.

Theorem 2. *$A_i^{(r)}$ and $B_i^{(r)}$ are the solutions of the above problem if* $A_i^{(r)}(p, q) = \frac{\sum_{t=1}^{m_i} \eta_t(p,q)}{\sum_{p=1}^{k} \sum_{t=1}^{m_i} \eta_t(p,q)}$ *and* $B_i^{(r)}(q, v) = \frac{\sum_{p=1}^{k} \sum_{t=1}^{m_i} I(x_t = o_{iv}) \eta_t(p,q)}{\sum_{p=1}^{k} \sum_{t=1}^{m_i} \eta_t(p,q)}$ *for any* $1 \le p, q \le k$ *and* $1 \le v \le |O_i| = m_i^{(o)}$, *where* $\eta_t(p, q) = \beta_p(t-1) A_i^{(r-1)}(p, q) B_i^{(r-1)}(q, x_t) \gamma_q(t)$,

$\beta_p(t) = \Pr(x_1, x_2, \ldots, x_t, z_{it} = S_p | A_i^{(r-1)}, B_i^{(r-1)})$ *and* $\gamma_q(t) = \Pr(x_{t+1}, \ldots, x_{m_i-1}, x_{m_i}, z_{it} = S_q | A_i^{(r-1)}, B_i^{(r-1)})$. □

The proof of the above theorem is give in Appendix, and according to it, we need to determine $\beta_p(t)$ $(= \Pr(x_1, x_2, \ldots, x_t, z_{it} = S_p | A_i^{(r-1)}, B_i^{(r-1)}))$ *and* $\gamma_q(t)$ $(= \Pr(x_{t+1}, \ldots, x_{m_i-1}, x_{m_i}, z_{it} = S_q | A_i^{(r-1)}, B_i^{(r-1)}))$ *firstly. Fortunately,* $\beta_p(t)$ *and* $\gamma_q(t)$ *can be determined by the forward and backward procedures. The algorithms are given in Algorithms 1 and 2 where* $(\pi_1, \pi_2, \ldots, \pi_k)$ *denote the initial distribution of the states, which can be determined according to the background knowledge of the application. Otherwise, we can set* $\pi_p = 1/k$ *for* $1 \le p \le k$.

Based on the above algorithms and Theorem 2, the EM algorithm for determining the transition and emission probability matrices is presented as follows.

Step 1. All the sensors in the network are organized as a spanning tree rooted at the sink. The sink broadcasts the initial transition probability matrix $A^{(0)}$ along the spanning tree to the network.

Step 2. Each sensor $i(1 \le i \le n)$ initializes the local emission probability matrix $B_i^{(0)}$, and sets $A_i^{(0)}$ to be $A^{(0)}$ and $r = 1$, where r is the iteration times.

Step 3. For all $1 \le p, q \le k$ and $1 \le t \le m_i$, sensor nodes $i(1 \le i \le n)$ calculates $\beta_p(t-1)$, $\gamma_q(t)$ according to Algorithms 1 and 2, then it computes $\eta_t(p, q)$ by $\eta_t(p,q) = \beta_p(t-1) A_i^{(r-1)}(p,q) B_i^{(r-1)}(q, x_t) \gamma_q(t))$.

Step 4. Sensor i determines $A_i^{(r)}$ and $B_i^{(r)}$ by $A^{(r)}(p,q) = \frac{\sum_{t=1}^{m_i} \eta_t(p,q)}{\sum_{p=1}^{k} \sum_{t=1}^{m_i} \eta_t(p,q)}$ and $B_i^{(r)}(q, v) = \frac{\sum_{p=1}^{k} \sum_{t=1}^{m_i} I(x_t = o_{iv}) \eta_t(p,q)}{\sum_{p=1}^{k} \sum_{t=1}^{m_i} \eta_t(p,q)}$ for all $1 \le p, q \le k$ and $1 \le v \le m_i^{(o)}$, where $m_i^{(o)} = |O_i|$ denotes the number of the observations discovered in Section 3.1. Let $r = r + 1$.

Step 5. Step 3 and Step 4 are repeated iteratively until r exceeds R times or $\max_{1 \le p,q \le k} \{|A_i^{(r)}(p,q) - A_i^{(r-1)}(p,q)|\} \le \epsilon_1 \wedge \max_{1 \le q \le k, 1 \le v \le m_i^{(o)}} \{|B_i^{(r)}(p,q) - A_i^{(r-1)}(p,q)|\} \le \epsilon_2$, where R, ϵ_1 and ϵ_2 are given thresholds.

Step 6. Sensor node i transmits $A_i^{(r)}$ along the spanning tree towards the sink when the iteration is ended. $\{A_i^{(r)} | 1 \le i \le n\}$ are added together during the transmission. Finally, the sink determines A by $A = \sum_{i=1}^{n} \frac{1}{n} A_i^{(r)}$.

The communication cost of the algorithm is $O(k^2)$ since the $A^{(0)}$ needs to be broadcasted in Step 1, and the local transition probability matrices need to be transmitted towards the sink in Step 6. The computation complexity in each iteration is $O(k^2 m_i)$ since it needs to calculate $\eta_t(p, q)$ for all $1 \le p, q \le k$ and $1 \le t \le m_i$. Thus, the maximum computation cost of the above algorithm is equal to $O(Rk^2 m_i)$.

3.3. Discussion

To determine the Integration Model, $\mathcal{F}$, of multi-modal sensory data, we hae proposed three Observation Determination Algorithms and two algorithm for calculating the Transition and Emission Probability Matrices.

Among these algorithms, the simple observation determining method introduced in Section 3.1.1 and the Maximum Likelihood based algorithm given in Section 3.2.1 are more efficient since they does not require iterated computation. However, more detailed information,*e.g.* the corresponding states of each sensory data stream in training set, are also required by these algorithms.

On the other hand, although the algorithms introduced in Sections 3.1.2 and 3.2.2 are more complex and consume more computation resource for determining the observations, transition and emission probability matrices, the input information required by them are much fewer, so that these algorithms are suitable to deal with the situation that the limited information is available during training the integration model $\mathcal{F}$. Moreover, the greedy and dynamic programming method mentioned in Section 3.1.2 are designed for different optimal goals. Therefore, the users are able to choose any of the above algorithms adaptively based on the situation they have.

4. Case study: a cooperative event detection

Using the algorithms introduced in Section 3, the model for integrating multi-modal sensory data, denoted by $\mathcal{F}$, can be learned. Next, we will discuss the problem of how to use $\mathcal{F}$ for supporting the fusing computation. The following section takes the cooperation event detection as an example for studying such problem, and the reasons are as follows. First, the event detection is one of the most important and primary applications for sensor networks and IoT systems. Second, the event detection is sensitive on time and energy consumption, while the latency and transmission cost are dramatically reduced with the cooperation of the multi-modal sensory data since they provide more abundant information about the monitoring objects. Therefore, the efficiency of the event detection is largely improved with the help of the multi-modal sensory data, which is also verified in our experimental results. Due to the space limitation, the other fusing computation will be considered in our further works.

To utilize $\mathcal{F}$ for cooperative event detection, there still exist two problems that need to be solved:

1). How to identify the observations contained in current data streams?

2). How to deduce the most likely sequence of the states?

4.1. Observation identification algorithm

For each sensor node $i(1 \le i \le n)$, let $D_i(T_s^{(c)}, T_f^{(c)})$ denote the sensory data stream collected in the current time window $[T_s^{(c)}, T_f^{(c)}]$, and O_i denote its observation set with size $m_i^{(o)}$. Then, the problem of identifying the observations in $D_i(T_s^{(c)}, T_f^{(c)})$ can be defined as

$$\overrightarrow{O_i^{(c)}} = \arg\min_{\vec{O}} ED(D_i(T_s^{(c)}, T_s^{(f)}), \vec{O}) \qquad (5)$$

such that

1. $\overrightarrow{O_i^{(c)}} = (o_{ir_1}, o_{ir_2}, \ldots, o_{ir_l})$, $l \ge 1$, and
2. $o_{ir_1}, o_{ir_2}, \ldots, o_{ir_l} \in O_i$, where O_i is the set of the observations determined in integration model learning process (*i.e.* the algorithms in Section 3.1)

where $ED(D_i(T_s^{(c)}, T_s^{(f)}), \vec{O})$ denotes the minimum weight edit distance of two sequences.

