Wireless Netw (2014) 20:2409–2418
DOI 10.1007/s11276-014-0757-3

Trajectory-based multi-dimensional outlier detection in wireless sensor networks using Hidden Markov Models

Chen Wang · Hongzhi Lin · Hongbo Jiang

Published online: 14 June 2014
© Springer Science+Business Media New York 2014

Abstract Wireless sensor networks (WSNs) have been increasingly available for monitoring the traffic, weather, pollution, etc. Outlier detection in WSNs is an essential step for many important applications, such as abnormal event detection, fraud analysis, etc. While existing efforts focus on identifying individual outliers from sensory data, the unsupervised high semantic outlier detection in WSNs is more challenging and has received far less attentions. In addition, the correlation between multi-dimensional sensory data has not yet been considered when detecting outliers in WSNs. In this paper, based on multi-dimensional Hidden Markov Models, we propose a trajectory-based outlier detection algorithm by model training and model-based likelihood estimation. Our data preprocessing, clustering, model training and model updating schemes are developed to reduce the computational complexity and enhance the detecting performance. We also explore the possibility and feasibility of adapting the proposed algorithm to real-time outlier detections. Experimental results show that our methods achieve good performance on detecting various kinds of abnormal trajectories composed of multi-dimensional data.

Keywords Outlier detection · Sensor network · Multi-dimensional data · Hidden Markov Model

C. Wang · H. Lin (✉) · H. Jiang
Department of Electronics and Information Engineering, Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology, Wuhan 430074, China
e-mail: eihongzhilin2012@gmail.com

1 Introduction

Wireless sensor networks (WSNs) today are widely used as they are able to capture the phenomena of the physical world that were originally difficult or impossible to obtain by traditional techniques [1–3]. Among numerous applications in WSNs, outlier detection leads to a very important data mining problem, which is preliminary to many data analysis such as abnormal event detection, animal behavior change prediction, fraud analysis, and intrusion detection [4].

Some remarkable work has been done [5–11] in WSNs recently. Despite the significant progress, most of existing efforts concern distance-based or density-based outliers, with their focuses on temporal or spatial neighboring short-term data segments. These techniques can achieve favourable performance on detecting outliers whose values or distributions greatly differ from adjacent data segments or neighboring nodes, but would suffer notable performance degradation as to some high semantic outliers.

Consider the following example: a farmer may be interested in the temperature trajectory of a one-day period. Assume the normal temperature trajectory of a day is 25, 28, 25, 22, 25 °C. A value of 28 °C is reasonable when it appears at noon, but it is more likely to be an outlier when it appears in the midnight. An abnormal trajectory with the same value distribution as the normal one but different order of appearance, e.g., 25, 22, 25, 28, 25 °C, may have a terrible effect on crops. Similarly, in a health monitoring application for patients, a blood pressure value of 130 mmHg is normal in the morning, but may be a dangerous sign at night.

Rather than characterised by extreme values or short-term abnormal data, high semantic outliers always manifest as gradual and long-term deviations from the patterns

representing the normal behavior of sensory data (such as the gradual abnormal increasing/decreasing pattern and the order reversal, etc.). These outliers probably imply interesting high level semantic events that are of great interests to researchers. On the contrary, a single abnormal or impulse-like outlier may not be that important, as it is probably caused by frequent sensory errors or environmental noises. Existing algorithms based on short-term data analysis can barely capture the gradual or long-term trajectory changes. So we motivate our technique in the context of the above challenging problem, focusing our work on detecting such high semantic outliers rather than single value or impulse-like outliers.

Instead of detecting outliers by comparing temporal or spatial neighboring data [5, 6, 12, 13], we tackle the aforementioned problem from a new perspective: we observed that in many sensor network applications mentioned above, there always exist one or more periodical patterns representing the normal behavior of sensory data. So a periodical data stream as a whole can be considered as a trajectory pattern. As a consequence, almost all kinds of outliers can be presented as deviations from the normal trajectories. So we explore the normal behaviors of periodical trajectories through a machine learning procedure, and using information learned from the normal trajectories. Then the deviation between the underlying data trajectory and the normal trajectories can be estimated by calculating the probability of observing the underlying trajectory. In our implementation, Hidden Markov Models (HMMs) are used to extract the temporal pattern of the whole normal trajectories, rather than merely applied to model segments of a trajectory for the usage of indexing or querying. Instead of typically using HMMs with only a few discrete symbols [14, 15], our proposed algorithm utilizes HMMs with continuous observation densities, and thus can precisely extract the features of a trajectory. This procedure is followed by a clustering and in-cluster model aggregation method to utilize the spacial correlation of the neighboring nodes.

For applications where high level semantic outliers should be detected, analysis based on single dimensional data [14, 15] becomes ineffective, as the sensory data collected is often multi-dimensional. To illustrate by a concrete example, people would be able to monitor two normal trajectories of a data stream, one with an increasing temperature and decreasing air pressure, the other vice verse (as temperature and air pressure are always significantly reversely correlated). Now consider a special trajectory with an increasing temperature and an increasing air pressure, analysis based on any of the two attributes (temperature and air pressure) can hardly detect the outlier. Such a case implies that, to detect high level semantic outliers, the correlation between multiple dimensions should be taken into consideration. To capture the correlation among multiple dimensions, we apply a multi-dimensional HMM implementation, to approximate a great range of periodical trajectories of multi-dimensional data in WSN applications.

