Proceedings of the 8th International Conference on Applied Informatics
Eger, Hungary, January 27–30, 2010. Vol. 2. pp. 63–71.

An Enhanced Data Mining Technique for Efficient Query Processing in Wireless Sensor Networks*

Muhidul Islam Khan

Corvinus University of Budapest, Hungary
e-mail: muhit_smart@yahoo.com

Abstract

The way of collecting sensor data will face a revolution when the newly developing technology of distributed sensor networks becomes fully functional and widely available. Distributed sensor networks are indeed an attractive technology, but the program/stack memory and the battery life of today nodes do not enable complex data mining in runtime. Effective data mining can be implemented on the central base station, where the computational power is not generally constrained. Today's real-world databases are highly susceptible to noisy, missing, and inconsistent data because of their typically huge size and their likely origin from multiple, heterogeneous sources. Low-quality data will lead to low-quality mining results. There are many possible reasons for noisy data (having incorrect attribute values). The data collection sensor nodes used may be faulty. Errors in data transmission can also occur. There may be technology limitations, such as limited buffer size for coordinating synchronized data transfer and consumption. Incorrect data may also result from inconsistencies in naming conventions or data codes used or inconsistent formats for input fields. Duplicate tuples also require data cleaning. Preprocessing is required to remove noisy, missing and inconsistent data for efficient mining in Wireless Sensor Networks (WSN) data. A number of research works have been done for mining WSN data. No research work has been found to be done on preprocessing the WSN data for efficient query processing. In this paper, Artificial Neural Networks (ANNs) has been used for missing field data recovery. A pass through architecture via hash technique has been used to remove duplicate data values from a dataset. WSN datasets available in the internet has been used for experimental purpose. Bond Energy Algorithm has been applied for clustering the dataset. Cleaned and

*The work was carried out under the framework of EU funded eLINK project (eLINK – east-west Link for Innovation, Networking and Knowledge exchange Programme ECW-149674-UK-L12_2008-4949).

clustered dataset has shown better performance for query processing than dirty and non clustered data.

1. Introduction

Data mining is required for efficiently data retrieving from data sources. Sensor nodes are limited to battery power and memory space. A suitable technique is required for mining sensor data. Dirty data is a challenge for efficient mining and hence preprocessing is required. Various mining techniques can be applied to the preprocessed data. Clustering is required to make the searching efficient. Searching data from a huge database is not suitable. We need high quality data for high quality mining. Sensor networks collect information from sensor nodes which may be faulty. Data transmission errors, limited memory are also challenges for clean data. Duplicate records can hamper the preprocessed tasks. A suitable technique is required for cleaning the duplicate records. After preprocessing, clustering can be applied to data. If we can cluster the data it will make the query processing more efficient. In this paper, we have proposed data cleaning system based on a pass through architecture via hash technique.

We have designed clustering system based on Bond Energy algorithm. Queries can be performed on the clustered data that improves the query performance. There are some limited works on mining technique for sensor data. Section 2 mentions some related works on this. Section 3 discusses the proposed mining technique. Result and discussion is given in section 4. Section 5 is the conclusion.

2. Related Works

Several works and studies have been performed regarding data mining in WSN Data. These are Artificial Neural Network based approach [1], Kohonen Map Implementation [2], Adaptive Modular Approach [3], Local Hill Climbing Approach [4]. Artificial Neural Network Implementation [1] presents two possible implementations of the ART and FuzzyART neural networks algorithms. It is unsupervised learning methods for categorization of the sensory inputs. They are tested on data obtained from a set of several motes, equipped with several sensors. A framework for building and deploying predictors in sensor networks that pushes most of the work out to the sensors themselves. Kohonen Map Implementation [2] enables sensors to respond to changes in data by relearning when their local predictive accuracy changes. This creates new possibilities, such as allowing sensors to predict only some target classes. Adaptive Modular Approach [3] is a two-layer modular architecture to adaptively perform data mining tasks in large sensor networks. Lower layer performs data aggregation and upper layer performs local learning technique. Local Hill Climbing Approach [4] shows that in some cases hill climbing can be solved using a local algorithm. Local algorithms are important for sensor networks because they have superb message pruning capabilities and because they perform

their entire computation in-network.

In Artificial Neural Network Implementation, two models are presented for categorization of sensory inputs. There is no clustering and cleaning technique. In Kohonen Map Implementation, it allows sensors to predict only some target classes. It is not a proper data mining technique. In Adaptive Modular Approach, it gives a structure only. Data mining technique is not purely defined. In Local Hill Climbing Approach, it develops local algorithms for sensors which is capable of solving only the centralized problem. It solves a particular problem. Nothing is mentioned about clustering technique.

