Using Temporal Correlation and Time Series to Detect Missing Activity-Driven Sensor Events

Juan Ye
School of Computer Science
University of St Andrews
St Andrews, Fife, UK
Email: juan.ye@st-andrews.ac.uk

Graeme Stevenson
School of Computer Science
University of St Andrews
St Andrews, Fife, UK
Email: graeme.stevenson@st-andrews.ac.uk

Simon Dobson
School of Computer Science
University of St Andrews
St Andrews, Fife, UK
Email: simon.dobson@st-andrews.ac.uk

***Abstract*—Increasing numbers of sensors are being deployed in environments to monitor our behaviours and environmental phenomena. Missing data is an inevitable problem in almost every sensorised environment, due to physical failure, poor connection, or dislodgement. This results in an incomplete view of the real-world, leading to poor prediction and consequently, degraded quality of system services. This paper explores generic solutions towards detecting missing data on event-driven sensors using both temporal correlation and time series analysis. The solutions are evaluated on a real-world dataset and achieve promising results with accuracy around 80%.**

I. Introduction

Missing sensor data is a common, nearly inevitable issue in many sensorised environments which can be caused by temporary or permanent communication disconnection, battery failure, or the physical degradation of sensors [9]. The effect of missing sensor data is an incomplete view on the real world, which can greatly impact a system's capability to infer the current situation or activity being carried out by a person, leading to degradation of the quality of services provided by the system as a result.

Missing data is a well understood problem in wireless sensor networks, and many techniques have been proposed to address numeric-valued, frequent-sampling sensors, including Kalman filters [11], time series analysis [10], and Gaussian Processes [8]. However, the missing data problem on characteristic-valued, event-driven sensors has not been well studied, because such data is less regular and predictable. Sensors that fit this category include RFID, state-change sensors, and infrared positioning sensors, all of which have been widely used in environments to detect human activities.

A naive approach to detect missing data from these sensors is to set a reasonably long period (say one day) over which if the sensor has not reported any reading, then it is considered to have fail. This approach only targets detection of the complete breakdown of sensors, and, as such, is unable to detect intermittent missing data from the sensors. Furthermore, the firing of sensors is closely related to activities and some activities might not be performed over a long period (say one week). For example, if the user does not cook spaghetti for a long time, then the sensor on the spaghetti box will stay inactive during that time, or if the user goes out for a couple of days, then no activities will be performed and hence no sensor will fire. Clearly, this does not imply their failure.

From above, we can see that the challenge of detecting missing data on such sensors is a subtle one. In this paper, we propose a generic technique to detect missing data in event-driven sensors using temporal correlation and time series analysis. The temporal correlation captures how different sensors are correlated in a streaming sequence of sensor events. Given that two sensors are highly correlated, if one of the sensors fires, then we would expect the other also to fire. The time series characterises the firing intervals of individual sensors, reflecting their usage patterns, which suggests a pattern of activities, and thus can be used to predict the next firing time. According to the recorded firing history of a sensor, if we predict the sensor should fire at the current time and it does not, then we derive the missing data from this sensor.

The proposed technique is evaluated on the dataset from the University of Amsterdam [5] (known as 'House A'). The dataset was collected from the real-world single-resident house instrumented with wireless sensor network. The sensor network in the first house is composed of 14 state-change sensors on the household objects like doors, cupboards, and toilet flush. All these sensors output binary readings (0 or 1), indicating whether or not a sensor fires. This dataset is quite clean, that is, involving less noise and leading to high recognition accuracies over various techniques. Thus it is a good candidate for proof of concept; that is, it allows us to focus on the missing data while not worrying about the other noise [12]. We will form our analysis and discussion on this dataset throughout the paper.

The rest of the paper is organised as follows. Section II reviews the existing work in dealing with missing sensor data. Section III explores the solution space in both temporal correlation and time series of sensor events from both assumptions and theoretical background. Section IV sets up the experiments to evaluate the different strategies of combining the two solutions and compares and discusses the implications of each strategy. Section V concludes the paper and points out the direction of the future work.