On the downside, directly applying the typical multi-dimensional HMM implementations as is done in other communities is infeasible. Sensors have severely limited memory, energy and processing ability, yielding most high semantic outlier detection algorithms and multi-dimensional data analysis methods infeasible. To address this problem, in this paper we develop a data preprocessing procedure, which divides data into segments and extracts the feature of each segments by the Fourier transform. Our analysis and experiments show that such procedure can greatly reduce the computation cost and storage occupation, which makes our outlier detection method feasible for sensor networks.

Besides, most applications in WSNs demand real-time outlier detection, while few HMM based algorithms have taken time-constrained requirements into consideration. So in addition to the off-line statistic method, we also develop a segmental and real-time detection method that can be combined with the statistic method to meet different level of time-constrained requirements. Experimental results show that our detection algorithm is able to tolerate slight changes in trajectories, distinguish obvious abnormal trajectories when enough information of such trajectories are obtained, and detect significant trajectory changes immediately.

We summarize our main contributions as follows:

- Based on trajectory analysis instead of individual sensor data value analysis, we develop an outlier detection algorithm by HMM training and model-based likelihood estimation. To the best of our knowledge, this is *the first trajectory-based outlier detection scheme for WSNs*.
- Our algorithm is able to deal with multi-dimensional sensor data, as our multi-dimensional HMM implementation can efficiently handle the correlation among multi-dimensional data.
- We propose a preprocessing procedure, greatly reducing the size of the data while capturing the features of the trajectory.
- To capture trajectory changes immediately, we propose a trajectory-based real-time outlier detection method, which can be used along with our static method to fit different level of instantaneity requirements.

The reminder of the paper is organized as follows. In Sect. 2, we introduce some basic knowledge of the HMM and present our basic idea on how to apply the HMM to the outlier detection in WSNs. In Sect. 3, we develop a series

of algorithms to make trajectory-based outlier detection feasible in WSNs. Section 4 presents the evaluation results, and Sect. 5 reviews some related work. Finally, we conclude this paper in Sect. 6.

2 Preliminaries

Before introducing the proposed approach, we provide in this section some necessary background and the basic idea of the trajectory-based outlier detection in WSNs.

2.1 HMM representation of sensor data

For better understanding the distributed phenomena, HMMs have been shown the capability of well transforming numerical sensor data into symbols [14, 15], in order to model high level semantic events from low level sensor signals. HMMs have been successfully applied to many communities, such as speech recognition [16, 17] and vision tasks [21], as a powerful tool to represent the context information of particular types of stochastic processes.

An HMM is a statistical Markov model where the modeled system is assumed to be a Markov process with hidden (unobserved) states. In an HMM, there are N states, denoted as $S = S_1, S_2, \ldots, S_N$, which are not directly visible, while M observation symbols $V = v_1, v_2, \ldots, v_M$, dependent on the state, are visible. The observation sequence $O = O_1 O_2 \ldots O_T (O_t \in S)$, provides some information about the sequence of states $Q = q_1 q_2 \ldots q_T (q_t \in V)$. An HMM can be considered as a double stochastic process consisting of a Markov chain, representing the state transition probability, and a stochastic process, describing the relationship between the observation symbols and hidden states. Similar to the regular Markov model, the distribution of the state transition probability is $A = a_{ij}$, where $a_{ij} = P(q_{t+1} = S_j \mid q_t = S_i)$, and the initial state distribution is $\pi = \pi_i$, where $\pi_i = P(q_1 = S_i)$. Each state has a probability distribution over the possible observation symbol $B = b_j(k)$, where $b_j(k) = P(v_k \, at \, t \mid q_t = S_j)$. An HMM is completely characterized by the above parameters, denoted as $\theta = (A, B, \pi)$.

2.2 Trajectory-based outlier detection in WSNs

Different from previous outlier detection schemes in WSNs [5, 6], our algorithm considers the outliers as unusual events [4], with a set of observations obtained by sensor nodes as a trajectory.

2.2.1 Normal trajectory-based data modeling

We detect outliers by first modeling normal trajectory-based data. To learn the HMM for a usual event, a set of parameters θ^* is calculated by maximizing the likelihood of the observation sequence (trajectory) O_r as $\theta_r^* = \texttt{argmax}_\theta P(O_r \mid \theta_r)$. The probability density function of each HMM state is assumed to be a Gaussian Model or a Gaussian Mixture Model (GMM), so it can be solved by the standard Expectation-Maximization (EM) algorithm [17].

2.2.2 Abnormal trajectory detection

After representing the training data acquired from the history using an HMM model, the detection of outlier can be performed by efficiently compute the probability of observing the newly sensed trajectory $p(o_1 o_2 \cdots o_k)$ under the trained HMM model. Denote the likelihood of observing O_l by $L(O_l \mid \theta_r^*)$. In our approach, if the maximum likelihood is less than a threshold Th_A, i.e., $L(O_l \mid \theta_r^*) < Th_A$, then this trajectory O_l is identified to be an outlier. We will detail this process in the following section.

3 Algorithm design

In this section, we describe our trajectory-based outlier detection algorithm in detail. The proposed algorithm includes mainly three phases:

Setup phase First, history data of several periods is collected by each node as the training set. Then clustering algorithm described in Sect. 3.3 is used to cluster nodes exhibiting similar characteristics and to select cluster head. After that, multi-dimensional HMM implementation presented in Sect. 3.2 is adopted in each cluster head to train models of several normal trajectories, which will be delivered to each in-cluster node and be kept there for the use of outlier detection.