3. Proposed Mining Technique

Our proposed mining technique works on preprocessed data. The features of the proposed mining technique are as follows:

- Preprocessing of sensor data. A pass through architecture via hash technique is used to identify and remove the duplicate records.
- Artificial Neural Networks (ANNs) has been used for missing field data recovery.
- Data Clustering is performed on both clean and dirty data based on Bond Energy Algorithm. It reduces the query response time over the network.

3.1. Proposed Technique for Data Cleaning

A method and system for removing duplicate query results in a database system comprising a plurality of data sources. The method and system includes issuing a query from a user to a first data source. In response to receiving a first query result from the data source, a first hash index is computed for the first query result and the first query result is passed on to the user. The method and system further includes receiving a second query result and computing a second hash index for the second query result. The first hash index is then compared with the second hash index to cheek for a hash collision. If the first, and second hash indexes match, the first data source is queried for data corresponding to the second query result. And if the first data source contains the data, then the second query result is considered a duplicate and is discarded. A completely different approach to searching is provided by hashing: directly referencing records in a table by doing arithmetic transformations on keys into table addresses. If we were to know that the keys are distinct integers from 1 to N, then we could store the record with key i in table position i, ready for immediate access with the key value. Hashing is a generalization of this trivial method for typical searching applications when we don't have such specialized knowledge about the key values. The first step in a search using hashing is to compute a hash function which transforms the search key into a table address. No hash function is perfect, and two or more different

keys might hash to the same table address: the second part of a hashing search is collision resolution process which deals with such keys.

Hash Functions

The first problem we must address is the computation of the hash function which transforms keys into table addresses. This is an arithmetic computation with properties similar to the random number generators that we have studied. What is needed is a function which transforms keys (usually integers or short character strings) into integers in the range [O..M-11], where A4 is the amount of memory available. An ideal hash function is one which is easy to compute and approximates a "random" function: for each input, every output should be "equally likely." Since the methods that we have used are arithmetic, the first step is to transform keys into numbers which can be operated on (and are as large as possible). For example, this could involve removing bits from character strings and packing them together in a machine word. From now on, we assume that such an operation has been performed and that our keys are integers which fit into a machine word. One commonly used method is to take A4 to be prime and, for any key k, compute $h(k) = k \bmod M$. This is a straightforward method which is easy to compute in many environments and spreads the key values out well. A second commonly used method is similar to the linear congruential random number generator: take $M = 2^m$ and $h(k)$ to be the leading m bits of (bkmod w), where w is the word size of the computer and b is chosen as for the random number generator. This can be more efficiently computed than the method above on some computers, and it has the advantage that it can spread out key values which are close to one another (e.g., templ, temp$, temp3).

3.2. Proposed Technique for Missing Data Handling

ANNs are able to solve problems without any prior assumptions. As long as enough data are available, a neural network will extract any regularities or patterns that may exist and use it to form a relationship between input and output. ANNs have probably become the most efficient tools for generalization problems. The technique is also able to provide a map from one multivariable space to another through training, even when given a set of data with noise. These properties make ANNs well suited to problems of estimation and prediction for flow phenomena. Usually, the data set is divided into training, cross-validation, and testing portions. The training part is used to identify the optimal weights to bridge the input/output series while the cross-validation is used to monitor the training process to avoid overtraining. The testing part is used to examine the performance of the ANNs so it is not used in the training process. The most popular ANNs algorithm is the classical multilayer perceptron (MLP) model. MLPs (Figure 1) are feed-forward neural networks trained with a standard backpropagation algorithm. This is a topology of ANNs with eight inputs, one hidden layer (three nodes), and one output system. They are supervised networks, so they must be trained for the desired

response. They can learn how to transform the input data into the desired response if sufficient patterns are present in the training data set. With one or two hidden layers, an MLP can approximate the performance of optimal statistical classifiers in difficult problems. Two other two algorithms, namely time-lagged neural networks (TDNN) and recurrent neural networks (RNN) are more powerful algorithms to solve time series forecasting and prediction problems requiring the capability of addressing time delay problems.