II. Related Work

The missing data problem is well recognised in the field of wireless sensor networks. From an early stage, Madden et al. [7] propose to estimate missing values by taking the average of all the values reported by nearby sensors during the same time window. Jiang et al. [3] improve this approach

by making use of a sliding window where the values reported in the latest w time windows are considered and the weighted average is performed; that is, the more recent value is assigned with a higher weight. Ciampi et al. [1] go further to take spatial correlation into account and propose a trend cluster discovery process to determine prominent data trends and estimate the missing data from the recorded geographically data window. Pan et al. [9] design the K-nearest neighbour estimation algorithm to estimate the missing data based on the spatial correlation of sensor data. In our work, we only consider the temporal correlation of sensors. The reasons are: (1) smart home environments are often much smaller than the open environments where the above techniques apply and thus it becomes feasible to consider all the sensors together without the need of separating them into regions; and (2) the spatial correlation is more useful to detect added noise rather than missing data. That is, if the sensors in one region report, then we cannot guarantee all the other sensors in the same region should fire but we can specify that a sensor in another disjoint region should not report.

Approaches from the field of database management have been applied to handle missing data in sensor streams, like sampling, histograms, and wavelets. For example, Vijayakumar et al. [11] model the input sensor stream as a time series and use Kalman filters to predict the missing event. The Kalman filter is an optimal recursive data processing and mathematical estimation algorithm that is often used for data assimilation and prediction. In this work, they only consider the univariate time series that consists of single observations recorded sequentially over equal time increments. Due to the different nature of event-driven sensors, we consider the non-linear time series, which will be explained in details in Section III.

Osborne et al. [8] use the Gaussian process to build a probabilistic model of the environment variables being measured by the sensors, which is supposed to be tolerant to missing data. The model then can be used to model the accuracy of the sensor readings, and predict the future reading and as well as the trend of change of the environment variables.

All the above approaches target scalar observations that sample frequently and regularly. These observations often measure a single environmental variable or multiple environment variables that are highly correlated, for example, temperature and humidity. They are not applicable to event-driven sensors due to the numeric nature of the observations and their sampling frequencies.

Gruenwald et al. [2] apply the association rule mining algorithm to detect missing data. One example of the rule is: if a sensor A reports a value v, then it is very likely that a sensor B will report a value u. Based on the original association mining algorithm, the approach associates more recent data with a higher reliability. The principle of this work is similar to ours. However, the way to generate transactions as the input for the algorithm still requires the regularly sampled sensor stream. In Section III, we will discuss different ways to generate temporal correlations of sensors.

III. Solution Framework

In this section, we explore the solution space in terms of temporal correlation and time series. We will look at different types of correlations and what series to construct, which one is more suitable for detecting missing data, and how we use them to detect.

A. Temporal Correlation of Sensors

Generally we consider two types of temporal relationships between sensors: *sequential* – where one sensor reports before another sensor, and *correlation* – where two sensors report within a close time interval (say, one minute) but in no particular order. For each type, we look at *continuous* and *discontinuous* relationships. In the following we present a detailed definition of these temporal relationships and also give examples of their occurrence from within our chosen dataset.

Let a sensor event be represented as (t, s), indicating a sensor s fires at the timestamp t, and the entire sensor stream be represented as $L = \langle (t_1, s_1), ..., (t_n, s_n) \rangle$, where n is the total number of events. *Continuous sequential* relationship (CS) captures where one sensor fires immediately after the other sensor in the entire sensor stream. The CS relationship between any two sensors s_i and s_j is defined as [6]:

$$CS(i,j) = \frac{\sum_{k=1}^{n-1} \sigma((t_k, s_k), (t_{k+1}, s_{k+1}))}{n}$$

$$\sigma((t_k, s_k), (t_{k+1}, s_{k+1}) = \begin{cases} 1 & \text{if } s_k = s_i \wedge s_{k+1} = s_j \\ 0 & \text{otherwise.} \end{cases}$$