Let $\overrightarrow{O_i^{(c)}}$ be the optimal solution of the problem given by Formula (5), and $\{(x_p, y_{ps}^{(l)}, y_{pf}^{(l)}) \mid 1 \le p \le m_i^{(o)},\ 1 \le l \le x_p\}$ satisfy that x_p denotes the times of o_{ip} appearing in $\overrightarrow{O_i^{(c)}}$, $y_{ps}^{(l)}$ and $y_{pf}^{(l)}$ is the start and end position of l-th appearance of o_{ip} in $\overrightarrow{O_i^{(c)}}$. If $o_{ip} \notin \overrightarrow{O_i^{(c)}}$, $x_p = 0$. Therefore, the problem in Formula (5) can be formalized as an integer programming problem as follows.

$$Min \sum_{p=1}^{m_i^{(o)}} \sum_{l=1}^{x_p} ED(D_i(y_{ps}^{(l)}, y_{pf}^{(l)}), o_{ip}) \qquad (6)$$

such that

(1) x_p is an integer in range $[0, |D_i(T_s^{(c)}, T_f^{(c)})|]$ for all $1 \le p \le m_i^{(o)}$;

(2) $\{y_{ps}^{(l)} | 1 \le l \le x_p\}$ and $\{y_{pf}^{(l)} | 1 \le l \le x_p\}$ are integers in range $[0, |D_i(T_s^{(c)}, T_f^{(c)})|]$ for all $1 \le p \le m_i^{(o)}$;

(3) $\exists p \in [1, m_i^{(o)}]$ satisfies that $x_p > 0$ and $y_{pf}^{(l)} > y_{ps}^{(l)}$ for all $1 \le l \le x_p$;

(4) $\exists p \in [1, m_i^{(o)}]$ satisfies that $y_{ps}^{(1)} = 1$ and $\exists q \in [1, m_i^{(o)}]$ satisfies that $y_{qf}^{(x_q)} = |D_i(T_s^{(c)}, T_f^{(c)})|$;

(5) $\exists q \in [1, m_i^{(o)}]$ and $l_2 \in [1, x_q]$ satisfies that $y_{pf}^{(l_1)} = y_{qs}^{(l_2)}$ if $y_{pf}^{(l_1)} \neq |D_i(T_s^{(c)}, T_f^{(c)})|$ for $\forall p \in [1, m_i^{(o)}]$ and $\forall l_1 \in [1, x_p]$;

(6) $y_{qf}^{(l_2)} \le y_{ps}^{(l_1)} \le y_{pf}^{(l_1)}$ OR $y_{ps}^{(l_1)} \le y_{pf}^{(l_1)} \le y_{qs}^{(l_2)}$ for $\forall p, q \in [1, m_i^{(o)}]$, $\forall l_1 \in [1, x_p]$ and $\forall l_2 \in [1, x_q]$.

Since the integer programming problem is NP-hard, it is also hard to compute the optimal solution of the problem presented in Formula (5). The naive method to deal with the problem is the enumerating algorithm. Suppose that the current time window is small enough so that there are at most h observations in it. Then, the enumerating algorithm enumerates $\sum_{r=1}^{h} \left(m_i^{(o)}\right)^r$ observation sequences to form the candidate set, where $m_i^{(o)}$ is the size of the observation set that is determined in integration model learning process. After that, the weight edit distance between $D_i(T_s^{(c)}, T_f^{(c)})$ and each candidate sequence is calculated, and the one with the smallest weight edit distance is chosen and returned.

The computation complexity of the enumerating algorithm is equal to $O(|D_i(T_s^{(c)}, T_f^{(c)})|^2 \left(m_i^{(o)}\right)^h)$. It is unacceptable when the time window is large. Another heuristic algorithm based on a greedy strategy is provided.

The detail steps of the heuristic algorithm are as follows.

First,let $\overrightarrow{O_i^c} = ()$. Let S with l "Null" be the matching sequence, where $l = |D_i(T_s^{(c)}, T_f^{(c)})|$.

Second, for each observation o_{ip}, scan $D_i(T_s^{(c)}, T_f^{(c)})$ sequentially, find an integer λ_{ps} satisfies that the sub stream $D_i(\lambda_{ps}, \lambda_{ps} + |o_{ip}|)$ has the smallest weight edit distance with o_{ip}. If there exists multiple integers satisfy the above condition, then let λ_{ps} be the smallest one.

Third, select an observation o_{iq} from O_i satisfying that $ED(D_i(\lambda_{qs}, \lambda_{qs} + |o_{iq}|), o_{iq})/|o_{iq}|$ is smallest among all observations in O_i. If there exists multiple observations satisfies the above ones, random select one.

Fourth, update the sub sequence of S whose start and end positions are λ_{qs} and $\lambda_{qs} + |o_{iq}|$ to be o_{iq}, and insert o_{iq} to $\overrightarrow{O_i^c}$ according to λ_{qs}. Change the snapshot to be "Null" in $D_i(T_s^{(c)}, T_f^{(c)})$ from position λ_{qs} to $\lambda_{qf} + |o_{iq}|$.

Fifth, for each $o_{ip} \in O_i$, if $[\lambda_{ps}, \lambda_{ps} + |o_{ip}|]$ overlaps $[\lambda_{qs}, \lambda_{qs} + |o_{iq}|]$, then recalculate λ_{ps} as shown in step 2.

Finally, repeat Step 3, Step 4 and Step 5 until the number of consecutive "Null" in S is smaller than any length of observation in O_i. Return $\overrightarrow{O_i^c}$.

Above algorithm has polynomial complexity, and is efficient to process data stream sampled in a large time window.

4.2. Deducing the most likely state sequence

Let $\overrightarrow{O_i^{(c)}} = (x_1^{(c)}, x_2^{(c)}, \ldots, x_{\tau_i}^{(c)})$, where $\overrightarrow{O_i^{(c)}}$ is the observation sequence in current time window returned by the method in Section 4.1, and τ_i be the length of $\overrightarrow{O_i^{(c)}}$. The problem of deducing most likely state sequence is defined as

$$\overrightarrow{z_i^{(c)}} = arg\max_{\overrightarrow{z}} \Pr(\overrightarrow{z}\,|\overrightarrow{O_i^{(c)}}, A, B_i) \qquad (7)$$

where $\overrightarrow{z}$ denotes the random state sequence in current time window with length τ_i, A and B_i are the transition and emission probability matrices.

Since $\sum_{\overrightarrow{z}} \Pr(\overrightarrow{O_i^{(c)}}, \overrightarrow{z}\,|A, B_i) = \Pr(\overrightarrow{O_i^{(c)}}|A, B_i)$ is a constant value when $\overrightarrow{O_i^{(c)}}$, A and B_i are given, we have

$$\overrightarrow{z_i^{(c)}} = arg\max_{\overrightarrow{z}} \Pr(\overrightarrow{z}\,|\overrightarrow{O_i^{(c)}}, A, B_i) = arg\max_{\overrightarrow{z}} \frac{\Pr(\overrightarrow{z}, \overrightarrow{O_i^{(c)}}|A, B_i)}{\Pr(\overrightarrow{O_i^{(c)}}|A, B_i)}$$
$$= arg\max_{\overrightarrow{z}} \Pr(\overrightarrow{z}, \overrightarrow{O_i^{(c)}}|A, B_i)$$

Therefore, the problem can be solved by a dynamic programming method. Let

$$\mu(j, t) = \max_{(z_{i1}, \ldots, z_{it-1}) \in S^{t-1}} \Pr(x_1^{(c)}, \ldots, x_t^{(c)}, z_{i1}, \ldots, z_{it-1}, z_{it} = S_j|A, B_i)$$

where $z_{ip}\,(1 \le p \le t)$ denotes the random state. Therefore,

$$\max_{\overrightarrow{z}} \Pr(\overrightarrow{z}\,|\overrightarrow{O_i}, A, B_i) = \max_{j=1}^{k} \mu(j, \tau_i) \qquad (8)$$

and

$$\mu(j, t) = \max_{p=1}^{k} \mu(p, t-1) A(p, j) B_i(j, x_t^c) \qquad (9)$$

Thus, $\overrightarrow{z_i^{(c)}}$ can be obtained by solving the dynamic programming function according to Formula (9). The algorithm is shown in Algorithm 6, where $\pi_p\,(1 \le p \le k)$ is the steady-state probability of the Markov process, and can be determined by transition probability matrix A.

4.3. Cooperative event detection algorithm

Based on the discussions in Sections 4.1 and 4.2, the cooperative event detection algorithm has three steps.

First, each sensor node $i\,(1 \le i \le n)$ retrieves the observation sequence from its current data stream $D_i(T_s^{(c)}, T_f^{(c)})$ using the algorithm in 4.1. It determines the most likely state sequence $\overrightarrow{z_i^{(c)}}$ using the algorithm in Section 4.2.

Second, each sensor $i\,(1 \le i \le n)$ transmits the last state in state sequence $\overrightarrow{z_i^{(c)}}$, that is $z_{i\tau_i}$, to the sink by the spanning tree routing protocol.

