

A Data Imputation Model in Sensor Databases

Nan Jiang

The University of Oklahoma
School of Computer Science
Norman, OK, 73019, USA
nan_jiang@ou.edu

Abstract. Data missing is a common problem in database query processing, which can cause bias or lead to inefficient analyses, and this problem happens more often in sensor databases. The reasons include power outage at the sensor node, sensors time synchronization, occurrences of local interferences, unstable wireless network communication, etc. Therefore, in sensor database applications, there is a need for data imputation, especially for those applications in which the query response time is tight, and the accuracy of the query results is important. In this paper, we present a data imputation application based on association rule mining of closed frequent itemsets. They are subsets of all frequent patterns but provide complete and condensed information since they do not include redundant patterns. Experimental results compared with the existing techniques using real-life sensor data show that our proposed technique effectively imputes missing sensor data as well as achieves time and space efficiency.

1 Introduction

A wireless sensor network (WSN) is a wireless network consisting of spatially distributed autonomous devices using sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, at different locations [15]. Recent advances in sensor technology have made possible the development of relatively low cost and low-energy-consumption micro sensors which can be integrated in a wireless sensor network. These devices - Wireless Integrated Network Sensors (WINS) - will enable fundamental changes in applications spanning the home, office, clinic, factory, vehicle, metropolitan area, and the global environment [3].

Many researches have been conducted by different organizations regarding wireless sensor networks to address many different issues such as power awareness, security, routing protocol, heterogeneous sensor networking and so on, but few of them discuss how to impute the missing data when data is lost or corrupted. In this paper we present a data imputation model to impute data tuples in a sensor database using a data mining technique based on closed pattern mining for association rules. Its goal is to derive imputed values that are not only accurate but also timely. This is significant to many applications where exact data may not be necessary and certain approximate data is acceptable, such as traffic management, intrusion detection, and

network routing. This research also reduces the chance that real-time applications would miss their deadlines due to lack of data.

The data imputation model using association rule mining on stream data based on closed frequent itemsets (CARM) [11] discovers relationships between sensors and use them to compensate for missing and corrupted data. The derived association rules provide complete and non-redundant information, since closed pattern mining provides complete and condensed information and thus they do not include redundant patterns [21]. Experimental results compared with the existing techniques using real-life sensor data show that our proposed model effectively imputes missing sensor data as well as achieves time and space efficiency.

The remainder of this paper is organized as follows. We review the existing data imputation solutions in Section 2. We discuss the definitions of terms used in the paper in Section 3. In Section 4, we present our proposed online data imputation application based on the closed pattern mining. Section 5 depicts the performance evaluation of the proposed data imputation model comparing with the existing techniques using real-life traffic data. Finally, Section 6 concludes the paper.

2 Related Works

Many articles have been published to deal with the missing data problem, and a lot of software has been developed based on these methods. Some of the methods totally delete the missing data before analyzing them, like listwise and pairwise deletion [21] while some other methods focus on imputing the missing data based on the available information. The most popular statistical imputation methods include mean substitution, imputation by regression [4], hot deck imputation [9], cold deck imputation, expectation maximization (EM) [13], maximum likelihood [2, 12], multiple imputations [16, 18], and bayesian analysis [6].

Also, there are some mathematical approaches to perform the imputation, like SVDimpute, which is a singular value decomposition based method, called SVDimpute. This method is applied for imputing missing values in DNA microarrays. A DNA microarray is a matrix m rows of which are expression levels of genes and n columns are different experimental conditions. The missing values occur for diverse reasons, including insufficient resolution, image corruption, or simply due to dust or scratches on the slide [20].

But none of these approaches is specially suited for wireless sensor network environments, where streams of data constantly sent from the sensors to the servers, due to several reasons. First, how much old information should be based on to get the associated information for the missing data imputation? Using all of the old readings to perform the imputation is unreasonable, especially when using an iteration procedure until convergence to get the imputation like in the EM algorithm. On the other hand, using only the previous round of sensor readings to perform the imputation is also not a good choice since data streams often have a changing data distribution. Some of the statistical methods use all of the available data points in a database to construct the best possible results, like the Maximum Likelihood. In the wireless sensor networks, the missing sensor data may or may not be related to all of the available information, thus using all of the available information to process the

result is not an optimal choice and would use more time and memory space than necessary when it comes to the implementation stage.