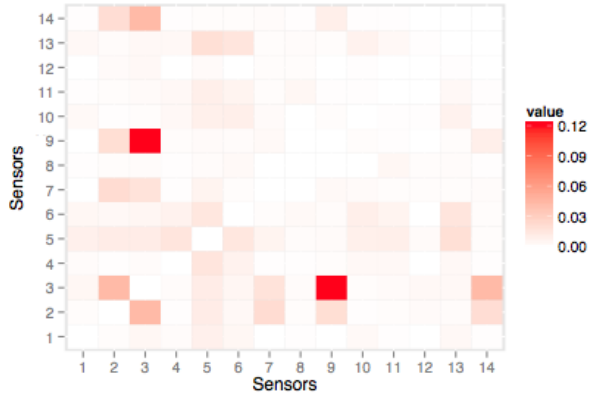

Fig. 1. Continuous correlations between sensors

The *continuous correlation* relationship (CC) captures when two sensors s_i and s_j indicate fire consecutively, ignoring order: $CC(i,j) = CS(i,j) + CS(j,i)$. Often the CS relationship is rather sparse because human users rarely follow the exactly same procedure when performing an activity. Thus, we only present the CC relationships in the dataset in Figure 1. The figure shows that the highest continuous correlation exists between the bathroom door (i.e., Sensor 3) and the toilet flush (i.e., Sensor 9), which takes up 12% and can be observed in the majority of toileting activities.

As continuous sequential relationships are few, we also examine *discontinuous sequential* relationship (DS) where one sensors fires after the other sensor within a certain interval in the entire sensor stream. The discontinuous sequential

relationship between any two sensors s_i and s_j within the interval d is defined as:

$$DS(i,j,d) = \frac{\sum_{k=1}^{n-1} \sigma((t_k, s_k), (t_{k+l}, s_{k+l}))}{n} (l \geq 1 \wedge k + l \leq n)$$

$$\sigma((t_k, s_k), (t_{k+l}, s_{k+l}) = \begin{cases} 1 & \text{if } s_k = s_i \wedge s_{k+l} = s_j \\ & \wedge |t_{k+l} - t_k| <= d, \\ 0 & \text{otherwise.} \end{cases}$$

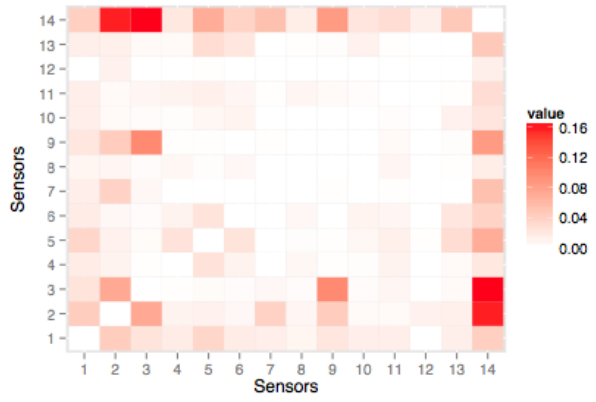

Fig. 2. Discontinuous correlations between sensors

Similarly, the *discontinuous correlation* relationship (DC) is defined as $DC(i,j,d) = DS(i,j,d) + DS(i,j,d)$, indicating that two sensors fire together within a certain time interval. Figure 2 visualises the DC relationships within the 1-minute time interval on the dataset. Figure 2 (a) shows that the highest discontinuous correlations exist between the bed, bedroom door, and bathroom door, which are often observed when the user goes to toilet during the night sleep.

From the above analysis and demonstration, we can see that the temporal correlation can suggest the pattern of underlying activities. Here we will use a less strict correlation – the DC relationship – to detect the missing sensor data. The process is as follows:
1. Use the DC formula to construct the correlation matrix using a fixed interval (say 1 minute).
2. Choose a threshold θ, over which two sensors are considered highly correlated. If one of these sensors fires and the other does not fire within the interval, then we conclude that the other sensor has failed to report.

Instead of pre-defining a threshold, we use a data-driven approach where we order all the non-zero values in the correlation matrix in an ascending order and choose a value that is greater than a percentage (e.g., 60%) of all the values. This is more flexible than the pre-defined threshold, which could vary between datasets; that is, a dataset that contains more sensors will have a sparse matrix, and the threshold will be much smaller.

B. Non-linear Time Series of Sensors

As we have indicated, event-driven sensors do not report readings at a regular interval; however, the firing history of a sensor might suggest the pattern of how a user interacts with an object or a place associated with this sensor. To characterise the firing history, we construct the time series that is composed of the temporal distance between events of the same sensor. For example, the list $[163, 33044, 6, 28045, 7222]$ represents the temporal distance (in seconds) between events that are reported from the bedroom door; that is, the distance between the first and second report of this sensor is 163 seconds, and that between the second and third is 33044 seconds (i.e., 9 hours), and so on so forth. This list could imply the sleep pattern of the subject; i.e., the sleeping cycle and the night toilet trip. Figure 3 shows the time series of the temporal distances of all the sensors in the dataset. To note that the x-axis indicates the serial number of reports, but not the regular interval; e.g., '1' means the first time of the sensor reporting an event. Also the number of reports varies from sensors to sensors; e.g., the sensors 2 and 3 (mounted on the toilet and bathroom door) fire far more frequently than the sensor 12 (mounted on the washing machine).