Third, the sink obtains $\{S_{r_1}, S_{r_2}, \ldots, S_{r_h}\} (= \bigcup_{i=1}^{n} \{z_{i\tau_i}\})$ after receiving

$\{z_{i\tau_i} \mid 1 \le i \le n\}$ from the network. Let $S_e \in \{S_1, S_2, \ldots, S_k\}$ denote the state of event e. Then, the sink calculates $\varphi = \Pr(X_{t+1} = S_e|(X_t = S_{r_1}) \bigvee (X_t = S_{r_2}) \bigvee \ldots \bigvee (X_t = S_{r_h}))$ by the following formula

$$\varphi = \frac{\sum_{i=1}^{h} \Pr(X_{t+1} = S_e \bigcap X_t = S_{r_i})}{\sum_{i=1}^{h} \Pr(X_t = S_{r_i})}$$
$$= \frac{\sum_{i=1}^{h} \Pr(X_{t+1} = S_e|X_t = S_{r_i}) \Pr(X_t = S_{r_i})}{\sum_{i=1}^{h} \Pr(X_t = S_{r_i})}$$

since $X_t = S_{r_j}$ and $X_t = S_{r_i}$ are mutually disjoint with each other, where X_t and X_{t+1} are random variables. Based on the transition probability matrix A, $\Pr(X_{t+1} = S_e|X_t = S_{r_i}) = A(e, r_i)$ and $\Pr(X_t = S_{r_i})$ can be determined by the steady-state probability of the Markov process. Thus, the probability that event e will happen in next time slot, i.e. φ, can be obtained by the sink. If φ is larger than a given threshold, the sink reports e to the users.

Since only states are required to be transmitted, the above algorithm saves a lot of energy during event detection. Meanwhile, the computation cost of the sink is $O(n)$ when executing the above algorithm, which is also very low.

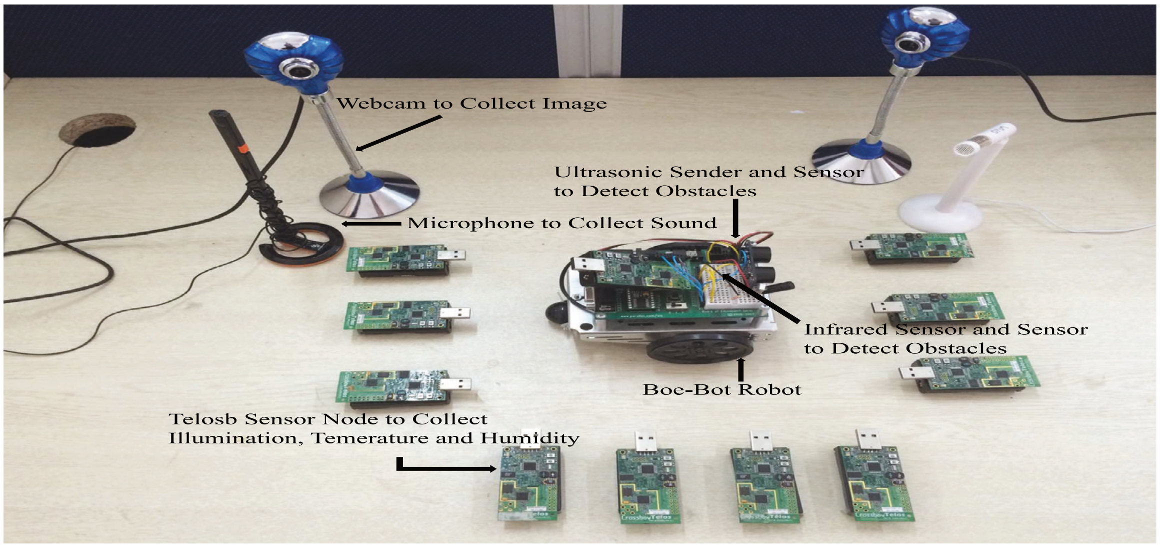

Webcam to Collect Image
Ultrasonic Sender and Sensor to Detect Obstacles
Microphone to Collect Sound
Infrared Sensor and Sensor to Detect Obstacles
Boe-Bot Robot
Telosb Sensor Node to Collect Illumination, Temerature and Humidity

Fig. 2. The devices in the system.

5. Experimental results

Two real testbeds is used to evaluate the performance of our proposed model.

The first one is an indoor intrusion detection system. It is based on TinyOS 2.1.0 and consists of two Boe-Bot Robots [28] which can move automatically according to the instructions. The ultrasonic and infrared ray sensors are deployed on the two robots to measure distance, detect obstacles and sensing the temperature from intruder. Furthermore, ten TelosB sensors are also deployed in the monitored region, which are static and can continuously sample the temperature, humidity and light intensity from the monitored area. Three of them are used for routing, one is preserved as the sink. Finally, the system also contains two camera and two microphone to catch the variation of video and audio in the monitored region. The devices used in the system is presented in Fig. 2.

The second one is the human motion monitoring system. We use five iPhones to monitor the motions of the holders. The operating system is iOS7.1. The accelerometer and the gyroscope embedded in the iPhone are used to sample the velocity and the angular velocity of the human motion. The camera and microphone are also utilized to sample the video and audio data from holders.

In these systems, the number of computation operations and transmissions is calculated while the proposed algorithms are applied. According to [29], the energy cost of a sensor to send and receive one byte is set to be 0.0144 mJ and 0.0057 mJ, respectively. The energy consumed of executing 1000 instructions of CPU for a sensor is equal to that consumed by sending a bit message.

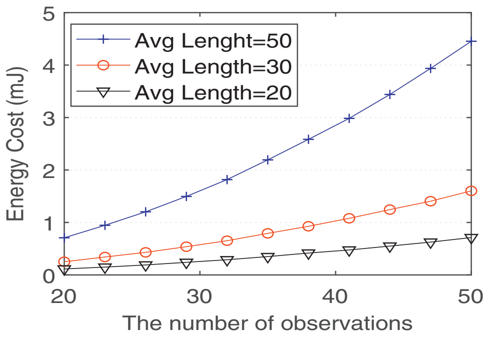


Fig. 3. Energy cost of merging observations.

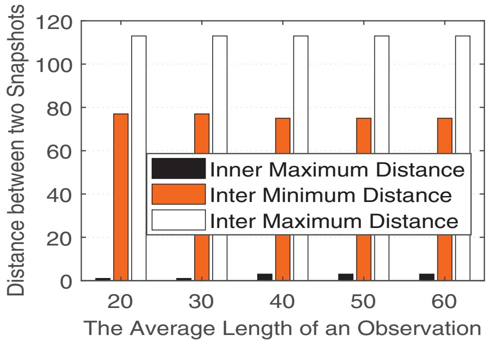


Fig. 4. Comparison on the distance of two snapshots.

5.1. *The performance of learning algorithm*

The first group experiments are to investigate the energy cost of the redundant observation reducing algorithm in Section 3.1. In the experiments, the energy consumption is calculated while the number of original observations varies from 20 to 50, and the average length of an observation is set to be 20, 30 and 50 respectively, where the length of an observation equals to the number of snapshot it contains. According to Fig. 3, it costs little energy for deleting the redundant observations from the original observation set even when its size is large since the energy consumed by the computation is quite small.

The second group experiments are to compare the distance of two snapshots inner an observation with that between different observations. In the experiments, the maximum distance of two snapshots inner an observation, the minimum and maximum distances between two snapshots in different observations were calculated while the average length of an observation increased from 20 to 60. Fig. 4 shows that the distance of two snapshots inner an observation is much smaller than that of two snapshots belonging to different observations, so that the observations in a data stream can be partitioned by similarity comparison.

The third group of the experiment is to investigate the recall and precision rate of the similarity based method in Section 3.1. In the experiments, the number of the real observations in a stream is set to be 15 and 20, the average length of an observation is set to 50 and 100. The recall rate is calculate while the given bound

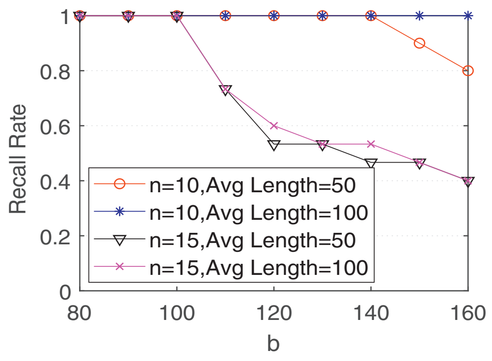


Fig. 5. Recall rate of similarity based algorithm.

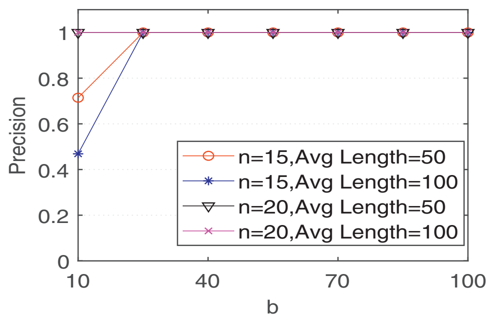


Fig. 6. Precision rate of similarity based algorithm.