Second, which information should be used to perform the missing data imputation? In a wireless sensor network, data is collected within certain scopes and reported to the servers during a certain period of time. Different sensors have different readings at different time periods, and the current readings of one sensor may relate not only to its previous readings, but also to other sensors' previous or current readings. Therefore, replacement of missing values with randomly selected values present in a pool of similar complete cases or by a value which is independent of the data set like in the hot/cold deck imputation is difficult to implement. This is because even though we may get the complete set of information of a certain wireless sensor network, it is not easy to decide which information is similar to the current round of missing sensor's information. In other words, it is hard to draw the pool for a certain sensor's certain round of readings when the application needs to perform the data imputation.

Third, the missing data may or may not miss at random (MAR), while most of the statistical techniques, such as maximum likelihood [2, 12] and multiple imputations [16, 18], are based on the MAR assumption. According to the definition in [12], Data on Y are Missing At Random (MAR) if the probability that Y is missing does not depend on the value of Y after controlling other observed variables X . For example, we are modeling weight (Y) as a function of gender (X). One gender may be less likely to disclose its weight, that is, the probability that Y is missing depends only on the value of X . Such data are MAR.

As there are more and more stream data applications emerge, proper data estimation algorithms for stream data are needed. In the prediction model of both TinyDB and BBQ, multivariate data modeling techniques are used [6]. Such models can be learned from historical data using standard algorithms. For a specific model, training data needs to be collected for some period of time before predicting values. These models need to be updated over time to reflect the most recent changes. Choosing the best model for the given query workload and environment is an important issue in this case. While our proposed technique catches the relationship among sensors using association rule mining, this can be applied to a broad applications without model updating.

In [14], the authors propose using pattern discovery in multiple time-series to estimate missing data, but it's not well suited for sensor networks, where the relationships between sensors decided not only by the time trends, but also some other factors, like locations and so on.

In [8], the authors proposed the WARM (Window Association Rule Mining) algorithm for imputing missing sensor data. WARM uses association rule mining to identify sensors that report the same data for a number of times in a sliding window, called related sensors, and then imputes the missing data from a sensor by using the data reported by its related sensors. WARM has been reported to perform better than the average approach where the average value reported by all sensors in the window is used for imputation. However, there exist some limitations in WARM. First, it is based on 2-frequent itemsets association rule mining, which means it can discover relationships only between two sensors and ignore the cases where missing values are related with multiple sensors. Second, it finds those relationships only when both

sensors report the same value and ignores the cases where missing values can be imputed by the relationships between sensors that report different values.

In view of the above challenges, in this paper we present a data imputation model using CARM (Closed Itemsets based Association Rule Mining) [11], which can derive the most recent association rules between sensors based on the closed itemsets in the current sliding window. The definition of closed itemsets is given in Section 3.

3 Definitions

In this section, we describe the notations and definitions that are used throughout this paper.

Let $D = \{d_1, d_2, \dots, d_n\}$ be a set of n item ids, and $V = \{v_1, v_2, \dots, v_m\}$ be a set of m item values. An item I is a combination of D and V , denoted as $I = D.V$. For example, $d_n.v_m$ means that an item with id d_n has the value v_m . A subset $X \subseteq I$ is called an itemset. A k -subset is called a k -itemset. Each transaction t is a set of items in I . Given a set of transactions T , the support of an itemset X is the percentage of transactions that contain X . A frequent itemset is an itemset the support of which is above or equal to a user-defined support threshold.

Let T and X be subsets of all the transactions and items appearing in a data stream D , respectively. The concept of closed itemset is based on the two following functions, f and g : $f(T) = \{i \in I \mid \forall t \in T, i \in t\}$ and $g(X) = \{t \in D \mid \forall i \in X, i \in t\}$. Function f returns the set of itemsets included in all the transactions belonging to T , while function g returns the set of transactions containing a given itemset X . An itemset X is said to be closed if and only if $C(X) = f(g(X)) = f \bullet g(X) = X$ where the composite function $C = f \bullet g$ is called Galois operator or closure operator [19].