Outlier detection phase First, data preprocessing algorithm described in Sect. 3.1 is applied in each node as soon as a segment of the trajectory data has been received, and the preprocessed data is stored and used to detect outliers. Then outlier detection methods presented in Sect. 3.5 are adopted according to different level of instantaneity requirements.

Model updating phase Preprocessed data stored in each node is uploaded to the cluster head every several periods, and then the model updating algorithm presented in Sect. 3.4 is adopted there.

We now present each phase of our algorithms below in details. After that, we briefly analysis the feasibility of applying these methods to sensor networks in Sect. 3.6.

3.1 Data preprocessing

In WSNs, sensors have very limited memory, energy and processing ability, yielding it infeasible to train our HMM model directly by using the enormous sensed data. So a data preprocessing technique is necessary to preform the data aggregation (compression) and feature extraction. An intuitive way is to divide a periodical trajectory into several segments and adopt the average value of each segment to the model training. Unfortunately, such a method cannot fully capture the feature of the trajectory data and therefore achieves undesirable training results. Another challenge lies in the fact that the data sensed is always more than one dimensional (temperature, humidity, etc.), so the data preprocessing algorithm should be able to deal with multi-dimensional data.

To solve above problems, individual nodes divide a periodical trajectory into several segments, then perform a Fourier transform on the data of each segment. We found that the energy distribution mostly concentrates among several of the lowest frequency components of the transformed data, so these components are able to capture the feature of a trajectory effectively. On the other hand, a high frequency component is probably caused by a single extreme value but may have few contributions to the pattern of a trajectory. So for each segment, we combine the lowest frequency components of several multiple dimensions of the sensed data, so as to form the multi-dimensional preprocessed data. In this way, multi-dimensional data can be combined and compressed with the least information loss.

With the data ceaselessly coming, the preprocessing algorithm is applied when data of a whole segment is received. The preprocessed data is stored in each sensor node and can be utilized in algorithms described in Sect. 3.5.

3.2 Multi-dimensional HMM implementation

Now we introduce how we build our HMMs using the multi-dimensional preprocessed data. In our implementation, the probability density function of each HMM state is assumed to be a multi-dimensional GMM:

$$b_j(\mathbf{O}) = \sum_{m=1}^{M} c_{jm} \mathfrak{N}[\mathbf{O}, \boldsymbol{\mu}_{jm}, \mathbf{U}_{jm}] \quad (1)$$

where the vector $\mathbf{O}$ is the modeled multi-dimensional preprocessed data, c_{jm} is the mixture coefficient for the mth mixture in state j, and $\mathfrak{N}$ is a Gaussian density function, with a mean vector $\boldsymbol{\mu}_{jm}$ and a covariance matrix $\mathbf{U}_{jm}$ for the mth mixture component in state j.

We observe that the parameter $\mathbf{U}_{jm}$ is able to capture the correlation among multiple dimensions, and such multi-dimensional probability density function can be used to approximate an arbitrarily closely, any finite, continuous density function. Hence it can be applied to model a wide range of periodical trajectories of multi-dimensional data in WSN applications.

To estimate the GMM parameters, we use the standard EM algorithm [17], which contains two main steps called E-step and M-step. In the E-step, a segmentation of the training samples is obtained to maximize the likelihood of the data, given the parameters of the GMMs. This is followed by an M-step, where the parameters of the GMMs are re-estimated based on this segmentation. By doing so, the multi-dimensional GMM is obtained and is adopted for the model training in Sect. 3.3.

3.3 Clustering and model training

If each node trains an HMM separately, a serious overfitting problem is more likely to take place due to the lack of variance and neighboring information [18], which may degrade the detection performance. Meanwhile, if we train only one HMM in a centralized manner, large variance and high communication cost will render our method infeasible for WSNs.

We solve the above problems by bringing in spacial correlations and introduce appropriate variances. That is, to group the neighboring nodes with similar trajectories into one cluster, and to train HMM models separately in each cluster.

First each node learns an HMM using the method in Sect. 3.2, denoted by m_i locally according to its training data, i.e., a set of observations $\mathbf{O}_i$. A clustering algorithm then divides the whole network into clusters such that the nodes within one cluster exhibit similar characteristics. To measure the similarity distance between two nodes, we define the distance d_{ij} between two HMMs as:

$$d_{ij} = L(\mathbf{O}_i|m_i) + L(\mathbf{O}_j|m_j) - L(\mathbf{O}_i|m_j) - L(\mathbf{O}_j|m_i) \quad (2)$$

Note that while our approach is cluster-based, the construction of initial clusters is out of the scope of this work. Numerous recent studies, e.g. [18–20], have focused on sensor node clustering as well as cluster head selection. We bear this in mind and state that many of these algorithms can be used in conjunction with our algorithm as long as it is based on the similarity measurement.