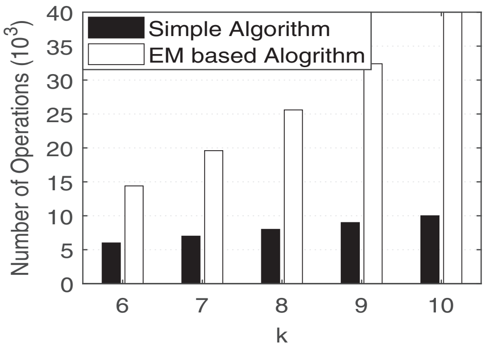


Fig. 7. Computation complexity of determining transition and emission matrices.

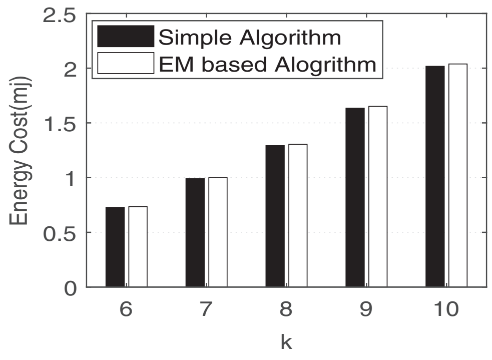


Fig. 8. Energy cost of determining transition and emission matrices.

b is increase from 80 to 160, and the precision rate is calculated while b grows from 10 to 100. The results in Fig. 5 show that the recall rate is close to 1 expect when the given bound is too large. Fig. 6 shows that the precision rate is also approached to 1 expect when the given bound is too small. Therefore, the above results indicate that the bound b is easy to set for determine the observation according to the similarity, which is because the observations corresponding to different state are easy to be distinguished according to Fig. 3.

The fourth group experiments are to investigate the computation complexity and energy cost of the transition and emission probability matrices determining algorithms. In the experiments, the number of operations and the energy cost of the simple algorithm and EM algorithm were calculated while the number of states, k, increase from 6 to 10. Fig. 7 show that EM algorithm needs more operations than the simple algorithm since its input information is much less. However, the energy consumed by the two algorithms is almost the same according to Fig. 8. since the data size transmitted by both algorithms is the same and the energy costed by computation is quite smaller than that costed by transmission for a sensor device. These results also verify that EM algorithm is energy efficient even that it can deal with more complicate situation.

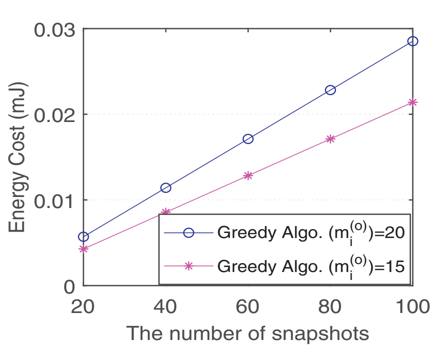
5.2. The performance of event detection algorithm

The first group experiments are to investigate the energy cost of observation identification algorithms. which are introduced in Section 4.1. In the experiments, the energy consumed by enumerating and greedy algorithms was calculated while the number of snapshots in the current time window varied from 20 to 100, and the average length of an observation is 20. The experimental results is presented in Fig. 9(a) and (b). These figures show that the energy cost of enumerating algorithm is 10^6 times more than that of greedy algorithm. Since greedy algorithm is only a polynomial time algorithm, it needs much fewer computation operations to identify observations, so that much energy is saved.

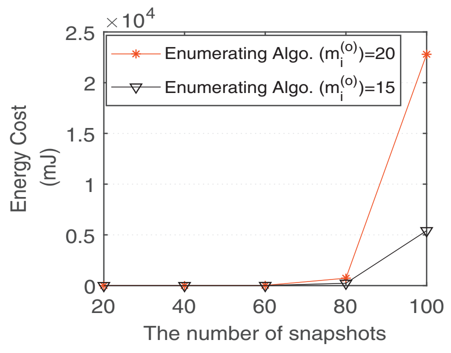
The second group experiments are to evaluate the ratio bound of the greedy algorithm. In the experiments, the relative weighted edit distances between the observation sequences and the original data stream were calculated while the number of snapshots increased from 21 to 103, and the average length of an observation is 20, where the observation sequences are returned by the enumerating and greedy algorithm, respectively. The experimental results are given in Fig. 9(c). It shows that the relative weighted edit distances brought by the greedy algorithm and enumerating algorithm are almost the same. According to discussion in Section 4.1, the enumerating algorithm is optimal, that is, the weighted edit distance brought by it is minimum, so that the results generated by the greed algorithm is very close to the optimal ones, and thus the greedy algorithm achieve the excellent ratio bound during identifying the observations.

In the third group experiments, the relative weighted edit distance between the observation sequence returned by the greedy algorithm and the original data stream was calculated while the average number of observations in a time window increased from 6 to 15, and the average length of each observations equaled to 20 and 40. The experimental results are presented in Fig. 10. It shows that the distance between the result returned by the greedy algorithm and the original data stream is quite small, that is, the observation identifying by the greedy algorithm is very close to the original data stream, which means that the greedy algorithm achieve high accuracy.

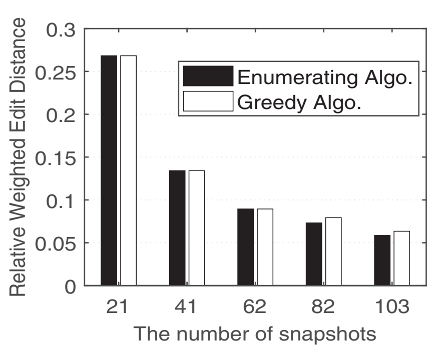
The fourth group experiments are to evaluate the energy cost of the state sequence deducing algorithm in Section 4.2. In the experiments, the energy consumed by the algorithm was computed while the number of states, k, increased from 2 to 10, and the number of observations in a time window is equal to 3 and 8. The experimental results are presented in Fig. 11. It shows that the energy consumed by Algorithm 3 in Section 4.2 is extremely small.



(a) Energy Cost of Greedy Algorithm



(b) Energy Cost of Enumerating Algorithm



(c) Ratio Bound

Fig. 9. Comparison of observation identification algorithms.

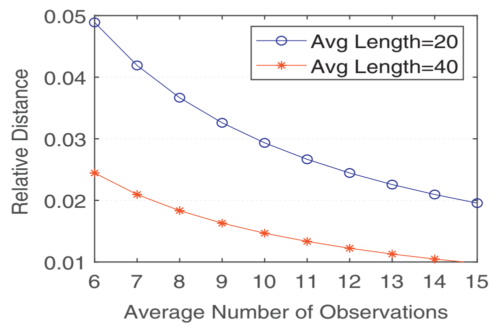


Fig. 10. Relative weighted edit distance of greedy algorithm.

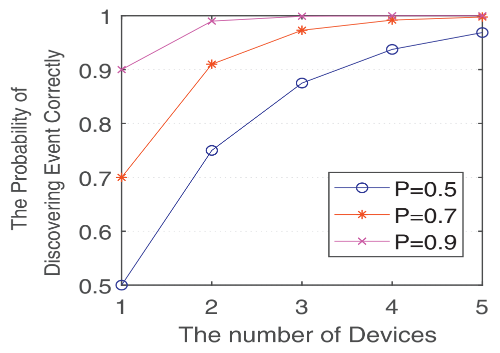


Fig. 12. The probability of detecting the event.

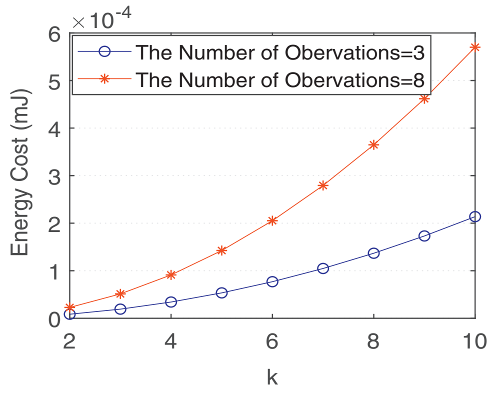


Fig. 11. Energy cost of state sequence assignment algorithm.