For example, let $I = \{A, B, C, D\}$ be a set of 4 items, and $T = \{CD, AB, ABC, ABC\}$ be a set of transactions in data streams, then the closed itemsets and their support counts are $\{(C, 3), (AB, 3), (CD, 1), (ABC, 2)\}$. Each of the closed itemsets X satisfies $C(X) = f(g(X)) = f \bullet g(X) = X$. Take AB as an example, $g(AB) = \{AB, ABC, ABC\}$, $f \bullet g(AB) = AB$, so $C(AB) = f(g(AB)) = f \bullet g(AB) = AB$. Closed frequent itemsets are those closed itemsets that have support equal to or greater than the user-defined minimum support. If the user defined the absolute support to be 2, then the closed frequent itemsets are $\{(C, 3), (AB, 3), (ABC, 2)\}$. The frequent itemsets are $\{(A, 3), (B, 3), (C, 3), (AB, 3), (AC, 2), (BC, 2), (ABC, 2)\}$, from which we can see that closed frequent itemsets are smaller subsets of frequent itemsets and contain all itemsets and support information in the frequent itemsets.

From the above discussion, we can see that a closed itemset X is an itemset the closure $C(X)$ of which is equal to itself ($C(X) = X$). The closure checking is to check the closure of an itemset X to see whether or not it is equal to itself, ie., whether or not it is a closed itemset.

An association rule $X \rightarrow Y$ (s, c) is said to hold if both s and c are above or equal to a user-specified minimum support and confidence, respectively, where X and Y are sensor readings from different sensors, s is the percentage of records that contain both X and Y in the data stream, called support of the rule, and c is the percentage of records containing X that also contain Y , called the confidence of the rule. The task of

mining association rules then is to find all the association rules among the sensors which satisfy both the user-specified minimum support and minimum confidence.

4 Data Imputation Model Based on Closed Pattern Mining

The data imputation model uses association rule mining [1] on stream data to compensate for missing and corrupted data. An association rule is an implication of the form $X \Rightarrow Y (s, c)$, where X and Y are frequent itemsets in a database and $X \cap Y = \emptyset$, s is the percentage of records that contain both X and Y in the database, called support of the rule, and c is the percentage of records containing X that also contain Y , called the confidence of the rule. An association rule is said to hold if both s and c are above or equal to a user-specified minimum support and confidence. An itemset is frequent if its support is above or equal to a user-defined support threshold. A k -frequent itemset is a frequent itemset with k items. Our goal is to find the relationships between sensors, and later use these relationships (or association rules) to impute the values of missing sensor readings.

When a transaction arrives or leaves the current data stream sliding window, the CFI-Stream algorithm [10] checks each itemset in the transaction on the fly and updates the associated closed itemsets' supports. The current closed itemsets are maintained and updated in real time in the DIU tree. The closed frequent itemsets can be output at any time at users' specified thresholds by browsing the DIU tree.

A lexicographical ordered direct update tree is used to maintain the current closed itemsets. Each node in the DIU tree represents a closed itemset. There are k levels in the DIU tree, where each level i stores the closed i -itemsets. The parameter k is the maximum length of the current closed itemsets. Each node in the DIU tree stores a closed itemset, its current support information, and the links to its immediate parent and children nodes. Figure 1 illustrates the DIU tree after the first four transactions arrive. The support of each node is labeled in the upper right corner of the node itself. The figure shows that currently there are 4 closed itemsets, C , AB , CD , and ABC , in the DIU tree, and their associated supports are 3, 3, 1, and 2.

We assume in this paper that all current closed itemsets are already derived, and based on these closed itemsets, we generate association rules for data imputation. Please refer to [10] for the detailed discussion of the update of the DIU tree and the closure checking procedure for addition and deletion operations.