Accordingly, each cluster is associated with a set of corresponding HMMs. Note that there are probably more

than one normal trajectory; if we only train one model in each cluster, the large variation will lead to a worse detection performance. So each cluster head, with the knowledge of all trajectories in this cluster and N HMMs, will regroup all these HMMs into new models. Similar to that in [21], it works as follows. We first calculate the distance d_{ij} between two trajectories $\mathbf{O}_i$ and $\mathbf{O}_j$ by Eq. (2). When there exists a distance between two trajectories is less than a given threshold, we merge two HMMs m_i and m_j associated with $\mathbf{O}_i$ and $\mathbf{O}_j$, respectively, and then the problem becomes how to merge those two HMMs into a new HMM m_{ij}. To this end, we recalculate a new set of parameters θ^*_{ij} of the usual-event HMM model by maximizing the likelihood of observations as:

$$\theta^*_{ij} = \arg\max_{\theta_{ij}} \left(\log P(\mathbf{O}_j|m_{ij}) + \log P(\mathbf{O}_i|m_{ij})\right) \quad (3)$$

where θ_{ij} are the parameters with the HMM m_{ij}. Afterward, multiple HMMs will be merged and multiple trajectories will be used in the model training with the method in Sect. 3.2.

Finally the cluster head obtains several HMMs indicating the usual events. Often the number of HMMs is small since our trajectories are based on a long time period. These trained HMMs profiling usual trajectories are diffused to every node within the cluster for the use of the outlier detection.

3.4 Dynamic model update

Often when the cluster head receives the data from a sensor node, it will update its local HMMs accordingly, so that the cluster head can always offer a timely and updated HMMs to each node for efficiently detecting the outlier. Given a new trajectory $\mathbf{O}'_i$, first the cluster will select the *most matched* HMM by $m^*_k = \arg\max_k P(\mathbf{O}'_i \mid m_k)$. We assume here that the cluster head holds a buffer to maintain the history trajectory set. If there are K trajectories that can be modeled by m_k, each of them is associated with a timestamp. When we train a new HMM m^*_k, the timestamp can be used as a weigh value to indicate the importance of current data compared with history information.

To train the new HMM m^*_k, unsupervised incremental Maximum Likelihood Linear Regression (MLLR) [16] represents an effective way to progressively adapt an initial set of continuous density HMMs to the current conditions. For HMM adaptation, the means and variance of Gaussian components in the system are adapted using transformation metrics associated with 8 static regression classes. For each regression class, a block diagonal transformation matrix (with blocks corresponding to static features and the first and second order time derivatives, respectively) is utilized to adapt the means, while a diagonal transformation matrix is for the variances. The model parameters are adapted after the processing.

3.5 Trajectory-based outlier detection

3.5.1 Static trajectory-based outlier detection

After multiple HMMs that describe the normal trajectories are learned in each cluster head and diffused to all the nodes within the cluster as described in Sect. 3.3, the outlier detection then can be performed to new trajectories locally at each node. Specifically, given a new trajectory $\mathbf{O}_i$ collected by a sensor node, we calculate the probability of observing $\mathbf{O}_i$ given any HMM of normal events, denoted by $P(\mathbf{O}_i \mid m_k)$, where m_k represents one of the HMMs. An outlier can be detected when the maximum probability is less than a given threshold, i.e.,

$$\max_k P(\mathbf{O}_i \mid m_k) < Th_A \quad (4)$$

where Th_A is the predefined threshold. Slightly different with that in Sect. 2.2, we use the learned multiple HMMs to describe the possible normal events instead of one HMM that often incurs too large variations.

3.5.2 Real-time trajectory-based outlier detection

The static detection algorithm requires the data of the entire observed trajectory to precisely determine whether such trajectory appears abnormal. However, many applications of sensor networks require real-time outlier detection schemes. We claim that our trajectory-based outlier detection can be modified to fit such real-time feature of the sensor networks.

First, we present that the outlier detection can be performed using several segments of the trajectory instead of the whole trajectory. In our experiment, a one-day period-based trajectory is divided into 8 segments, and we use the data of 1–8 segments to detect the outlier. The performance shown in Sect. 4 implies that the trajectory changes can be detected without the appearance of the whole trajectory.

For emergencies that cause great trajectory change, we propose a real-time trajectory-based outlier detection method based on the segment detection method. During the updating procedure, each cluster head chooses one trajectory of each category of the HMM it has learned and sends them to other nodes. Every a short time we use each of these trajectories to estimate (predict) the upcoming data, and then we complete the current segment by connecting the data we have already obtained in the current segment and the estimated upcoming data of the current segment, so there are k possible current segments (assume k HMMs are learned). Denote the possible trajectory of the combination of k possible estimated current segment and segments that

have already arrived as $\mathbf{O}_h, h = 1, 2 \ldots k$, an outlier can be detected when $\max_{k,h} P(\mathbf{O}_h \mid m_k) < Th_A$.

We use an example to illustrate our algorithm: in our experiment, we apply the real-time detection algorithm for fire alarm. Assume a fire happens at 4:30 am. Accordingly the data of the first segment (0:00 am–3:00 am) has already been captured and preprocessed. We combine the obtained data (3:00 am–4:30 am) and the estimated upcoming data (4:30 am–6 :00 am) to form the estimated current segment. Then we denote the trajectory of the current segment together with the segment already obtained as $\mathbf{O}_h$, and estimate the likelihood of observing $\mathbf{O}_h$. Such algorithm is done every 2.5 min. So after 2.5 min, a sudden rise in temperature which changes the trajectory of the current segment can be obtained, and as a result, the fire can be detected.

