Estimating Missing Values of WSN using modified Frequent itemsets mining and NN search

Mengzhenfeng Luo[1], Fan Wang[1] and Xiaopeng Hu[1]

[1]School of Computer Science and Technology, Dalian University of Technology, Dalian City, China

ryuukazenomai@foxmail.com, wangfan@dlut.edu.cn, xphu@dlut.edu.cn

Keywords: Wireless Sensor Network, Missing Data Imputation, Modified Frequent Itemsets Mining, Nearest Neighbor Estimation

Abstract. Due to causes such as unreliability of the transport protocol, energy exhaustion and noise disturbance etc in wireless sensor network, the uploaded data on the sensor node usually tend to be incomplete, which brings about a series of inconvenience for analysis and operation in the subsequent. Therefore, it is necessary for us to make compensation for the missing data. In this paper, we put forward one kind of method of combing with modified Frequent Itemsets mining and NN search(FINN) to make estimation for the missing data in the wireless sensor network and use estimated data to replace missing value. Because the final operation only uses similar data, it is unnecessary to use all the data, so it can reduce unnecessary error and enhance precision of estimated value.

1. Introduction

In recent years, with the gradual progress in embedded technology, sensor technology and low-consumption wireless communication technology, people are increasingly paying high attention to wireless sensor network technology. They are usually used in important fields such as environment monitoring, space exploration, military monitoring etc. In the sensor network, it conveys information among sensor nodes and transmits collected data to base station for analysis and procession. Because wireless sensor network is not using reliable transmission protocol, and couples with problems such as energy exhaustion of sensor, signal failure and signal none-synchronization etc in the network, the finally collected data may be missed. Incomplete data will produce influence on analysis and algorithm of plenty of data, which will even produce wrong result.

Changing unreliable transmission protocol as reliable transmission protocol in the wireless sensor network may solve the similar problems, but as for wireless sensor network of using battery, its cost is very higher, because reliable transmission requires overtime retransmission and back to ack, which will certainly increase energy consumption. Moreover, reliable transmission protocol may cause bigger delay for network.

We can also use statistics method to make estimation for missing data and use estimated value to replace missing value. This may be one better method, because incomplete data may cause wrong analysis result, so we can use the most possible value to make up for the missing data, afterwards, we can normally use subsequent analysis algorithm. Because there is usually certain time and space-time association among sensor nodes, we usually use it to make up for the missing data by using this kind of space-time association

In this paper, we put forward one kind of prior modified Frequent Item sets, and then find nearest neighbor method to make compensation and estimation for the missing data. This kind of method combines with the advantages of Frequent Itemsets mining and nearest neighbor search, which makes estimated value much more precise, meanwhile, it meets requirement of wireless sensor network which needs timely updating.

The remaining parts of this paper are as follows:

The second chapter introduces some conditions of related work, the third chapter describes main content of algorithm, the fourth chapter introduces test result and evaluation, the fifth chapter makes

summary of this paper.

2. Related Work

When the data is incomplete, there are many strategies can solve it. For example, one usual strategy is to abandon the incomplete data and only use complete data to make analysis and procession in later stage, or it directly uses the collected data of the last time to replace missing data of this time, but this may produce bigger error. There are also some strategies of using statistics to make estimation for the missing data. There are many similar methods like this, and there are many papers have made discussion on it, such as regression[1], hot deck imputation[2,3], expectation maximization[4], maximum likelihood[5,6], Bayesian analysis[7] etc, but these methods are not applicable to wireless sensor network. The reasons are as follows: firstly, the node data in wireless sensor network will update timely, its data is one kind of data stream, secondly, and the missing data may not produce association for all the data, using all the data to make estimation on the missing data may waste too much time. Moreover, according to [10], the missing data can be divided into MCAR, MAR and MNAR, the above-mentioned methods usually require data to be MAR, while data in wireless sensor network may not be MAR.

In [8], it puts forward one kind of method of using association rule mining to estimate missing data of sensor, which is WARM. In the WARM, it sets one sliding window to store the uploaded data in recently in the w round sensors and uses it to estimate value of missing data. WARM is trying to find sensors with the same reported data in the window data, which is association rule; it regards them as related sensors. It uses data value of related sensors to estimate missing data value. The association rule in WARM is that similar to association rule of $X \to Y|s$, of which, X and Y are the subsets of sensor set of size one, and $X \cap Y = \phi$, s is a sensor state out of all possible sensor states. Association rule meets certain support and confidence, support is the percentage when X and Y appear S state at the same time in the sliding window, while confidence X is the percentage when its state is S at the same time Y is also s.