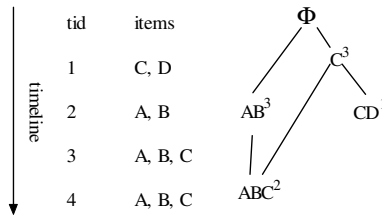


Fig. 1. The lexicographical ordered direct update tree

Based on the current closed itemsets in the DIU tree, instead of generating all possible association rules, we generate the rules that have strong relationships with the current round of sensor readings where one or more readings are missing. We achieve this through browsing the DIU tree, which stores all of the closed itemsets. Based on the users' specified support and confidence thresholds, we find out rules through paths (links) of closed itemsets that suit the users' needs, i.e., satisfy the users' specified support and confidence thresholds. For example, if the user's specified support and confidence threshold is 0.3 and 0.6 respectively, it means that we find out all the closed itemsets whose appearances are equal or greater than 30% of all transaction sets, and we find out all the association rules that related with the specified closed itemset whose possibilities to happen are equal or greater than 60%. The mining process is online and incremental, which is especially beneficial when users have different specified thresholds in their online queries. Please refer to [11] for a detailed discussion of the procedures of the CARM algorithm.

The data imputation model provides the following functionalities as shown in figure 2. It first preprocesses the data received from the input module, then judges if it contains missing or corrupted value. If yes, it performs data imputation, outputs the imputed value and stores it in the database; otherwise, it stores the value in the database directly. The users can thus query the database and get the query results in real time.

Below is an example of data imputation using closed itemsets based association rule mining. Assume a wireless sensor network consists of four sensors S_1 , S_2 , S_3 , and S_4 that send their readings to a server in a certain time interval. Each sensor detects the temperature of a room during a certain period of time. The different values of the observed temperature from each sensor are represented as follows: $S_1.value_1$, $S_2.value_2$, $S_3.value_3$, $S_4.value_4$ and so on, where the value of each sensor may or may not be the same. The sensor node sends the generated tuple to the server using its radio unit.

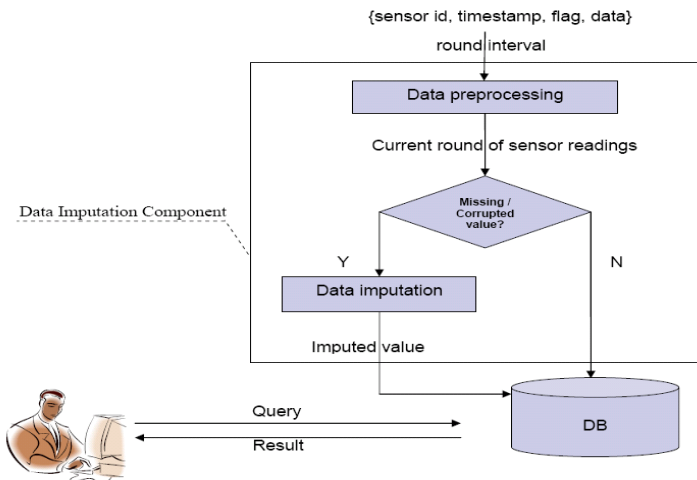


Fig. 2. Data imputation model

Assume the sensors have reported the following values as shown in figure 3 for the last 4 rounds of sensor readings, where round number 1 is the oldest round and round number 4 is the newest one. Assume minimum support = 50%, and minimum confidence = 50%. From the figure we can see all closed frequent itemsets from the current readings are $\{S_1.70, S_3.72, S_4.60\} = 2$, $\{S_1.70, S_4.60\} = 3$, $\{S_2.60, S_4.60\} = 2$, and $\{S_4.60\} = 4$. Based on the non-redundant association rules derived from the closed frequent itemsets, we can derive $\{S_1.70, S_4.60\} \rightarrow S_3.72$, support = 1/2, confidence = 2/3 and $S_4.60 \rightarrow S_2.60$, support = 1/2, confidence = 1/2, we can impute the missing value $S_3.72$ and $S_4.60$. Compared with WARM, CARM can find out the relationship between multiple sensors instead of pairs of sensors; it can also derive the relationship among sensors with different values instead of only same value $S_4.60 \rightarrow S_2.60$, therefore it increases the number of missing values that can be imputed.