3.6 Discussion

As data is segmented and compressed in the preprocessing procedure, the number of segments, hidden states and observation symbols (Gaussian density function) are all restricted. Our experiments in Sect. 4 show that 8 segments, 5 states and 4 symbols are often enough to model a daily trajectory composed of 2,880 data. Our analysis below will show the feasibility to apply the proposed method in WSNs. Note that we only consider the detection cost and model updating cost as the algorithm setup procedure is carried out only once.

Storage occupation A node needs to store: (1) the preprocessed data that has not updated to its cluster head yet; (2) the HMM of normal trajectories; (3) the trajectory data of each HMM to conduct real-time detection. In our experiment, data of a daily trajectory is about 23 k, and becomes 0.3 k after preprocessing; an HMM model is about 20 k. So a sensor node with 512 k memory is more than sufficient.

Computational cost The computational complexity of our detection algorithms is $O(N^2 L)$, where N is the number of states and L is the number of segments. Such cost is fairly small considering the size of N and L. The complexity of FFT is $O(n \log(n))$, where n is the number of data points in one segment. We note that such cost is affordable to a sensor node. The complexity of the model updating is comparatively higher, but as the size of parameters is small, it is still available to operate in cluster heads.

Communication cost As the spacial correlation is introduced by the model training instead of by the communication, the outlier detection process can be run inside each node. So the only communication occurs at the periodic model updating procedure. But as the size of the HMM and preprocessed data is small, the communication cost is trivial accordingly.

4 Performance evaluation

To evaluate the performance of the proposed outlier detection scheme, we conduct a series of experiments. We have implemented a simulator and quantified several performance aspects of our scheme. We first describe our experimental evaluation methodology and then present the results and the analysis.

4.1 Experimental setup

Our experiments are conducted using the well-known Intel lab dataset—a real dataset generated by 54 sensors located in one room. The sensors sense data of various properties every 30 s for more than one month. We experiment our scheme by testing the abnormal trajectory of the temperature and voltage data in a one-day period. Data of the first 15 days is utilized as the training set, while data of the rest 10 days is considered as the testing set. We assume no consecutive and significant noise in the training set, so the accuracy of our trained HMMs will be within the acceptable range for detecting the abnormal trajectories. We randomly add 20 % of different kinds of abnormal trajectories as outliers to form the testing set, i.e., there are more than 750 trajectories in the training set and more than 500 trajectories in the testing set. Each experiment is run for 100 times to leverage the randomness.

We divide the daily sensed data stream into 8 segments, and to each segment we adopt the Fourier transform to the temperature data, then we take the lowest 4 components of the transformed data together with the voltage data to form a 5 dimensional preprocessed data, as we note that variations in voltage are highly correlated with temperature (as the voltage provided by lithium batteries is affected by temperature fluctuations).

We firstly adopt the clustering algorithm [19] to divide the sensors into 9 clusters using the data of the first 15 days. Then to each cluster, we regroup the HMMs to train 4 HMMs as 4 normal trajectories in each cluster. These models are directly used to estimate the likelihood of observing a coming trajectory for the next 10 days.

There are two types of errors in the trajectory-based outlier detection, i.e., the false alarm (FA) error, when the method passes a normal trajectory, and the false rejection (FR) error, when the method rejects an abnormal trajectory. Accordingly, we adopt the false alarm rate (FAR) and the false rejection rate (FRR) to evaluate the performance.

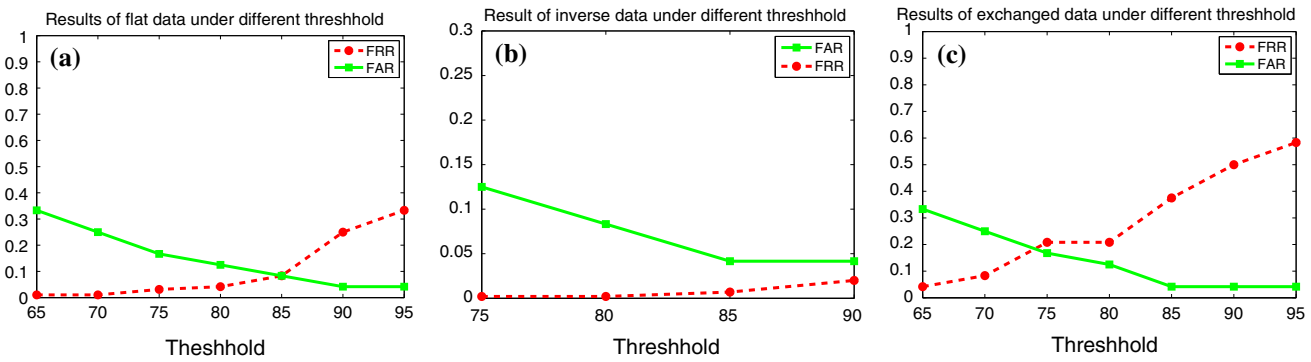

Fig. 1 Performance with different abnormal trajectories. **a** Performance with flat data. **b** Performance with inverse data. **c** Performance with exchanged data