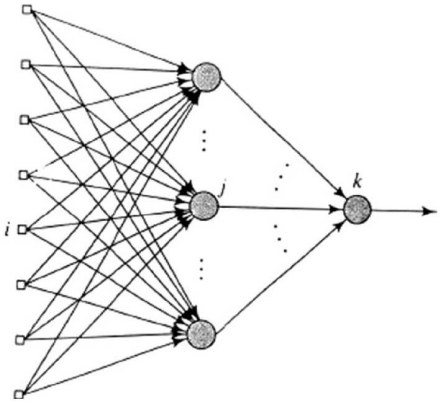

Figure 1: Fully connected feed-forward network with one hidden layer and output layer

DATA RECOVERY SYSTEM (DRS): The DRS for missing data is based on the transfer function (response function) approach. The identification of system response is constructed by the training and cross-validation processes of learning from a common period between input and output series for ANNs. The testing portion (performance), which is not involved in the training, is used to compute the simulated output from additional input series. The simulated output using optimal weights from the best fit activation function (transfer function) can generate a recovered data series. This series is called the missing window. Three types of DRS are defined as follows:

a. **Self-recovery:** This type of recovery is based on a single time series itself. In this situation no other series can be used as the reference to create the response bridge. The method is to break a long time series into two portions. The first part of the data is considered as the input, and the second part is regarded as the output function and contains the missing window. Longer time series training data sets that contain more significant patterns are critical for output performance.

b. **Neighboring station recovery:** This is the most typical recovery case. Obviously, the local recovery should have better performance than the remote recovery. If the involvement between input and output functions is

a different parameter, this recovery is classified as the different parameter recovery. Otherwise, it is called the same parameter recovery.

c. **Multivariate parameter recovery:** Since the system response from input to output could involve more than one variable and receive different time delay, this more complex system requires physical cause and effect to identify the system structure. For example, the salinity variation for a particular location could be caused by the source tide, local wind, and nearby freshwater inflow for an estuary system.

3.3. Proposed Technique for Clustering

Clustering is similar to classification in that data are grouped. The groups are not predefined. The grouping is accomplished by finding similarities between data according to characteristics found in the actual data. Elements from different clusters are not alike. The distance between points in a cluster is less than the distance between a point in the cluster and point outside it. Bond Energy clustering algorithm in which items are moved among sets of clusters until the desired set is reached. A high degree of similarity among clusters is obtained, while a high degree of dissimilarity among elements in different clusters is achieved simultaneously.

Bond Energy Algorithm:
Step 1: Set $i = 1$. Arbitrarily select any row and place it.
Step 2: Place each of the remaining $n - i$ rows in each of the $i + 1$ positions (i.e. above and below the previously placed i rows) and determine the row bond energy for each placement using the formula

$$\sum_{i=1}^{i+1} \sum_{j=1}^{m} a_{ij}(a_{i-1,j} + a_{i+1,j}).$$

Select the row that increases the bond energy the most and place it in the corresponding position.
Step 3: Set $i = i + 1$. If $i < n$, go to step 2; otherwise go to step 4.
Step 4: Set $j = 1$. Arbitrarily select any column and place it.
Step 5: Place each of the remaining m-j rows in each of the $j + 1$ positions (i.e. to the left and right of the previously placed j columns) and determine the column bond energy for each placement using the formula

$$\sum_{i=1}^{n} \sum_{j=1}^{j+1} a_{ij}(a_{i,j-1} + a_{i,j+1}).$$

Step 6: Set $j = j + 1$. If $j < m$, go to step 5; otherwise stop.

4. Result and Discussion

We have implemented A Pass Trough Architecture via Hash Technique in Java platform. We have used both dirty and clean data for our experiment purpose. The data we have used here is the average daily sensed temperatures, computed from 24 hourly temperature readings. Source: http://www.engr.udayton.edu. The data fields in each file consist of month, day, year, average daily temperature. Total Data Values=55000.

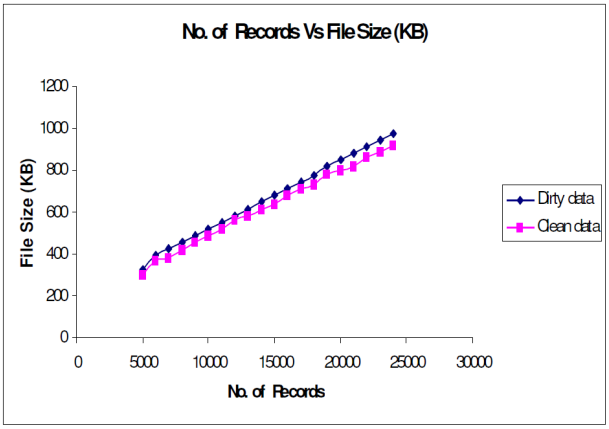

Figure 2: No. of Records Vs File Size

We have applied Bond Energy Clustering technique to cluster sensor data. The data set that we have used is available on the site www.engr.udayton.edu. The clustering technique has been applied for both noisy and clean data. The simulation result has shown that the number of iterations required is less for clean data than for noisy data. For same number of clusters and same number of data we get better performance for clean data.