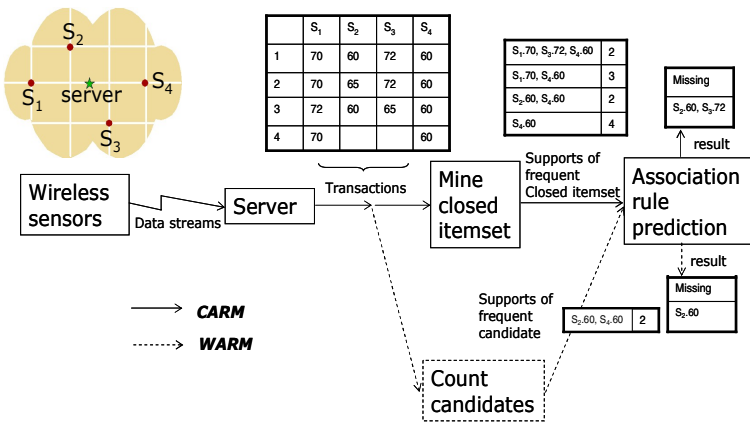


Fig. 3. An example of data imputation using closed itemsets mining

5 Experimental Evaluations

The performance of our proposed data imputation model is studied by means of simulation. Several different simulation experiments are conducted in order to evaluate the proposed technique and compare it with four existing statistical techniques: the Average Window Size (AWS) approach, the Simple Linear Regression (SLR) approach, the Curve Regression (CE) approach, and the Multiple Regression (MR) approach, and with the WARM approach, the current state-of-the-art data imputation algorithm in sensor database [8].

The simulation model consists of 108 sensor nodes. All sensor nodes report to a single server. The sensors are deployed on city streets, collect and store the number of the vehicles detected for a given time interval. The actual vehicle counts taken as sensor readings that are used as input for our simulation experiments are traffic data provided by [1]. The data was collected in year 2000 at various locations throughout the city of Austin, Texas. The data represents the current location, the time interval, and the number of vehicles detected during this interval. From this set we generated

four different input data sets corresponding to the different numbers of the possible sensor states used for the simulation experiments.

The evaluation of the achieved accuracy of an imputation of the missing values is done by using the average Root Mean Square Error (RMSE):

$$RMSE = \frac{1}{numStates} \sqrt{\frac{\sum_{i=1}^{#imputations} (Xa_i - Xe_i)^2}{#imputations}}$$

where Xa_i and Xe_i are the actual value and the imputed value, respectively; #imputations is the number of imputations performed in a simulation run; and numStates is the number of subsets, in which the actual readings are distributed. The expression $\sqrt{\frac{\sum_{i=1}^{#imputation\ s} (Xa_i - Xe_i)^2}{#imputation\ s}}$ represents the standard error and is an imputation of the

standard deviation under the assumption that the errors in the imputed values (i.e. $Xa_i - Xe_i$) are normally distributed. Thus, the RMSE allows the construction of confidence intervals describing the performance of different candidate missing value estimators. The smaller the RMSE (the standard deviation), the better the estimator. The calculated RMSE for each different set of input data (e.g. Set10 means that the sensor readings are split into 10 subsets) is divided by the number of subsets and the result is the average standard deviation in each case. This is done to keep the measure comparable across experiments.

From figure 4 we can see that CARM gives the best result of the above approaches regarding to the accuracy, follows by the WARM and AWS approaches. The regression approaches performs no better than WARM, CARM and AWS approaches, the main reason might be that it only considers the relationship between the neighbor nodes, while CARM and WARM find out all of the relationship between the existing sensors. Also from figure 4, we can see that the proposed CARM approach provides better imputation accuracy than the WARM approach does. This is because CARM performs the imputation based on the association rules derived by a compact and complete set of information, while WARM performs the imputation based on the association rules derived only by 2-frequent itemsets in the current sliding window.