4.2 Experimental results

4.2.1 Off-line trajectory-based outlier detection

We note that most abnormal trajectories are combinations of some basic trajectory changes such as shifting, reversing and scrambled ordering etc., and a flat trajectory is more likely to be presented by an idle sensor node. So in our experiment, we use these abnormal trajectories as our added outliers. Figure 1 shows the performance of the static outlier detection algorithm facing these basic trajectory changes with different thresholds. The threshold reflects the sensitivity of the detecting algorithm. Trajectories are more likely to be detected as outliers under a lower threshold and verse vice. Accordingly, as the threshold increases, the method is more likely to pass a normal trajectory, and less likely to reject an abnormal trajectory, resulting in an increasing FRR and a decreasing FAR. In Fig. 1(a) and (b), it can be seen that the reversed and flat trajectories can be precisely detected. In Fig. 1(c) we note that the exchanged data, which indicates the scrambled ordering phenomena, achieves a comparatively mediocre performance. The results could also offer an empirical method to choose an appropriate threshold. For example, for reversed and flat trajectories, higher thresholds lead to more precise detecting results, and a threshold in the range, say 80–90, can be regarded as a good threshold. Thus different applications can choose different thresholds according to the corresponding level of the sensitivity requirement.

We evaluate the sensitivity of our scheme to the trajectory shift by the recognition rate. Outliers are produced by adding an offset to the normal trajectories. In Fig. 2(a) we observe that less than 10 % of the shifted temperature trajectories are considered as outliers if the offset is within 0.5° C, and more than 80 % are recognized as outliers with a shift more than 3° C. Similar results are shown in Fig. 2(b) to the voltage value shifting. We note that small shifts can be tolerated and large shifts are more likely to be detected in the above experiments, and the abnormal value of either the voltage or temperature data can be detected. Such results validate that our HMM implementation is able to approximate the trajectory pattern of the multi-dimensional data reasonably and efficiently. As mentioned before, a threshold value in a good threshold range leads to acceptable results, we therefore choose the same threshold for all clusters. Note that we can choose different thresholds to fit different sensitivity demands of the shifted value according to various application requirements.

We note that our experiment results above cannot be directly compared with other outlier detection algorithms in WSNs [5–11]. Because they concentrate on the single outlier with a much higher or lower value compared with its neighboring data, while we focus on trajectory-based outliers that obviously differ from normal trajectories, which is more challenging and practical to many applications in sensor networks.

4.2.2 Real-time trajectory-based outlier detection

As mentioned in the experiment setup part, we divide the daily trajectory into 8 segments, and we use the data of 1–8 segments as the testing data to detect outliers. Reversed trajectories are added as outliers to test the performance of segmental outlier detection. Figure 3 shows the tendency that the detect performance will be better if more segmental data is obtained.

To evaluate the performance of the real-time trajectory-based outlier detection, we adopt the real data trajectory of a fire hazard. We randomly insert the occurrence of a fire into normal trajectories as outliers. A real-time outlier detection is done every 2.5 min. A sudden and violent trajectory change will still be captured under the Fourier

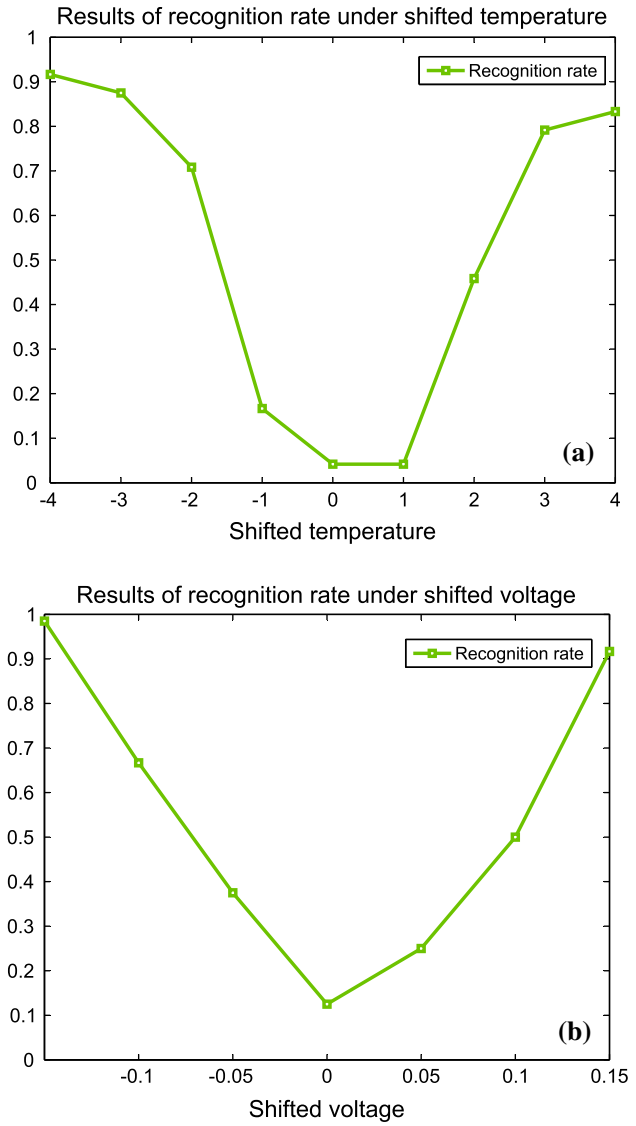

Fig. 2 Recognition rate under shifted temperature and voltage. **a** Performance with shifted temperature. **b** Performance with shifted voltage