WARM mainly uses 3 kinds of data structures to store data in the window: the Buffer, the Cube, and the Counter, and it also uses these 3 kinds of algorithms to work on this kind of data model: checkBuffer(), update(), and estimateValue(). Firstly, checkBuffer() is used to check whether data of the Buffer is missing or not in this round, if it has no missing, it will use update() to update the Cube and the Counter, otherwise it will use estimteValue() to estimate missing value and store it in the Buffer and then use update() algorithm.

Because WARM only needs to find 1- and 2- frequent itemsets, so it can save plenty of time compared to mining all the association rules. But the association rule found by WARM is established on the basis of sensor uploading the same data, while it neglects the possible association among data difference or similar sensors, which will affect estimation precision and even cause failure itself.

[9] puts forward one kind of method to find k-nearest Neighbor to induce the missing data according to similarity, which is regarded as MSD. Firstly, it divides data into Data Unit, one Data Unit corresponds to p observation values with one attribute in one sensor node, and its dimension is P. If there is missing data, MSD will find k Data Unit with the smallest weight distance with missing data, which is k-nearest Neighbor. It will use k Data Units to separately make linear regression operation with the Data Unit having missing data, and then gets the estimated value.

3. Improved Algorithm-FINN

According to the already existed algorithms and their advantages, we put forward one kind of improved algorithm and regard it as FINN. According to the spatial correlation of sensor data, we are trying to find nodes that may be similar to nodes with missing data. While according to the temporal correlation of sensor data, we use linear regression operation to get the final estimated value. This kind of algorithm is to find similar nodes mainly by mining similar 2-frequent itemsets,

and it uses similar value to make estimation for the missing data. Considering there may be failure on mining, we use nearest neighbor method to estimate the missing value. At the same time, we also maintain 2 groups of data structures, and they are respectively used to find frequent itemsets and nearest neighbor. Combing with 2 kinds of methods will not increase time complexity of algorithm.

3.1Some Definitions

Suppose $I=\{i_1,i_2,....i_n\}$ is one set of sensors, sliding window is w round, D is the set of the latest data of w round uploaded by the sensors.

Definition 1: similar threshold e

If sensor data v_{i_j} and v_{i_k} meets $\left|v_{i_j}-v_{i_k}\right|<e$ in the same round data, then we regard v_{i_j} and v_{i_k} are similar.

Definition 2: similarity s

Under condition that v_{i_j} and v_{i_k} are similar, if they have relatively higher similarity, then value of $\left|v_{i_j}-v_{i_k}\right|$ is much more smaller; therefore, we define similarity of 2 values as follows:

$$s=\begin{cases}1-\dfrac{\left|v_{i_j}-v_{i_k}\right|}{e}, & \left|v_{i_j}-v_{i_k}\right|<e \\ 0, & \left|v_{i_j}-v_{i_k}\right|\geq e\end{cases} \qquad (1)$$

When 2 data are completely the same, its similarity is 1. When they are not similar, its similarity is 0.

Definition 3: similarity support $sSup$

$$sSup_{<i_j,i_k>}=\frac{\sum_{t=1}^{w}s_{<i_j,i_k,t>}}{w}\times 100\% \qquad (2)$$

Of which, $s_{<i_j,i_k,t>}$ indicates the similarity of sensor i_j and sensor i_k in the t round. Because we are not using the original support definition, we regard it as similarity support $sSup$.

Definition 4: similarity 2-frequent itemsets

If similarity support $sSup_{<i_j,i_k>}$ of itemsets $X=\{i_j,i_k\}$ is bigger than the minimum support $minSup$ defined by users, then we regard X as similar 2-frequent itemsets.

3.2 Data Model

WARM use one data structure called the Cube to store whether states of 2 sensors are the same or not. While we use two structures similar to the Cube, one is used to store similarity of sensors; the other one is used to store difference square between sensor data, they are used to calculate the Euclidean distance. Of which, Cube [i_j] [i_k].slice[t] indicates the similarity of i_j and i_k of the t round data in the first cube, while it indicates square of data difference of i_j and i_k of t round data in the other cube.