Algorithm 1: The Algorithm for computing $\beta_p(t)$.

```
   Input: $A_i^{(r-1)}$, $B_i^{(r-1)}$, $\overrightarrow{O_i} = (x_1, x_2, \ldots, x_{m_i})$
   Output: $\{\beta_p(t) \mid 1 \le p \le k, 1 \le t \le m_i\}$
1  $\beta_p(1) = \pi_p B_i^{(r-1)}(p, x_1)$ for all $1 \le p \le k$;for $2 \le t \le m_i$
   do
2      for $1 \le q \le k$ do
3          $\beta_q(t) = \sum_{p=1}^{k} \beta_p(t-1) A_i^{(r-1)}(p,q) B_i^{(r-1)}(q, x_t)$
4  Return $\{\beta_p(t) | 1 \le p \le k, 1 \le t \le m_i\}$;
```

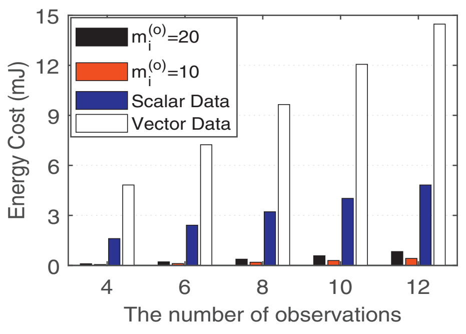
Algorithm 2: The Algorithm for computing $\gamma_q(t)$.

```
   Input: $A_i^{(r-1)}$, $B_i^{(r-1)}$, $\overrightarrow{O_i} = (x_1, x_2, \ldots, x_{m_i})$
   Output: $\{\gamma_q(t) \mid 1 \le q \le k, 1 \le t \le m_i\}$
1  $\gamma_q(m_i) = B_i^{(r-1)}(q, x_{m_i})$ for all $1 \le p \le k$;for
   $t = m_i - 1;\ t \ge 1;\ t--$ do
2      for $1 \le q \le k$ do
3          $\gamma_q(t) = \sum_{p=1}^{k} A_i^{(r-1)}(q,p) B_i^{(r-1)}(q, x_{t+1}) \gamma_p(t+1)$;
4  Return $\{\gamma_q(t) | 1 \le q \le k, 1 \le t \le m_i\}$;
```

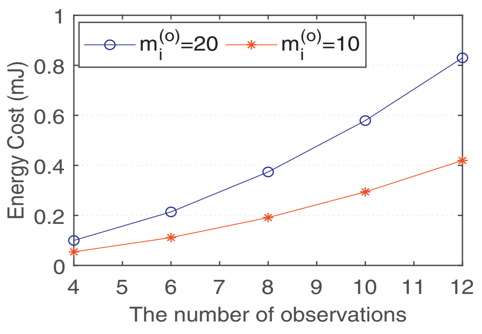
Based on Algorithm 3, the most like state sequence, *i.e.* $\overrightarrow{z_i^{(c)}}$, is determined by the dynamic programming algorithm, which is a polynomial algorithm. Meanwhile, most of input information is stored locally. Therefore, the energy consumed by transmission and computation is quite low, so that lots of energy could be saved for obtaining $\overrightarrow{z_i^{(c)}}$.

In the fifth group experiments, the energy cost of the event detection algorithm was computed while the number of observations in current time window increased from 4 to 12, and the size of the training observation set equals to 10 and 20. Fig. 13(a) and (b) show that the energy cost of our event detection algorithm is extremely small comparing with that of transmitting scalar data or vector data since only a state is required to be transmitted.

In the last group of experiments, the probability of correctly detecting the event was calculated while the number of sensor devices increased from 1 to 5. The results in Fig. 12 show that such probability is largely improved when more sensor devices are deployed in the monitored region Meanwhile, we also find that the monitoring accuracy can be significantly enhanced when multi-modal sensory data are involved in the system since they can catch different properties of the monitored objects.



(a) Comparison with Others



(b) Comparison with Self

Fig. 13. Energy cost of event detection algorithm.

Algorithm 3: The Algorithm of Deducing the Most likely State Sequence.

```
Input: $A$, $B_i^{(r-1)}$, $\overrightarrow{O_i^{(c)}} = (x_1^{(c)}, x_2^{(c)}, \ldots, x_{\tau_i}^{(c)})$
Output: $\overrightarrow{z_i^{(c)}}$
1  $\overrightarrow{z_i^{(c)}} = ()$;
2  for $1 \le p \le k$ do
3      $\mu(p,1) = \pi_p B_i(p, x_1^{(c)})$;
4  $z_{i1} = \arg\max_{p=1}^{k} \mu(p,1)$;
5  Insert $z_{i1}$ into $\overrightarrow{z_i^{(c)}}$;
6  for $2 \le t \le \tau_i$ do
7      for $1 \le q \le k$ do
8          $\mu(j,t) = \max_{p=1}^{k} \mu(p,t-1)A(p,j)B_i(j,x_t^c)$;
9      $z_{it} = \max_{j=1}^{k} \mu(j,t)$;
10     Insert $z_{it}$ into $\overrightarrow{z_i^{(c)}}$;
11 Return $\overrightarrow{z_i^{(c)}}$;
```

Algorithm 4: Redundant Observation Removing Algorithm.

```
Input: The original observation set
       $O_i = \{o_{iq} \mid 1 \le q \le m_i\}$, the original sequence
       $\overrightarrow{O_i} = (o_{i1}, o_{i2}, \ldots, o_{im_i})$, the bound $b$
Output: The final observation set and observation
        sequence.
1  for $1 \le q \le m_i$ do
2      Count the appearance time of each $o_{iq}$;
3  for $1 \le q \le m_i$ do
4      for $1 \le r \le m_i$ do
5          $ED(o_{iq}, o_{ir})$ is calculated by [26]
6          if $ED(o_{iq}, o_{ir}) \le b$ then
7              if The appearance times of $o_{iq}$ is larger then
8                  Delete all $o_{ir}$ from $O_i$;
9                  Use $o_{iq}$ to replace $o_{ir}$ in $\overrightarrow{O_i}$, andUpdate the
                   appearance times of $o_{iq}$;
10             else
11                 Delete all $o_{iq}$ from $O_i$;
12                 Use $o_{ir}$ to replace $o_{ip}$ in $\overrightarrow{O_i}$, andUpdate the
                   appearance times of $o_{ir}$;
13 Return $O_i$ and $\overrightarrow{O_i}$;
```

Algorithm 5: Greedy Algorithm for Determining Observation Sequence and Set.

```
Input: Training data set $D_i(T_s, T_f)$ and distance bound
       $b_1$
Output: Observation set $O_i$ and observation sequence
        $\overrightarrow{O_i}$
1  $O_i = \emptyset$, $l = 1$, $o_{il} = \emptyset$, $\overrightarrow{O_i} = ()$;
2  for each $d_{it_j} \in D_i(T_s, T_f)$ do
3      if $o_{il} == \emptyset$ then
4          $o_{il} = o_{il} \bigcup \{d_{it_j}\}$;
5      else
6          Insert = ture;
7          for each $d_{it_p} \in o_{il}$ do
8              if $Dis(d_{it_j}, d_{it_p}) > b_1$ then
9                  Insert = faulse,
10                 break;
11         if Insert == ture then
12             $o_{il} = o_{il} \bigcup \{d_{it_j}\}$,
13         else
14             $O_i = O_i \bigcup \{o_{il}\}$,Insert $o_{il}$ into
               $\overrightarrow{O_i} o_{i(l+1)} = \emptyset$,$l = l + 1$;
15 Call the algorithm in section 3.1.1 to remove and
   replace the redundant observations in $O_i$ and $\overrightarrow{O_i}$;
16 Return $O_i$ and $\overrightarrow{O_i}$;
```

6. Related works

Currently, there doesn't exist any work considering how to deal with multi-modal sensory data at the same time. The published literatures including the data integration and event detection techniques are the only ones which are related to our work.

Flora et al. [18] and Khalil et al. [19] proposed two semantics based sensory data integration method for WSNs. In their works, the authors used XML schema to denote sensory data, and discussed how to integrate the sensory data when the XML structures adopted by different sensors are not same. The sensory data considered by them is still described by the same language, and the modality of data that they can deal with is simple. Jirkovský et al. [20] discuss the approach of understanding data heterogeneity in Cyber-Physical Systems(CPS). The authors summarize the challenges for data integration in CPS, and proposed the shared SHS ontology and SBDH based on Semantic Web technology in order to integrate sensory data. However, similar as [18,19], the sensory data consider by it also share the same modality, and the het-

Algorithm 6: The Maximum Likelihood based Algorithm for Setting Transition and Emission Probability Matrices. The reference cited in this table is [38]

```
Input: ${\overrightarrow{O_i} \mid 1 \le i \le n}$ and ${\overrightarrow{S^{(i)}} \mid 1 \le i \le n}$
Output: The transition matrix $A$ and the emission
        matrices ${B_i \mid 1 \le i \le n}$
1  All the sensors in the network are organized as a
   spanning tree rooted at the sink according to [38];
2  for each sensor node $i$ $(1 \le i \le n)$ do
3      for each $q$ $(1 \le q \le k)$ do
4          for each $p$ $(1 \le p \le k)$ do
5              $count_1 = \sum_{t=1}^{m_i} I(S_t^{(i)} = S_q \bigwedge S_{t-1}^{(i)} = S_p)$,$count_2 =$
               $\sum_{t=1}^{m_i} I(S_{t-1}^{(i)} = S_p)$,$A_i(p,q) = \frac{cout_1}{count_2}$;
6          $count_3 = \sum_{t=1}^{m_i} I(S_t^{(i)} = S_q)$;
7          for each $v$ $(1 \le v \le m_i^{(o)})$ do
8              $cout_4 = \sum_{t=1}^{m_i} I(S_t^{(i)} = S_q \bigwedge x_t = o_{iv})$,$B_i(q,v) =$
               $\frac{count_4}{count_3}$;
9      Return $B_i$ by each sensor node;
10     Transmit $A_i$ towards the sink;
11 ${A_i \mid 1 \le i \le n}$ are transmitted and aggregated along
   the spanning tree;
12 The sink obtains $\sum_{i=1}^n A_i$,$A = \frac{1}{n}\sum_{i=1}^n A_i$;
13 Return $A$;
```

erogeneity of data only reflects in their representation. Thus, the problem of how to support the fusing computation on multi-modal sensory data are not considered either. Besides, the approach proposed by it is centralized, not suitable and efficient for IoT systems.