As the pattern of this temporal distance series depends on the underlying activities, rather than linear being with time, we adopt the nonlinear time series analysis technique to predict the next reporting time for each sensor. By definition, the time series is a sequence of scalar measurements of some quantity which depends on the current state of the system [4]. Here we only explain how to use the nonlinear time series to predict the next reporting time of a certain sensor and direct the interested readers on the theory to the reference [4]. The process goes as follows:
1. Construct a time series $T = (o_1, o_2, \ldots, o_n)$, consisting of time distances between two consecutive readings of a sensor.
2. At the current time i, given the latest recorded trace of this sensor $c = (o'_{i-m}, \ldots, o'_{i-2}, o'_{i-1})$, which is m-dimensioned and consecutive, search the time series T and find all the segments S that are similar to c; that is, $\forall s \in S$, $s = (o_{j-m}, \ldots, o_{j-2}, o_{j-1}, o_j)$, $\forall k \in [1, m]$, $|o_{j-k} - o'_{i-k}| \leq \tau$, where τ is the time tolerance, indicating if the time distance between the paired elements is no greater than τ, then we consider the paired elements are close. If all the paired elements between the given trace c and a segment s are close, then we consider s is similar to c, and o_j in s suggests one possible next predicting time distance.
3. The predicted time distance from the last report time is estimated by averaging all the next values; that is, $td = \frac{\sum_{s \in S} o_j}{|S|}$. The next reporting time for this sensor is then the last reported timestamp plus td. If the next reporting time is close to the current timestamp, then we expect the sensor to report; otherwise, we derive that the data is missing from this sensor.

IV. Experiment and Evaluation

In this section, we evaluate the effectiveness of the proposed algorithm on the dataset [5]. The effectiveness is measured in *precision* – the percentage of the times of detected missing events are actually missing from the data, and *recall* – the percentage of all missing events that are correctly detected.

A. Experiment Setup

To conduct the experiment, we take the whole dataset and split it into two parts: *training* and *testing* sets. The training set is used to compute the correlation matrix between sensors and