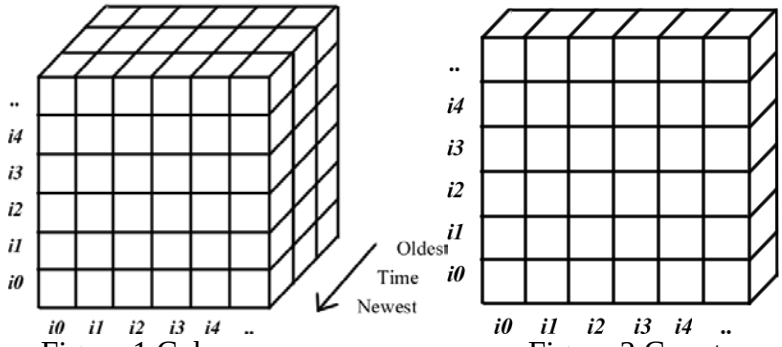

Figure 1 Cube Figure 2 Counter

We use 2 structures similar to the Counter to respectively store sum of similarity and the

Euclidean distance. Our counter structure size is only n^2. Of which, Counter $[i_j]$ $[i_k]$ indicates the similarity sum of $[i_j]$ and $[i_k]$ of *w* round data in the first counter, while it indicates Euclidean distance square of $[i_j]$ and $[i_k]$ in the second counter.

We use the same data structure as the Buffer of WARM to store the latest round of data. Because after data is used to update cube and counter, it is unnecessary to be stored in the buffer, so the buffer size is only n.

Algorithm Description： When the newest round of data is uploaded, we store data in the buffer and check whether data is missing or not. If there is no missing data, we use data in buffer to update 2 cubes and 2 counters. If there is missing data, for each sensor having missing data, we try to find similar 2-frequent itemsets include itself but excludes other sensors with missing data, which means we try to find items that meet the minimum similarity support in the first counter. If we can find it, then we use every sensor that can be found to make linear regression operation with the missing data sensor, then we get the estimated value by their weighted sum. If we can not find it, then we find the sensor with the smallest distance in the second counter, which is the nearest Neighbor, and then we carry out linear regression operation with the sensor having missing data. The description of algorithm process is as follows:

Table 1 The Pseudo Code of FINN

The Pseudo Code of FINN

```
mainProc()
{
   //check if there is missing data
   for(int i = 0; i < numberOfSensors; i++)
      if(Buffer[i] == -1)
         Buffer[i] = estimateValue(i);
   //insert new nodes at the front of two Cube
   for(int i = 0; i < numberOfSensors; i++)
      for(int j = 0; j < numberOfSensors; j++)
      {
         if (abs(Buffer[i] - Buffer[j]) < e)
            Cube0[i][j].slice[0] = 1 - abs(Buffer[i] - Buffer[j]) / e;
         else
            Cube0[i][j].slice[0] = 0;
         Cube1[i][j].slice[0] = (Buffer[i] - Buffer[j]) * (Buffer[i] - Buffer[j]);
         //update two Counter
         Counter0[i][j]=Counter0[i][j]-Cube0[i][j].slice[WindowSize]+Cube0[i][j].slice[0];
         Counter1[i][j]=Counter1[i][j]-Cube1[i][j].slice[WindowSize]+Cube1[i][j].slice[0];
      }
   discard the oldest nodes at the back of two Cube;
}
estimateValue(missingSensorID)
{
   int t = 0;
   //search similar 2-frequent itemsets
   for(int i = 0; i < numberOfSensors; i++)
      if( (Buffer[i] != -1) && (Counter0[i][missingSensorID] > minSup) )
      {
         weight[t] = Counter0[i][missingSensorID];
         y[t++] = linear regression on similar data(missingSensor,Sensor_i);
      }
   //get the final values
   x =  Σ(weight[f]); //f=1, 2, …, t
   y =  Σ(y[f]*weight[f]/x); //f=1, 2, …, t
   if (t > 0) return y;
   //if failed, use the nearest neighbor to estimate missing data
   find the sensor that has minimum Counter1[Sensor_tofind][missingSensorID]
   return linear regression(missingSensor,Sensor_found);
}
```