Meanwhile, the data integration methods in other areas cannot support the fusing computation on multiple sensory data as well. Dai et al. [30] studied a data integration problem for health-care data, the authors proposed a Neural Concept Liking approach for accurate concept linking, and give a data integration method accordingly. Similar as the above works, all the data considered by Dai et al. [30] is text snippets, and they cannot applyto multi-modal sensory data. A Bayesian framework for integrating the heterogeneous gene data, combining the evidences and determining a posterior probability of whether each pairs of genes had a functional relationship is studied by Troyanskaya et al. [31]. The authors introduced a system, named as MAGIC, to achieve such aim. However, the input data of MAGIC are real matrices and just has different formats, so that the multi-modal data had not been investigated by Troyanskaya et al. [31], either. Furthermore, all these algorithms are centralized, and not suitable for IoT systems.

For event detection, [32,33] propose several algorithms based on threshold and interest diffusion. These algorithms can only identify few events, and there exist lots of redundant reports since they do not consider the spatial and temporal correlations between sensory data. Furthermore, they only consider how to deal with scalar data, and are not applicable to complicated multi-modal sensory data. The works in [34–37] proposed pattern and statistical model based event detection algorithms. These algorithms save lots of energy since the correlation among sensory data is sufficiently considered. However, these algorithms also only can deal with a single modal sensory data and cannot process multi-modal sensory data.

7. Conclusion

This paper takes the event detection as an example to study the fusing computation algorithm on multi-modal sensory data. Firstly, a novel model to integrate multi-modal sensory data is proposed based on the Hidden Markov Model. Two model learning algorithms are given according to the maximum likelihood and EM estimation. Finally, the event detection algorithm is provided based on the learnt model and current collected sensory data. The theoretical analysis and extensive experiment results indicate that all the proposed algorithms have high performance in terms of accuracy and energy consumption.

Acknowledgments

This work is partly supported by the National Natural Science Foundation of China under Grant Nos. 61632010, U1509216 and National Natural Science Foundation of USA under Grant Nos. 1741277, 1851197, 1741287, 1741279, 1741338.

Appendix A

A1. Algorithms in Section 3.1 and the Proof of Theorem 1

This section will present the *Redundant Observation Removing Algorithm* discussed in Section 3.1.1 and the *Greedy Algorithm for Determining Observation Sequence and Set* involved in Section 3.1.2, respectively. Meanwhile, we will also provide the proof of Theorem 1, which is mentioned in Section 3.2.1.

Theorem 1. *A_i and B_i are the solution of the problem given in Formula (3) if $A_i(p,q) = \frac{\sum_{t=1}^{m_i} I(S_t^{(i)}=S_q \bigwedge S_{t-1}^{(i)}=S_p)}{\sum_{t=1}^{m_i} I(S_{t-1}^{(i)}=S_p)}$ and $B_i(q,v) = \frac{\sum_{t=1}^{m_i} I(S_t^{(i)}=S_q \bigwedge x_t=o_{iv})}{\sum_{t=1}^{m_i} I(S_t^{(i)}=S_q)}$ for all $1 \le q,\ p \le k$ and $1 \le v \le m_i^{(o)}$, where $I(X)$ is an indicate function, i.e. $I(X) = 1$ if random event X is true, otherwise $I(X) = 0$.*

Proof of Theorem 1. According to Formula (3), we have

$$\begin{aligned}
A_i, B_i &= arg\max_{A,B} \Pr(\overrightarrow{S^{(i)}}, \overrightarrow{O_i}|A,B) = arg\max_{A,B}(\log \Pr(\overrightarrow{S^{(i)}}, \overrightarrow{O_i}|A,B)) \\
&= arg\max_{A,B} \log\{\prod_{t=1}^{m_i} \Pr(x_t|S_t^{(i)}, B) \prod_{t=1}^{m_i} \Pr(S_t^{(i)}|S_{t-1}^{(i)}, A)\} \\
&= arg\max_{A,B} \sum_{t=1}^{m_i} (\log B(S_t^{(i)}, x_t) + \log A(S_{t-1}^{(i)}, S_t^{(i)})) \\
&= arg\max_{A,B} \sum_{p=1}^{k}\sum_{q=1}^{k}\sum_{v=1}^{|m_i^{(o)}|}\sum_{t=1}^{m_i} \{I(S_t^{(i)} = S_q \bigwedge x_t = o_{iv}) \times \log B(q,v) \\
&\quad + I(S_t^{(i)} = S_q \bigwedge S_{t-1}^{(i)} = S_p) \log A(p,q)\} \qquad (10)
\end{aligned}$$

Let $\mathcal{L}(A,B,\epsilon,\delta)$ satisfy

$$\begin{aligned}
\mathcal{L}(A,B,\epsilon,\delta) &= \sum_{p=1}^{k}\sum_{q=1}^{k}\sum_{v=1}^{|m_i^{(o)}|}\sum_{t=1}^{m_i} \{I(S_t^{(i)} = S_q \bigwedge x_t = o_{iv}) \log B(q,v) \\
&\quad + I(S_t^{(i)} = S_q \bigwedge S_{t-1}^{(i)} = S_p) \log A(p,q)\} \\
&\quad + \sum_{q=1}^{k} \epsilon_q (1 - \sum_{v=1}^{m_i^{(o)}} B(q,v)) + \sum_{p=1}^{k} \delta_p (1 - \sum_{q=1}^{k} A(p,q)) \qquad (11)
\end{aligned}$$

where $\delta_p = \sum_{t=1}^{m_i} I(S_{t-1}^{(i)} = S_p)$ and $\epsilon_q = \sum_{t=1}^{m_i} I(S_t^{(i)} = S_q)$.

Since $\sum_{v=1}^{m_i^{(o)}} B(q,v)=1$ and $\sum_{q=1}^{k} A(p,q)=1$, Formula (11) can be reduced to

$$A_i, B_i = arg\max_{A,B} \mathcal{L}(A,B,\epsilon,\delta) \tag{12}$$

According to the condition of Theorem 1, $A_i(p,q) = \frac{\sum_{t=1}^{m_i} I(S_t^{(i)}=S_q \wedge S_{t-1}^{(i)}=S_p)}{\sum_{t=1}^{m_i} I(S_{t-1}^{(i)}=S_p}$) for any $1\le p,\ q\le k$. Thus, $\frac{\partial \mathcal{L}(A,B,\epsilon,\delta)}{\partial A(p,q)}\Big|_{A=A_i} = 0.$ Similarly, $\frac{\partial \mathcal{L}(A,B,\epsilon,\delta)}{\partial B(q,v)}\Big|_{B=B_i} = 0$ Therefore, A_i and B_i are the solution of the problem given in Formula (3). □

A2. The algorithm in Section 3.2.1 and the Proof of Theorem 2

The following section will provide the pseudocode of the Algorithm discussed in Section 3.2.1, and the proof of Theorem 2 shown in Section 3.2.2.

Theorem 2. *$A_i^{(r)}$ and $B_i^{(r)}$ are the solutions of the above problem if $A_i^{(r)}(p,q) = \frac{\sum_{t=1}^{m_i}\eta_t(p,q)}{\sum_{p=1}^{k}\sum_{t=1}^{m_i}\eta_t(p,q)}$ and $B_i^{(r)}(q,v) = \frac{\sum_{p=1}^{k}\sum_{t=1}^{m_i} I(x_t=o_{iv})\eta_t(p,q)}{\sum_{p=1}^{k}\sum_{t=1}^{m_i}\eta_t(p,q)}$ for any $1\le p,q\le k$ and $1\le v\le |O_i| = m_i^{(o)}$, where*

$\eta_t(p,q) = \beta_p(t-1)A_i^{(r-1)}(p,q)B_i^{(r-1)}(q,x_t)\gamma_q(t)$, $\beta_p(t) = \Pr(x_1, x_2,\ldots,x_t, z_{it}=S_p|A_i^{(r-1)}, B_i^{(r-1)})$ and $\gamma_q(t) = \Pr(x_{t+1},\ldots,x_{m_i-1}, x_{m_i}, z_{it}=S_q|A_i^{(r-1)}, B_i^{(r-1)})$.