For Noisy data-

- ❖ The number of Iterations required is 10
- ❖ The minimum distance between initial clusters is 32.200

For Clean data-

- ❖ The number of Iterations required is 7
- ❖ The minimum distance between initial clusters is 30.700

Less iterations is required for clean data cause after removing dirty data, Bond Energy Algorithm has been applied to reduced size of data. As the size of data has been reduced the minimum distance between initial clusters is also less than dirty data. So, data cleaning can improve the effectiveness for data mining. SPSS is used as a simulator. We have performed clustering algorithm both on noisy and clean data. Figure 3 shows comparison between non clustered dirty data and non

clustered clean data. Figure 4 shows comparison between clustered dirty data and clustered clean data.

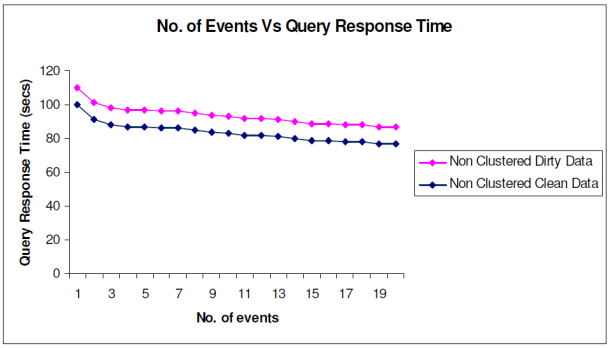

Figure 3: No. of Events Vs Query Response Time

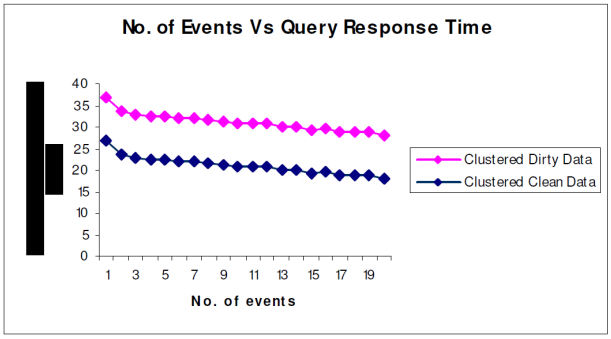

Figure 4: No. of Events Vs Query Response Time

The query response time has been decreased with the increasing number of events cause we have applied both clustered and non clustered data on the WSN protocol A-sLEACH. In the A-sLEACH cluster head takes the charge of query response after some rounds and they handle the operations. So, query response time has been decreased with the increasing of events. OMNET++ has been used as a simulator.

5. Conclusion

For efficient mining of sensor data clean data plays an effective role. In this paper, redundant data has been removed by a pass through architecture via hash technique and the Bond Energy Algorithm has been applied which will discover knowledge from preprocessed sensor data with less time requirement. Pass Trough Architecture via Hash Technique has been applied on both dirty and clean data. Clean data has reduced the file size in every cases. Clustering technique has been

applied both on dirty and clean data. Comparisons have been performed for clustered and non clustered data. The number of iterations both for dirty and clean data has also been compared. Better results have come for clean and clustered data. We have a future plan to implement various clustering techniques on sensor data and find out the best one.

References

[1] C.,Virginio, L., Luca and L., Paolo, "Challenges for data mining in distributed sensor networks", European Commission-Joint Research Centre, IPSC, TP210 via Fermi 1,21020 Isapa, Italy. Pattern Recognition, 2006. ICPR 2006. 18th International Conference on Volume 1, Issue , 0-0 0 Page(s):1000–1007.

[2] K., Andrea and D., Danco, "Data mining in wireless sensor networks based on artificial neural-networks algorithms", Computer Science Department, Faculty of Electrical Engineering, Skopje, Macedonia. 2005 SIAM International Conference on Data Mining, Sutton Place Hotel, New Port Beach, CS, April 21-23, 2005.

[3] M.,Sabine and S., David, "A distributed approach for prediction in sensor networks", School of Computing, Queen's University. 2005 SIAM International Conference on Data Mining, Sutton Place Hotel, New Port Beach, CS, April 21-23, 2005.

[4] B., Gianluca and B., Yann-Ael, "An adaptive modular approach to the mining WSN data", ULB Machine Learning Group, Universite Libre de Bruxelles, Belgium. 2005 SIAM International Conference on Data Mining, Sutton Place Hotel, New Port Beach, CS, April 21-23, 2005.

[5] Computer Science Dept., Technion-Israel, "Local hill climbing in sensor networks", 2005 SIAM International Conference on Data Mining, Sutton Place Hotel, New Port Beach, CS, April 21-23, 2005.

[6] For data files- `http://www.engr.udayton.edu`