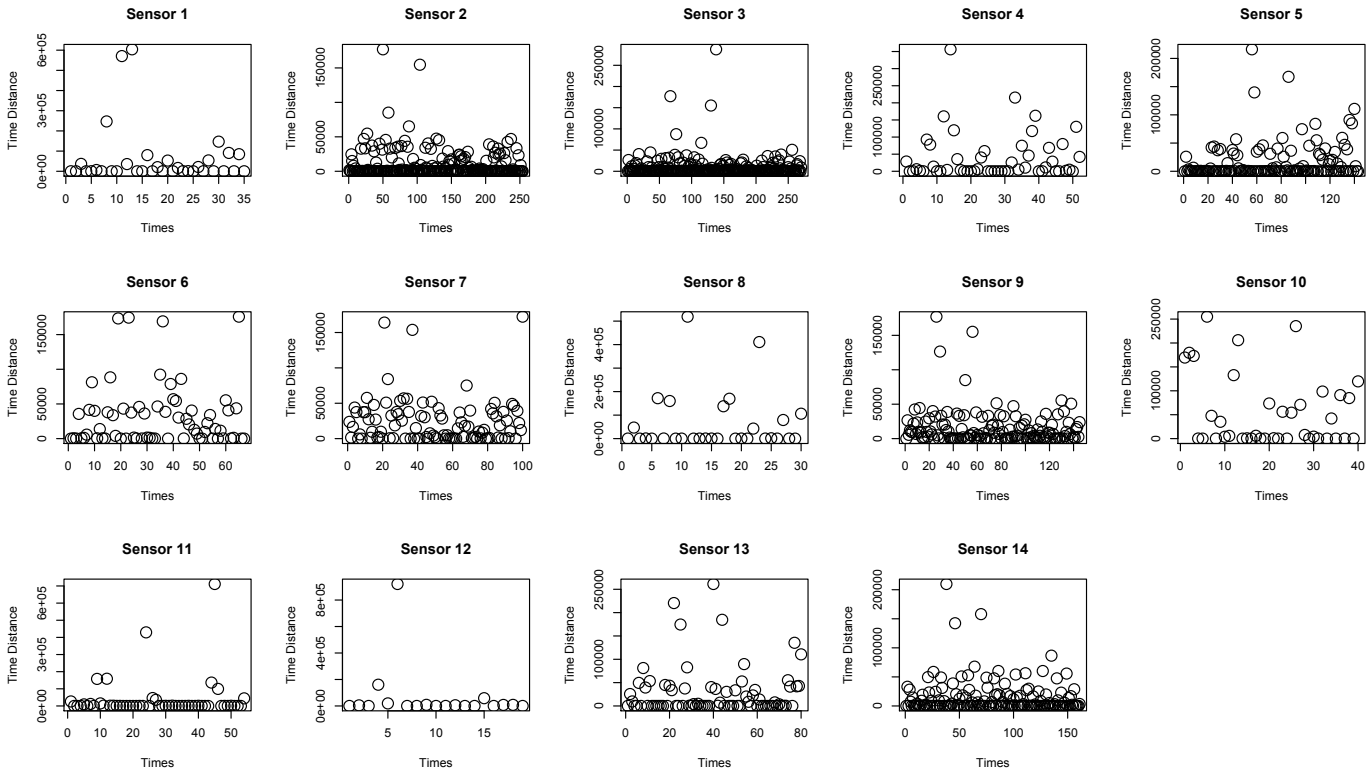

Fig. 3. Time series of temporal distances of all the sensors in TVK A dataset

construct the time series of firing patterns of each individual sensor. Both require that the data are continuous, so when generating the training and testing set we cannot randomly shuffle or sample from the dataset. Given a split rate (say 20%), we take the first 20% of the dataset as the training set, and the rest 80% as the testing set. We have set the split rates from 10% to 90% to see how the amount of training data affects the effectiveness of detection.

For the testing set, given an error percentage p , we randomly choose the number of events (that is, $p * N$, where N is the total number of events in the testing set) to remove. The removal percentage is also chosen from 10% to 90%. For each percentage, we run the experiment 100 times to avoid bias on certain sensors or events. The presented precision and recall is the averaged result over the 100 runs.

We conduct the following three types of experiments:
1. TC – use the temporal correlation alone and evaluate the impact of the correlation threshold θ on the performance;
2. TS – use the time series alone and evaluate the impact of the chosen dimension m and time tolerance margin τ on the performance;
3. TC + TS – use both the temporal correlation and the time series; that is, we join the results from both, and evaluate the impact of the amount of training data on the performance.

B. Results

1) Temporal Correlation Only: In the temporal correlation strategy, we need to choose a threshold over which two sensors are considered correlated; that is, if one sensor has

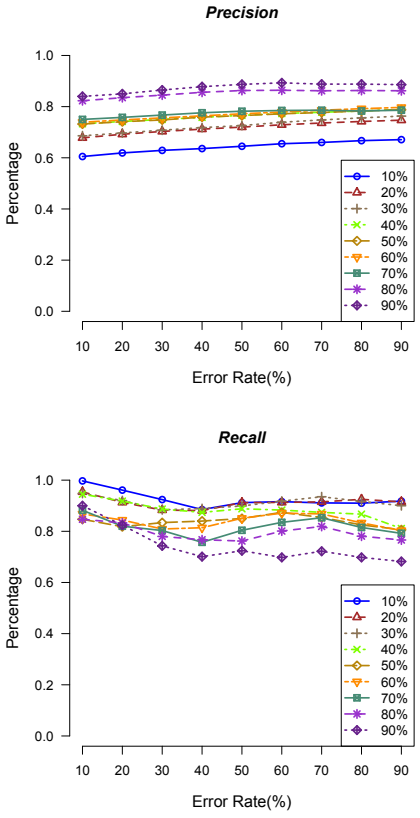

Fig. 4. Impact of thresholds on temporal correlation

reported a reading, another sensor is expected to report. The threshold is chosen by the cut percentage; for example, given the percentage 60%, we choose the value that is greater than 60% of the all values. Figure 4 presents the precision and recall of missing event detection with different threshold cuts, given that we split the whole data into half: the first half for training and the second half for testing. The results are expected: the higher the cut, the more selective the algorithm is at detecting missing events, thus leading to a higher precision and the lower the recall. However, the variety of cuts does not seem to affect the performance significantly. One of the reasons might be that the dataset only contains a very small number of sensors, and the correlation between these sensors is very explicit; that is, the cooking relevant activity tends to fire all the sensors in the kitchen, and there is less variety in the firing patterns.