Proof of Theorem 2. Let $\mathcal{L}(A_i^{(r)}, B_i^{(r)}, \epsilon, \delta)$ satisfy

$$\mathcal{L}(A,B,\epsilon,\delta) = \sum_{\vec{z_i}\in S^{m_p}} \varphi(\vec{z_i}) \sum_{p=1}^{k}\sum_{q=1}^{k}\sum_{v=1}^{|m_i^{(o)}|}\sum_{t=1}^{m_i}\{I(z_{it}=S_q \bigwedge x_t = o_{iv})$$
$$\log B(q,v) + I(z_{it}=S_q\bigwedge z_{it-1}=S_p)\log A(p,q)\}$$
$$+\sum_{q=1}^{k}\epsilon_q\left(1-\sum_{v=1}^{m_i^{(o)}}B(q,v)\right)+\sum_{p=1}^{k}\delta_p\left(1-\sum_{q=1}^{k}A(p,q)\right) \tag{13}$$

where δ_p and ϵ_q satisfies that
$\delta_p = \sum_{\vec{z_i}}\varphi(\vec{z_i})\sum_{t=1}^{m_i} I(z_{it-1}=S_p)$ and $\epsilon_q = \sum_{\vec{z_i}}\varphi(\vec{z_i})$ $\sum_{t=1}^{m_i} I(z_{it}=S_q)$.

Using the similar proof with Theorem 1, we have that $A_i^{(r)}$ and $B_i^{(r)}$ are the solution of the problem given in Formula (4) if $A_i^{(r)}, B_i^{(r)} = arg\max_{A,B}\mathcal{L}(A,B,\epsilon,\delta)$.. Thus,

$$\frac{\partial\mathcal{L}(A,B,\epsilon,\delta)}{\partial A(p,q)} = \sum_{\vec{z_i}\in S^{m_p}}\varphi(\vec{z_i})\frac{1}{A(p,q)}\sum_{t=1}^{m_i} I(z_{it}=S_q\bigwedge z_{it-1}=S_p)-\delta_p \tag{14}$$

Based on the condition in Theorem 2,

$$A_i^{(r)}(p,q) = \frac{\sum_{t=1}^{m_i}\eta_t(p,q)}{\sum_{p=1}^{k}\sum_{t=1}^{m_i}\eta_t(p,q)} = \frac{\frac{1}{\Pr(\vec{O_i}|A_i^{(r-1)},B_i^{(r-1)})}\sum_{t=1}^{m_i}\eta_t(p,q)}{\frac{1}{\Pr(\vec{O_i}|A_i^{(r-1)},B_i^{(r-1)})}\sum_{p=1}^{k}\sum_{t=1}^{m_i}\eta_t(p,q)} \tag{15}$$

where the numerator of Formula (15) satisfies that

$$\frac{\sum_{t=1}^{m_i}\eta_t(p,q)}{\Pr(\vec{O_i}|A_i^{(r-1)},B_i^{(r-1)})} = \frac{\sum_{t=1}^{m_i}\beta_p(t-1)A_i^{(r-1)}(p,q)B_i^{(r-1)}(q,x_t)\gamma_q(t)}{\Pr(\vec{O_i}|A_i^{(r-1)},B_i^{(r-1)})}$$
$$= \frac{1}{\Pr(\vec{O_i}|A_i^{(r-1)},B_i^{(r-1)})}\sum_{t=1}^{m_i}\sum_{\vec{z}} I(z_{it-1}=S_p\bigwedge z_{it}=S_q)$$
$$\Pr(\vec{z_i},\vec{O_i}|A_i^{(r-1)},B_i^{(r-1)})$$
$$= \sum_{t=1}^{m_i}\sum_{\vec{z_i}} I(z_{it-1}=S_p\bigwedge z_{it}=S_q)\Pr(\vec{z_i}|\vec{O_i},A_i^{(r-1)},B_i^{(r-1)}) \tag{16}$$

since $\eta_t(p,q) = \beta_p(t)A_i^{(r-1)}(p,q)B_i^{(r-1)}(q,x_t)\gamma_q(t+1)$, $\beta_p(t) = \Pr(x_1,x_2,\ldots,x_t,z_{it}=S_p|A_i^{(r-1)},B_i^{(r-1)})$ and $\gamma_q(t) = \Pr(x_{t+1},\ldots,x_{m_i-1}, x_{m_i}, z_{it}=S_q|A_i^{(r-1)},B_i^{(r-1)})$. According to the definition of $\varphi(\vec{z_i})$ ($=\Pr(\vec{z_i}|\vec{O_i},A^{(r-1)},B_i^{(r-1)})$), we have

$$\frac{1}{\Pr(\vec{O_i}|A_i^{(r-1)},B_i^{(r-1)})}\sum_{t=1}^{m_i}\eta_t(p,q)$$
$$= \sum_{\vec{z_i}}\varphi(\vec{z_i})\sum_{t=1}^{m_i} I(z_{it-1}=S_p\bigwedge z_{it}=S_q) \tag{17}$$

By the same way, the denominator of Formula (15) satisfies

$$\frac{1}{\Pr(\vec{O_i}|A_i^{(r-1)},B^{(r-1)})}\sum_{p=1}^{k}\sum_{t=1}^{m_i}\eta_t(p,q) = \sum_{\vec{z_i}}\varphi(\vec{z_i})\sum_{t=1}^{m_i} I(z_{it-1}=S_p) \tag{18}$$

From Formulas (15), (17) and (18), we have $A_i^{(r)}(p,q) = \frac{\sum_{\vec{z_i}}\varphi(\vec{z_i})\sum_{t=1}^{m_i} I(z_{it-1}=S_p\bigwedge z_{it}=S_q)}{\sum_{\vec{z_i}}\varphi(\vec{z_i})\sum_{t=1}^{m_i} I(z_{it-1}=S_p)}$. Therefore, $\frac{\partial\mathcal{L}(A,B,\epsilon,\delta)}{\partial A(p,q)}\Big|_{A=A_i^{(r)}} = \sum_{\vec{z_i}}\varphi(\vec{z_i})\sum_{t=1}^{m_i} I(z_{it-1}=S_p)-\delta_p$. Since $\delta_p = \sum_{\vec{z_i}}\varphi(\vec{z_i})\sum_{t=1}^{m_i} I(z_{it-1}=S_p)$, $\frac{\partial\mathcal{L}(A,B,\epsilon,\delta)}{\partial A(p,q)}\Big|_{A=A_i^{(r)}} = 0.$

Similarly, $\frac{\partial\mathcal{L}(A,B,\epsilon,\delta)}{\partial B(q,v)}\Big|_{B=B_i^{(r)}} = 0$. Therefore, $A_i^{(r)}$ and $\widehat{B_i^{(r)}}$ are the solution of the problem shown in Formula (4). □

References

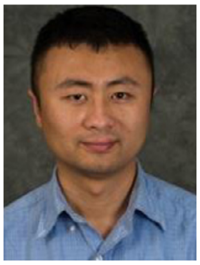
[1] Z. Li, T. He, Webee: Physical-layer cross-technology communication via emulation, in: MobiCom 2017, 2017, pp. 2–14.
[2] W. Jiang, Z. Yin, R. Liu, Z. Li, S.M. Kim, T. He, Bluebee: a 10,000× faster cross-technology communication via PHY emulation, in: SenSys, 2017, pp. 3:1–3:13.
[3] S. Wang, S.M. Kim, T. He, Symbol-level cross-technology communication via payload encoding, in: ICDCS, 2018, pp. 500–510.
[4] N.D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, A.T. Campbell, A survey of mobile phone sensing, Commun. Mag. IEEE 48 (9) (2010) 140–150.
[5] S. Cheng, Z. Cai, J. Li, H. Gao, Extracting kernel dataset from big sensory data in wireless sensor networks, IEEE Trans. Knowl. Data Eng. 29 (4) (2017) 813–827.
[6] S. Cheng, Z. Cai, J. Li, X. Fang, Drawing dominant dataset from big sensory data in wireless sensor networks, in: INFOCOM, 2015, pp. 531–539.
[7] M. Gupta, L.V. Shum, E.L. Bodanese, S. Hailes, Design and evaluation of an adaptive sampling strategy for a wireless air pollution sensor network, in: LCN, 2011, pp. 1003–1010.
[8] D. Chu, A. Deshpande, J.M. Hellerstein, W. Hong, Approximate data collection in sensor networks using probabilistic models, in: ICDE, 2006, pp. 48–59.
[9] M. Li, Y. Liu, L. Chen, Non-threshold based event detection for 3d environment monitoring in sensor networks, in: ICDCS, 2007, p. 9.
[10] F. Xiao, Q. Miao, X. Xie, L. Sun, R. Wang, SHMO: A seniors health monitoring system based on energy-free sensing, Comput. Netw. 132 (2018) 108–117.
[11] X. Zheng, Z. Cai, J. Li, H. Gao, A study on application-aware scheduling in wireless networks, IEEE Trans. Mob. Comput. 16 (7) (2017) 1787–1801.
[12] X. Zheng, Z. Cai, J. Li, H. Gao, An application-aware scheduling policy for real–time traffic, in: ICDCS, 2015, pp. 421–430.
[13] J. Considine, F. Li, G. Kollios, J. Byers, Approximate aggregation techniques for sensor databases, in: ICDE, IEEE Computer Society, Boston, MA, USA, 2004, pp. 449–460.
[14] A. Silberstein, R. Braynard, C.S. Ellis, K. Munagala, J. Yang, A sampling-based approach to optimizing top-k queries in sensor networks, in: ICDE, 2006, pp. 68–78.
[15] M. Macit, V.C. Gungor, G. Tuna, Comparison of qos-aware single-path vs. multi-path routing protocols for image transmission in wireless multimedia sensor networks, Ad Hoc Netw. 19 (2014) 132–141.
[16] M.Y. Donmez, S. Isik, C. Ersoy, Analysis of a prioritized contention model for multimedia wireless sensor networks, TOSN 10 (2) (2014) 36.
[17] T. Wang, J. Zeng, M.Z.A. Bhuiyan, Y. Chen, Y. Cai, H. Tian, M. Xie, Energy-efficient relay tracking with multiple mobile camera sensors, Comput. Netw. 133 (2018) 130–140.