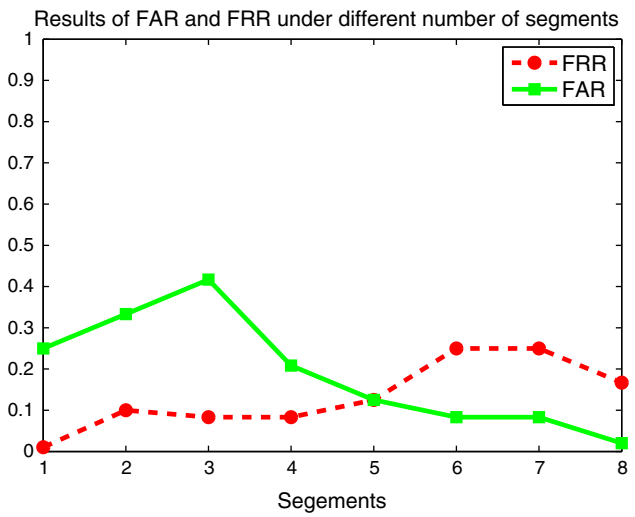

Fig. 3 Detection performance using segment data

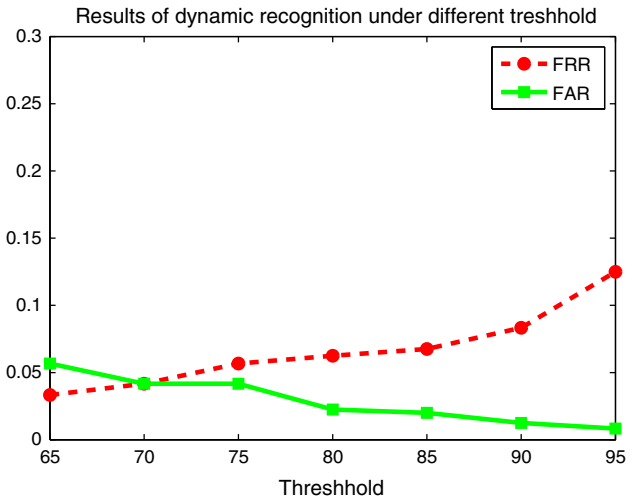

Fig. 4 Performance of the dynamic recognition algorithm

transform after connecting with the following estimated trajectories, and consequently such changes will be recognized as outliers. The satisfactory result in Fig. 4 shows that more than 95 % of the fire can be detected within 5 min.

Above experimental results show that our algorithm does well on the static outlier detection and achieve satisfactory performance on the real-time outlier detection when the trajectory change is notable. We note that such results fulfill the real need and achieve reasonable performance: slight changes in trajectories can be tolerated; obvious abnormal trajectories can be distinguished if enough information of such trajectory is attained; and sudden and violent trajectory changes that often imply emergencies can be detected immediately.

5 Related work

Outlier detection has been well studied in the database research community, and some researches have tried to address the outlier detection problem in the context of sensor networks. Despite the significant progress they have achieved, we note that further analysis of the outlier detection in WSNs is still an important and challenging problem, as some high semantic outliers based on the trajectory change may barely be detected using existing techniques.

Sheng et al. [6] proposed a histogram-based method where the sink collects histogram information among sensor nodes to extract the data distribution throughout the network and then identify global outliers. Subramaniam et al. [12] used kernel density estimator to approximate the

underlying distribution of sensory data, and it is scalable to detect multi-dimensional outliers. We note that such distribution-based outlier detection methods have some drawbacks. First, some outliers cannot be effectively represented by the distribution information, such as the order reversal etc. Second, the distribution function can only reflect short data series, so long-term gradual trajectory changes can barely be detected.

Branch et al. [5] proposed a nearest neighbor-based approach to detect global outliers. Zhuang et al. [13] used a wavelet transform and a dynamic time warping (DTW) based distance measurement to detect and correct short simple outliers. Gao et al. [10] presented a DTW-based segment outlier and unusual event detector, incorporating expert knowledge via semantic inferencing rules. Zhang et al. [8] proposed an ellipsoidal support vector machine based online outlier detection scheme, and Moshtaghi et al. [11] proposed an ellipsoidal model for anomaly detections; they both utilized the Mahalanobis/Euclidean distance to measure the similarity between any two data vectors. Except for the high communication overhead, these methods based on short-term neighboring data comparison are unavailable to detect long-term gradual trajectory changes.

As is discussed before, almost all existing works are based on short-term adjacent data analysis. So these approaches can barely achieve good performances towards the high semantic trajectory outliers as mentioned in Sect. 1. What is more, although some techniques above can be scalable to multi-dimensional data [7, 9, 12], none of them considers the correlation between the multiple dimensions. A multi-dimensional outlier as mentioned in Sect. 1 may probably be left out of considerations therefore.

6 Conclusion

In this paper, we demonstrate the problem of high semantic outlier detections in sensor networks by concrete examples. We motivate our work in the context of the challenging problems and propose the first trajectory-based outlier detection method in WSNs, to the best of our knowledge. Our multi-dimensional HMM implementation enables our algorithm dealing with multi-dimensional data. The proposed data preprocessing, clustering, model training and model updating methods are developed to fit the feature of sensor networks and to enhance the detection performance. In addition, our algorithm can run in a real-time manner. Simulation results show our algorithm achieves a reasonable good performance facing various abnormal trajectories.

Acknowledgments This work was supported in part by the National Natural Science Foundation of China under Grants 61202460 and 61271226; by the Fok Ying Tung Education Foundation under Grant 132036; by the Fundamental Research Funds for the Central Universities under Grants 2014XJGH003, 2014QN158 and 2014QN164; by the Program for New Century Excellent Talents in University under Grant NCET-10-408 (State Education Ministry); and by the CCF-Tencent Foundation under Grant CCF-TencentAGR20130101.