Figure 5 illustrates the Total Main Memory Access Time per round (TMMAT) in milliseconds of AWS, SLR, CE, MR, WARM and CARM approaches. The TMMAT is defined as the time for performing all main memory accesses required for updating

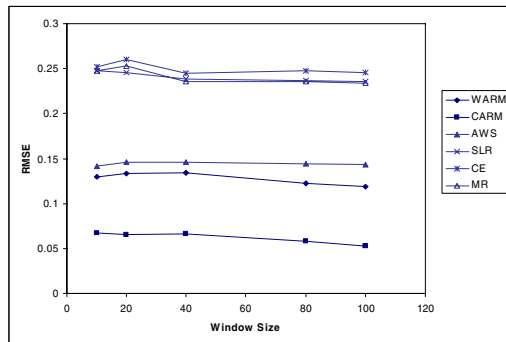


Fig. 4. RMSE for AWS, SLR, CE, MR, WARM and CARM approaches

the associated data structures and impute missing values per round of sensor readings. The experimental results show that in terms of TMMAT, the proposed CARM approach is outperformed by all other four statistical approaches, but it's still very fast comparing with the resend cases. The CARM approach is faster than the WARM technique. As shown in this figure, the TMMAT of WARM increases slightly when the window size increases since the information in WARM stores in the cube data structures, and the time needs to process this information increases when the size of the cube increases. For the CARM approach, the TMMAT first increases as the number of transactions increases since the number of closed itemsets that newly discovered increases; however, the average processing time decreases after the number of newly discovered closed itemsets reaches a threshold. This is because the number of closed itemsets which exist in the DIU tree increases, and they do not need to be processed; only their supports need to be updated incrementally.

Figure 6 illustrates Memory Space (in Kbytes) of AWS, SLR, CE, MR, WARM and CARM approaches. The experimental results show that in terms of Memory Space, the proposed CARM approach is outperformed by all other four statistical approaches, but it's still requires far less than the main memory provided in a contemporary computer. The results of the simulation experiments show that for 108 sensors, using WARM, the needed memory space is much higher than that using

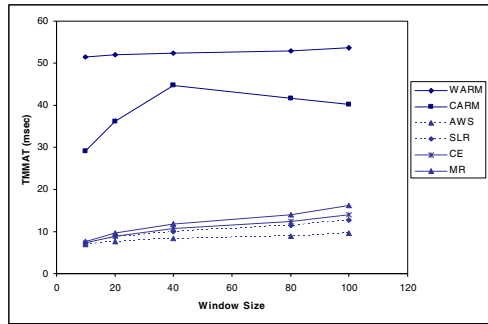


Fig. 5. TMMAT for AWS, SLR, CE, MR, WARM and CARM approaches

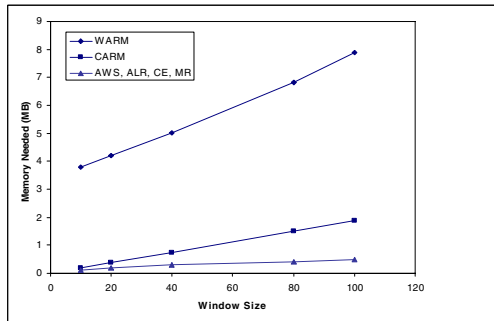


Fig. 6. Memory Needed for AWS, SLR, CE, MR, WARM and CARM approaches

CARM. This is because the DIU tree data structure uses much less memory space than the cube data structures, and it only stores the condensed closed itemsets information.

There is a possibility that the imputation algorithm alone will not be able to impute a missing value. The reasons for that are the following: no association rules between the sensor with the missing value (MS) and other sensors can be derived based on the current information. The percentage of cases in which a missing value cannot be imputed by the imputation algorithm alone (*PCE*) is computed for each simulation run using the following formula:
$$PCE = \frac{\#casesValue\ Cannot\ Be\ Imputed}{totalNumber\ of\ Attempts\ To\ Impute} * 100\%$$
 where

#casesValueCannotBeImputed is the number of cases in which a missing value cannot be imputed using the imputation algorithm alone, and *totalNumberOfAttemptsToImpute* is the total number of attempts to impute a missing value in a given simulation run. From figure 7 we can see the CARM algorithm greatly reduces the number of cases that can not be imputed directly by the association rules derived. This is because, compared with WARM, it considers the relationships between multiple sensors and enables us to find more associations between multiple sensors even when they have different values.