[18] A. Flora, C. Valentina, G. Andrea, M. Antonino, A semantic enriched data model for sensor network interoperability, Simul. Modell. Pract. Theory 19 (8) (2011) 1745–1757.
[19] I.I. Khalil, K. Reinhard, K. Gabriele, A semantic solution for data integration in mixed sensor networks, Comput. Commun. 28 (13) (2005) 1564–1574.
[20] V. Jirkovský, M. Obitko, V. Maŕık, Understanding data heterogeneity in the context of cyber-physical systems integration, IEEE Trans. Ind. Inf. 13 (2) (2017) 660–667.
[21] E.R. J, A. Lakhdar, M.J. B, Hidden Markov Models, Springer, 1994.
[22] R. Ben-El-Kezadri, G. Pau, T. Claveirole, Turbosync: Clock synchronization for shared media networks via principal component analysis with missing data, in: INFOCOM, 2011, pp. 1170–1178.
[23] S.B. Qaisar, S. Imtiaz, F. Faruq, A. Jamal, W. Iqbal, P. Glazier, S. Lee, A hidden markov model for detection & classification of arm action in cricket using wearable sensors, J. Mob. Multimedia 9 (1&2) (2013) 128–144.
[24] W. An, C. Park, X. Han, K.R. Pattipati, D.L. Kleinman, W.G. Kemple, Hidden markov model and auction-based formulations of sensor coordination mechanisms in dynamic task environments, IEEE Trans. Syst. Man Cybern. Part A 41 (6) (2011) 1092–1106.
[25] B. Gunilla, Distance transformations in digital images, Comput. Vis. Graphics Image Process. 34 (3) (1986) 344–371.
[26] S. David, Matching sequences under deletion/insertion constraints, Proc. Natl. Acad. Sci. USA 69 (1) (1972) 4–6.
[27] E.S. R, Maximum likelihood estimation: Logic and practice, Sage, 1993.
[28] http://www.parallax.com/product/boe-bot-robot
[29] http://cseweb.ucsd.edu/groups/csag/html/teaching/cse291s03/OtherCourse Material/Crossbow/MPRMIBSeriesUserManual.pdf
[30] J. Dai, M. Zhang, G. Chen, J. Fan, K.Y. Ngiam, B.C. Ooi, Fine-grained concept linking using neural networks in healthcare, in: Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10–15, 2018, 2018, pp. 51–66.
[31] O.G. Troyanskaya, K. Dolinski, A.B. Owen, R.B. Altman, D. Botstein, A bayesian framework for combining heterogeneous data sources for gene function prediction (in saccharomyces cerevisiae), Proc. Natl. Acad. Sci. 100 (14) (2003) 8348–8353.
[32] P. Themistoklis, P. Dimitris, K. Vana, G. Dimitrios, Distributed deviation detection in sensor networks, ACM SIGMOD Rec. 32 (4) (2003) 77–82.
[33] I. Chalermek, G. Ramesh, E. Deborah, Directed diffusion: a scalable and robust communication paradigm for sensor networks, in: Mobicom, 2000, pp. 56–67.
[34] W. Li, L. Jianxin, Z. Xiaomin, A frequent pattern based framework for event detection in sensor network stream data, in: Proceedings of the Third International Workshop on Knowledge Discovery from Sensor Data, 2009, pp. 87–96.
[35] X. Wenwei, L. Qiong, W. Hejun, Pattern-based event detection in sensor networks, Distrib. Parallel Databases 30 (1) (2012) 27–62.
[36] M. Ding, X. Cheng, Robust event boundary detection in sensor networks-a mixture model based approach, in: INFOCOM 2009, IEEE, 2009, pp. 2991–2995.
[37] J. Gupchup, A. Terzis, R. Burns, et al. Model-based event detection in wireless sensor networks[J]. arXiv preprint arXiv:0901.3923, 2009.
[38] G. Huang, X. Li, J. He, Dynamic minimal spanning tree routing protocol for large wireless sensor networks, in: Proceedings of the 1st IEEE Conference on Industrial Electronics and Applications, 2006, pp. 1–5.

Siyao Cheng is an Associate Professor in the School of Computer Science and Technology at Harbin Institute of Technology. She received the BS, Master and PhD degrees in computer science from Harbin Institute of Technology. She worked as a postdoctoral scholar at Georgia State University from Sept. 2013 to Sept. 2014. Her research interests include big sensory data management, wireless sensor networks, cognitive radio networks.

Yingshu Li received her Ph.D. and M.S. degrees from the Department of Computer Science and Engineering at University of Minnesota-Twin Cities. She received her B.S. degree from the Department of Computer Science and Engineering at Beijing Institute of Technology, China. Dr. Li is currently an Associate Professor in the Department of Computer Science at Georgia State University. Her research interests include Wireless Networking, Sensor Networks, Sensory Data Management, Social Networks, and Optimization. Dr. Li is the recipient of an NSF CAREER Award. She is a senior member of the IEEE.

Zhi Tian received the B.E. degree in electrical engineering (Automatic Control) from the University of Science and Technology of China (USTC), Hefei, China, in 1994, and the M.S. and Ph.D. degrees from George Mason University, Fairfax, VA, in 1998 and 2000 respectively. From 1994 to 1995, she studied in the graduate program of the department of Automation at Tsinghua University, Beijing, China. From 1995 to 2000, she was a graduate research assistant in the Center of Excellence in Command, Control, Communications and Intelligence (C3I) of George Mason University. From August 2000 to December 2014, she was on the faculty of the Electrical and Computer Engineering (ECE) department of Michigan Technological University, where she was promoted to Full Professor in 2011. From November 2011 to December 2014, Dr. Tian served as a Program Director in the Division of Electrical, Communications and Cyber Systems (ECCS) at the National Science Foundation (NSF), through an IPA assignment with Michigan Tech. In January 2015, she joined Mason as a Professor in the ECE department.



Wei Cheng received his Ph.D. degree in computer science from the George Washington University in 2010, and B.S. degree in applied mathematics, M.S. degree in computer science both from National University of Defense Technology, China, in 2002 and 2004. Currently, he is an Assistant Professor at University of Washington Tacoma. He has worked as a Postdoc Scholar at University of California Davis. His research interests span the areas of Security and Cyber-Physical Systems. In particular, he is interested in HCI based Security, Location Privacy Protection, Smart Cities, and Underwater Networks.

Xiuzhen Cheng received her M.S. and Ph.D. degrees in computer science from the University of Minnesota – Twin Cities, in 2000 and 2002, respectively. She is a professor at the Department of Computer Science, The George Washington University, Washington DC. Her current research interests focus on privacy-aware computing, wireless and mobile security, mobile handset networking systems (mobile health and safety), cognitive radio networks, and algorithm design and analysis. She has served on the editorial boards of several technical journals (e.g. IEEE Transactions on Parallel and Distributed Systems, IEEE Wireless Communications) and the technical program committees of various professional conferences/workshops (e.g. IEEE INFOCOM, IEEE ICDCS, ACM Mobihoc, IEEE/ACM IWQoS). She also has chaired several international conferences (e.g. IEEE CNS, WASA). She worked as a program director for the US National Science Foundation (NSF) from April to October in 2006 (full time), and from April 2008 to May 2010 (part time). She is a Fellow of IEEE.