C. Time Series Only

In the time series strategy we need to decide two parameters: embedding dimension m and time tolerance margin τ. For the sake of the space, we do not list the results on all the error rates, but only on selected ones, such as 10%, 50%, and 90%. Figure 5 presents the precision and recall over different time tolerance margins on different m values, given that we split the whole data into half and the first half for training and the second half for testing.

We see that the size of m has little effect on precision, but the smaller value of m consistently leads to higher recall. The reason for higher recall is that the smaller m allows us to capture more candidate sequences. Similarly, as the time tolerance value increases, recall improves. However, the overall performance does changes little, and is much lower than the performance of the temporal correlation. The reason might be that the dataset does not cover a long enough period (i.e., 28 days), and the time series pattern in the test data (i.e., the second half) is slightly different from the training data (i.e., the first half) in that the user often leaves home for a longer time (a couple of days). If a similar series pattern is not reflected in both the training and test data, then the prediction of the next firing time of the sensor is unreliable. Also, because we need the continuous series data, we cannot shuffle the data to generate better balanced training and test data. One of our future works is to evaluate our proposed approach on a dataset that covers a longer period.

D. Temporal Correlation and Time Series

Lastly, we want to compare the overall performance over these three different strategies. From the above two experiments, we can choose the parameters that best balance the precision and recall for the former two strategies; that is, here we choose the threshold cut for TC; and choose m as 2 and θ as 60 minutes. Then we compare the performance of the three of them in Figure 6. The TS strategy achieves the best precision and the worst recall, because it always infers a very small set of sensors. The TS + TC strategy performs best over the other two, but the improvement over TC is marginal. We also evaluate the impact of the amount of training data on the performance. We list the precision and recall for the split rates 10%, 50%, and 90%. It is clear that the more training data used, the better the performance in all these strategies.

V. Conclusion and Future Work

This paper presents our initial idea for detecting missing data from event-driven sensors. We explore two temporal solutions: temporal correlation and time series, both of which aim to characterise patterns of sensor usage in different human activities. We evaluate these solutions on a rather clean real-world dataset and obtain promising initial results. The temporal correlation works well on this dataset that contains a small number of sensors and the correlation between sensors is explicit. The time series does not work so well because the dataset is collected over a short period and the temporal pattern in the latter part of the dataset does not match well with the former. Our future plan is to evaluate the solutions on larger datasets over longer period and to also design strategies of combining these two solutions using, for example, evidential theory. Another extension is to include activity knowledge; that is, if we can infer the current activity or predict the future activity, we might derive what sensor events are expected.

References

[1] A. Ciampi, A. Appice, P. Guccione, and D. Malerba. Integrating trend clusters for spatio-temporal interpolation of missing sensor data. In *Proceedings of the 11th International Conference on Web and Wireless Geographical Information Systems*, W2GIS'12, pages 203–220, Berlin, Heidelberg, 2012. Springer-Verlag.

[2] L. Gruenwald, H. Chok, and M. Aboukhamis. Using data mining to estimate missing sensor data. In *Proceedings of ICDM Workshops 2007: The Seventh IEEE International Conference on Data Mining Workshops*, pages 207–212, Oct 2007.

[3] N. Jiang and L. Gruenwald. Estimating missing data in data streams. In R. Kotagiri, P. Krishna, M. Mohania, and E. Nantajeewarawat, editors, *Advances in Databases: Concepts, Systems and Applications*, volume 4443 of *Lecture Notes in Computer Science*, pages 981–987. Springer Berlin Heidelberg, 2007.

[4] H. Kantz and T. Schreiber. *Nonlinear time series analysis*, volume 7 of *Cambridge Nonlinear Science Series*. Cambridge University Press, Cambridge, 1997.

[5] T. L. M. Kasteren, G. Englebienne, and B. J. A. Krse. Human activity recognition from wireless sensor network data: Benchmark and software. In L. Chen, C. D. Nugent, J. Biswas, J. Hoey, and I. Khalil, editors, *Activity Recognition in Pervasive Intelligent Environments*, volume 4 of *Atlantis Ambient and Pervasive Intelligence*, pages 165–186. Atlantis Press, 2011.