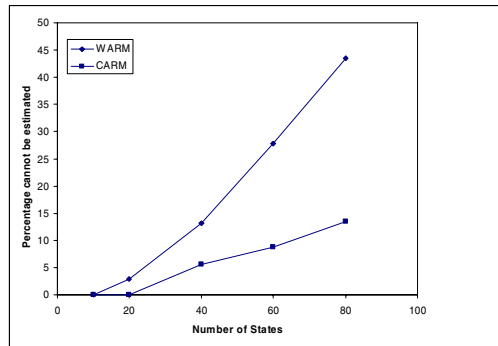


Fig. 7. PCE for WARM and CARM approaches

6 Conclusions

In this paper we present a novel data imputation model in sensor network databases based on closed pattern mining. This model integrates an incremental method to perform data imputation from the derived association rules based on closed pattern mining. Our performance study shows that it is able to impute missing sensor data online with both time and space efficiency, greatly improves the imputation accuracy and reduces the number of cases that cannot be imputed.

References

1. Austin Freeway ITS Data Archive: (accessed 2003), <http://austindata.tamu.edu/default.asp>
2. Allison, P.D.: Missing data. Thousand Oaks, CA Sage (2002)

3. Asada, G., Dong, M., Lin, T.S., Newberg, F., Pottie, G., Kaiser, W.J., Marcy, H.O.: Wireless Integrated Network Sensors: Low Power Systems on a Chip. In: European Solid State Circuits Conference (1998)
4. Cool, A.L.: A review of methods for dealing with missing data. Paper presented at the Annual Meeting of the Southwest Educational Research Association, Dallas, TX (2000)
5. Laird Dempster, N., Rubin, D.: Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society* (1977)
6. Deshpande, A., Guestrin, C., Madden, S., Hellerstein, J., Hong, W.: Model-driven data acquisition in sensor networks. In: VLDB (2004)
7. Carlin Gelman, J., Stern, H., Rubin, D.: Bayesian Data Analysis. Chapman & Hall, Sydney (1995)
8. Halatchev, M., Gruenwald, L.: Estimating Missing Values in Related Sensor Data Streams. Int'l. Conf. on Management of Data (2005)
9. Iannacchione, V.G.: Weighted sequential hot deck imputation macros. In: Proceedings of the SAS Users Group International Conference (1982)
10. Jiang, N., Gruenwald, L.: CFI-Stream: Mining Closed Frequent Itemsets in Data Streams. In: ACM SIGKDD international conference on knowledge discovery and data mining (2006)
11. Jiang, N., Gruenwald, L.: Estimating Missing Data in Data Streams. In: 12th International Conference on Database Systems for Advanced Applications (2007)
12. Little, R.J.A., Rubin, D.B.: Statistical analysis with missing data. John Wiley and Sons, New York (1987)
13. McLachlan, G., Thriyambakam, K.: The EM Algorithm and Extensions. John Wiley & Sons, New York (1997)
14. Papadimitriou, S., Sun, J., Faloutsos, C.: Streaming Pattern Discovery in Multiple Time-Series. In: VLDB (2005)
15. Romer, K., Mattern, F.: The Design Space of Wireless Sensor Networks, IEEE Wireless Communications (2004)
16. Rubin, D.: Multiple Imputations for Nonresponse in Surveys. John Wiley & Sons, New York (1987)
17. Rubin, D.: Multiple Imputations after 18 Years. *Journal of the American Statistical Association* (1996)
18. Shafer, J.: Model-Based Imputations of Census Short-Form Items. In: Proceedings of the Annual Research Conference, Bureau of the Census, Washington, DC (1995)
19. Taouil, R., Pasquier, N., Bastide, Y., Lakhal, L.: Mining Bases for Association Rules Using Closed Sets. In: International Conference on Data Engineering (2000)
20. Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., Altman, R.B.: Missing Value Estimation Methods for DNA Microarrays. In: *Bioinformatics* (2001)
21. Wilkinson & The APA Task Force on Statistical Inference (1999)
22. Zaki, M.J.: Generating non-redundant association rules. In: ACM SIGKDD international conference on knowledge discovery and data mining (2000)