References

1. Cayirci, E., & Coplu, T. (2007). SENDROM: Sensor networks for disaster relief operations management. *Wireless Networks*, *13*(3), 409–423.
2. Jiang, H., Jin, S., & Wang, C. (2010). Parameter-based data aggregation for statistical information extraction in wireless sensor networks. *IEEE Transactions on Vehicular Technology*, *59*(8), 3992–4001.
3. Conti, M., & Giordano, S. (2014). Mobile ad hoc networking: Milestones, challenges, and new research directions. *IEEE Communications Magazine*, *52*(1), 85–96.
4. Zhang, Y., Meratnia, N., & Havinga, P. (2010). Outlier detection techniques for wireless sensor networks: A survey. *IEEE Communications Surveys & Tutorials*, *12*(2), 159–170.
5. Branch, J., Giannella, C., Szymanski, B., Wolff, R., & Kargupta, H. (2013). In-network outlier detection in wireless sensor networks. *Knowledge and information systems*, *34*(1), 23–54.
6. Sheng, B., Li, Q., Mao, W., & Jin, W. (2007). Outlier detection in sensor networks. In: *Proceedings of 8th ACM MobiHoc* (pp. 219–228).
7. Zhang, Y., Hamm, N. A., Meratnia, N., Stein, A., van de Voort, M., & Havinga, P. J. (2012). Statistics-based outlier detection for wireless sensor networks. *International Journal of Geographical Information Science*, *26*(8), 1373–1392.
8. Zhang, Y., Meratnia, N., & Havinga, P. J. (2013). Distributed online outlier detection in wireless sensor networks using ellipsoidal support vector machine. *Ad Hoc Networks*, *11*(3), 1062–1074.
9. Burdakis, S., & Deligiannakis, A. (2012). Detecting outliers in sensor networks using the geometric approach. In: *Proceedings of 28th IEEE international conference on data engineering (ICDE)* (pp. 1108–1119).
10. Gao, L., Bruenig, M., & Hunter, J. (2013). Semantic-based detection of segment outliers and unusual events for wireless sensor networks. In: *Proceedings of 18th international conference on information quality (ICIQ)* (pp. 127–144).
11. Moshtaghi, M., Leckie, C., Karunasekera, S., & Rajasegarar, S. (2014). An adaptive elliptical anomaly detection model for wireless sensor networks. *Computer Networks*, *64*(8), 195–207.
12. Subramaniam, S., Palpanas, T., Papadopoulos, D., Kalogeraki, V., & Gunopulos, D. (2006). Online outlier detection in sensor data using non-parametric models. In: *Proceedings of 32nd international conference on very large data bases (VLDB)* (pp. 187–198).
13. Zhuang, Y., & Chen, L. (2006). In-network outlier cleaning for data collection in sensor networks. In: *Proceedings of 32nd international conference on very large data bases (VLDB)* (pp. 1–8).
14. Bhattacharya, A., Meka, A., & Singh, A. K. (2007). Mist: Distributed indexing and querying in sensor networks using statistical models. In: *Proceedings of 33rd international conference on very large data bases (VLDB)* (pp. 854–865).

15. Gales, M., & Young, S. (2008). The application of hidden Markov models in speech recognition. *Foundations and Trends in Signal Processing*, *1*(3), 195–304.
16. Gales, M. J., & Woodland, P. (1996). Mean and variance adaptation within the MLLR framework. *Computer Speech & Language*, *10*(4), 249–264.
17. Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, *77*(2), 257–286.
18. Razzaque, M., Bleakley, C., & Dobson, S. (2013). Compression in wireless sensor networks: A survey and comparative evaluation. *ACM Transactions on Sensor Networks, 10*(1), 5:1–5:44.
19. Jiang, H., Jin, S., & Wang, C. (2011). Prediction or not? An energy-efficient framework for clustering-based data collection in wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, *22*(6), 1064–1071.
20. Abbasi, A. A., & Younis, M. (2007). A survey on clustering algorithms for wireless sensor networks. *Computer Communications*, *30*(14), 2826–2841.
21. Jiang, F., Wu, Y., & Katsaggelos, A. K. (2009). A dynamic hierarchical clustering method for trajectory-based unusual video event detection. *IEEE Transactions on Image Processing*, *18*(4), 907–913.

Chen Wang received the B.S. and Ph.D. degrees from Department of Automation, Wuhan University, China, in 2008 and 2013, respectively. He has been a post-doc fellow at Huazhong University of Science and Technology since June 2013. His recent research interests include wireless networking and communication protocols.

Hongzhi Lin received the B.S., M.S., and Ph.D. degrees from HuaZong University of science and technology, China, in 2000, 2003, and 2008, respectively, all in electronic and information. He has published more than 10 papers in journal in the areas of wireless communication and networking. His current research interests are in the areas of wireless networking and digital signal processing.

Hongbo Jiang received the B.S. and M.S. degrees from Huazhong University of Science and Technology, China. He received his Ph.D. from Case Western Reserve University in 2008. After that he joined the faculty of Huazhong University of Science and Technology where he is a full professor. His research concerns computer networking, especially algorithms and architectures for high-performance networks and wireless networks.