[6] N. C. Krishnan and D. J. Cook. Activity recognition on streaming sensor data. *Pervasive and Mobile Computing*, 2012.

[7] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. Tinydb: An acquisitional query processing system for sensor networks. *ACM Trans. Database Syst.*, 30(1):122–173, Mar. 2005.

[8] M. A. Osborne, S. J. Roberts, A. Rogers, and N. R. Jennings. Real-time information processing of environmental sensor network data using bayesian gaussian processes. *ACM Trans. Sen. Netw.*, 9(1):1:1–1:32, Nov. 2012.

[9] L. Pan and J. Li. K-nearest neighbor based missing data estimation algorithm in wireless sensor networks. *Wireless Sensor Network*, 2:115 –122, 2010.

[10] P. P. Rodrigues and J. Gama. Online prediction of streaming sensor data. In *Proceeding of the 3rd International Workshop on Knowledge Discovery from Data Streams*, 2006.

[11] N. Vijayakumar and B. Plale. Prediction of missing events in sensor data streams using kalman filters. In *Proceeding of the 1st Int'l Workshop on Knowledge Discovery from Sensor Data, in conjunction with ACM KDDM' 07*, 2007.

[12] J. Ye, G. Stevenson, and S. Dobson. Usmart: an unsupervised semantic mining activity recognition technique. *ACM Transactions on Interactive Intelligent Systems*, 2014. To appear.

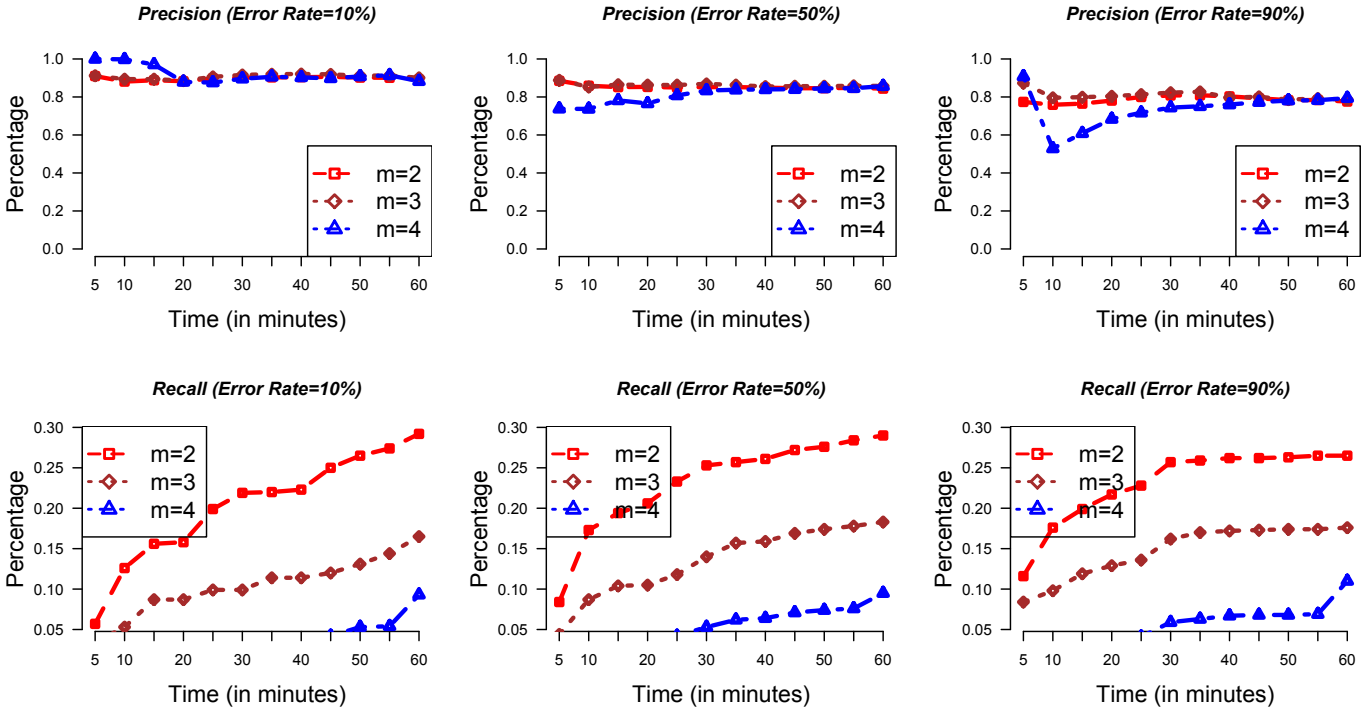

Fig. 5. Impact of embedding dimensions and time thresholds on detection precision and recall