Experimental Evaluation

We carry out test in one computer with CPU: Intel core i3 1.80GHz, memory: 4G, operation system: windows 7 of 64-bit. We implemented the algorithm by using C++ in Visual Studio 2010. As for test data, we use air temperature data of Australian Institute of Marine Science and simulate 5 sensor nodes. We compare the test result of algorithm with that of WARM and MSD:

We use PEF (Percentage of Estimating Failure) to evaluate the success rate of estimation; PEF uses the following formula to make calculation:

$$PEF = \frac{NumberOfFailedEstimating}{TotalNumberOfEstimating} \times 100\% \qquad (3)$$

Number of Failed Estimating indicates the times of failed estimation; Total Number of Estimating indicates the total times need to carry out estimation.

Table 2 PEF for WARM, MSD, and FINN

Window Size	PEF		
	WARM	MSD	FINN
5	23	0	0
10	69	0	0
20	100	0	0
80	100	0	0
100	100	0	0

From table 2, we can see that the failure rate of WARM will increase with increase of window; this is mainly because mining association rule of WARM is established on the basis of the same sensor data. While MSD and FINN will not have failure, this is because MSD will finally use linear regression to calculate estimated value, while even if FINN is failed in finding frequent itemsets, it will find nearest Neighbor and use linear regression to calculate the estimated value.

We use the average Root Mean Square Error (RMSE) to evaluate precision of estimation. RMSE can well reflect precision of estimated value, the smaller of RMSE, which represents that the estimated value of algorithm is much more precise.

Table 3 RMSE for WARM, MSD, and FINN

Window Size	RMSE		
	WARM	MSD	FINN
5	0.079	0.035	0.043
10	0.024	0.032	0.029
20	Not Defined	0.026	0.018
80	Not Defined	0.011	0.007
100	Not Defined	0.010	0.006

From table 3 we can see that FINN has relatively higher precision, and with increase of window, value estimated by FINN is much more precise, this is because we only use the similar data to carry out regression operation, while using all the data to carry out operation may increase unnecessary error.

Conclusion

In this paper, we put forward one kind of method by combing with modified Frequent Itemsets mining and NN search (FINN) to make estimation for the missing data. Because it mainly finds similar data to make operation, so it can reduce estimation error to a certain extent. Because there may be not only 1 similar node, so it can use similarity of many sensor nodes to get relatively precise estimation value. Even if it is failed in finding similar node, because the adjacent node usually has similar change trend, we can find nearest neighbor of node to make linear regression

operation, which can better estimate the missing value. The test result indicates that this algorithm has relatively better precision.

References

[1] Cool, A. L; A review of methods for dealing with missing data; Annual Meeting of the Southwest Educational Research Association, Dallas, TX. 2000.

[2] Gabriele B. Durrant; Imputation Methods for Handling Item-Nonresponse in the Social Sciences: A Methodological Review; ESRC National Centre for Research Methods and Southampton Statistical Sciences Research Institute (S3RI); June 2005.

[3] Zeng, Yan; A Study of Missing Data Imputation and Predictive Modeling of Strength properties of Wood Composites; Master's Thesis; 2011.

[4] G. McLachlan and K. Thriyambakam; The EM Algorithm and Extensions; John Wiley & Sons; 1997.

[5] Allison, P. D. Missing data. Thousand Oaks, CA: Sage; 2002.
P. Dempster; N. M. Laird; D. B. Rubin; Maximum Likelihood from Incomplete Data via the EM Algorithm; In Journal of the Royal Statistical Society. Series B (Methodological), Vol. 39, No. 1. (1977), pp.I-38.

[6] Gelman, J. Carlin, H. Stern, and D. Rubin; Bayesian Data Analysis; Chapman & Hall; 1995.

[7] M. Halatchev and L. Gruenwald; Estimating Missing Values in Related Sensor Data Streams; Int'l Conf. on Management of Data; January 2005.

[8] Niu. K, Zhao. F, and Qiao. XQ; A Missing Data Imputation Algorithm in Wireless Sensor Network based on Minimized Similarity Distortion; 2013 6th International Symposium on Computational Intelligence and Design (ISCID); 2013.

[9] Barladi, A. N. &s; Enders, C. K; An introduction to modem missing data analyses; Journal of School Psychology, 48, 5-37; 2010.