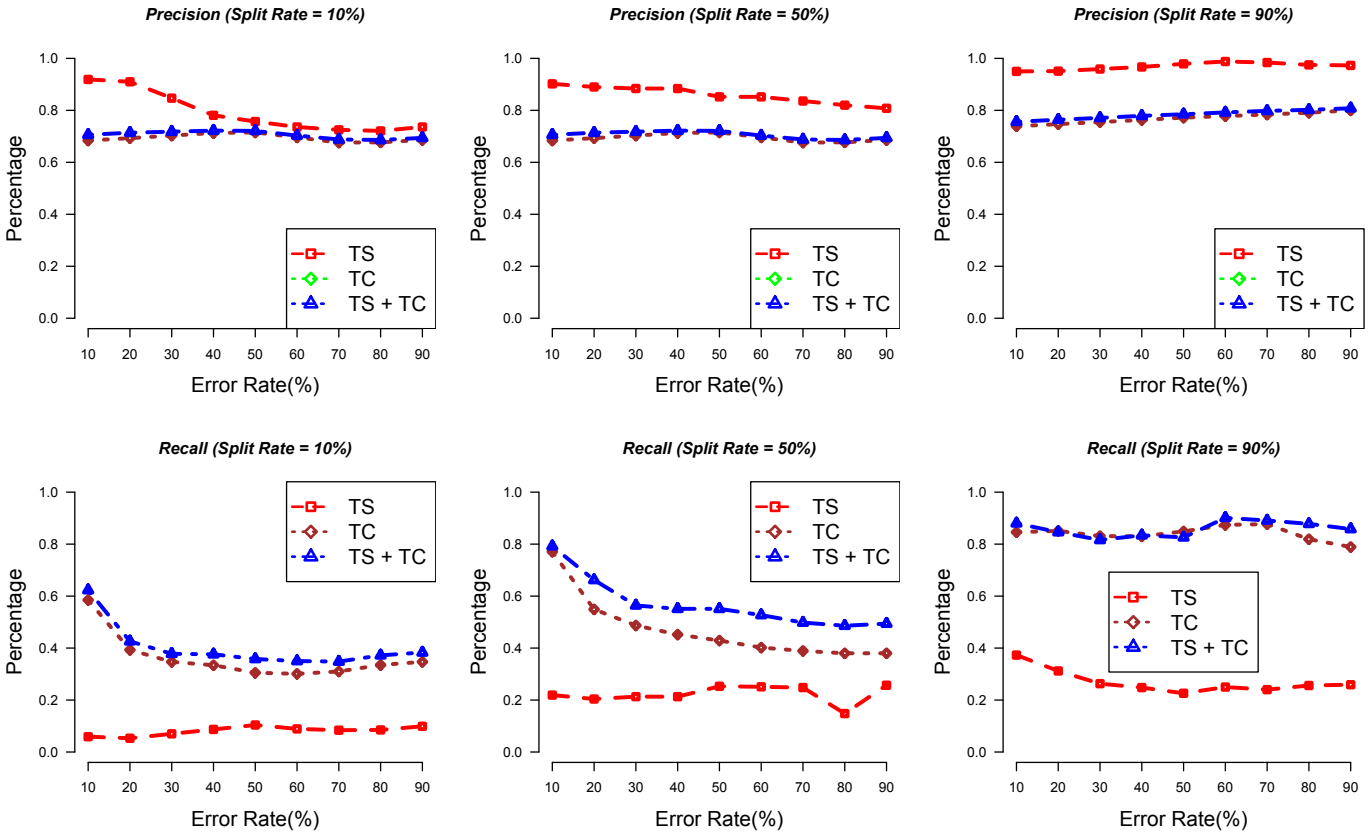

Fig. 6. Precision and recall of detecting missing sensor events