INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS
ISSN 2320-7345

EFFICIENT ALGORITHM FOR DETECTING AND CORRECTING ERRORS IN BIG SENSOR DATA

P.Kavya[1], M.Ananthi [2]

[1]PG Student [CSE], Dept of CSE, Info Institute of Engineering, Coimbatore, TamilNadu, India
[2]Assistant Professor, Dept. of CSE, Info Institute of Engineering, Coimbatore, TamilNadu, India

E-Mail: kavyapalanisamy92@gmail.com

Abstract: - Detecting error in the large volume of data is the most complex process where the number of data's are grows in size. In the existing work, time efficient approach is proposed to detect the errors resides in the big sensor data where the collected data would be partitioned into multiple partition and the errors will be identified and located by comparing it with the error patterns which are predefined. However this work couldn't correct the errors which requires sender to retransmit the data again which might increase the time complexity. This problem is overcome in the proposed methodology by introducing the forward error correction method which will correct the errors present in the big sensor data's automatically. In the proposed system, we propose neural network algorithm for error detection robustly rather than existing system. The error detection is used to reduce the incorrect information by fault sensors in big data set. The proposed approach is increasing the efficiency and reliability of big sensor data. From the experimental result, the conclusion says that the proposed system is superior to existing system by means of higher performance.

Keywords: Forward error correction, error detection, sensor data, neural network.

I. INTRODUCTION

The term big data refers to large amount of information which is collected by us and our surroundings. Big data varies depending on the capabilities of the user and their tools.2.5 quintillion bytes of data were created every day, and 90% of the data is created in the last two years alone [1],[6]. Big data requires a set of techniques and technologies with new forms of integration. The latency requirement is the main consideration for big data platform. Big data can be described by the following five characteristics, such as volume, variety, velocity, veracity and value.

One of the main and important sources for big data is the data collected by wireless sensor networks (WSN). Wireless sensor networks have potential of significantly enhancing people's ability to monitor and interact with their physical environment [2]. Sensor networks have been used in the context of high-end applications such as radiation and nuclear-threat detection systems, military applications, etc. A sensor network is an infrastructure comprised of sensing, computing, and communication elements. Big data set from sensors is often subject to corruption and losses due to wireless medium of communication and presence of hardware inaccuracies in the nodes. It is necessary that the data received is clean, accurate, and loss-less in WSN. However, effective detection and cleaning of sensor big

data errors is a challenging issue. Some work for big data analysis and error detection in complex networks including intelligence sensors networks has been done [7], [8]. Error detection and debugging with online data processing techniques has been also done [9], [13]. Since these techniques not deal with big data, so they were unable to cope with current dramatic increase of data size. Some work has been done about processing sensor data on cloud [11], [13]. However, fast detection of errors in big data remains challenging. The error detection approach in this paper will be based on neural networks.

This paper is organized as follows. Drawbacks of existing system were present in the Chapter 2. Chapter 3, outlines the proposed system and the workflow of the proposed work process. Chapter 4 presents the results to validate the performance. Finally, Chapter 5 draws some brief conclusions and future enhancement.

II. EXISTING SYSTEM

A. Partition of Sensing Data Set:

In order to effectively deploy our existing algorithm on cloud, the data sets need to be partitioned before feeding to the algorithm on cloud. When the whole data set D is partitioned into Di; 1 i q, it need to guarantee that the distribution of data set in a cluster Di is similar to D. A sub data set Di, here can be treated as a point in an m-dimension space. According to the partition principle, to avoid the new error or error type change, during the process of partition, light weighted error type matching has to be carried out for warning the new abnormalities during the partition.

B. Error Definition and modelling

The data record from a network node is denoted as r(n, t, f(n, t), g(n, l)), where n is the ID of the node in a network systems. t represents the window length of a time series. f(n, t) is the numerical values collected within window t from the node n. g(n, l) is a location function which records the cluster, the data source node and partition situation related to the node n. g(n, l) is used to calculate the distance between the data source node n and the node l which is the initial data source node. g(n, l) indicates that a current detected error data node is the initial data source node. Furthermore, g(n, l) is also used to parse the data routing between data communication nodes.

The errors which are defined here are, Node side Flat line error, Edge side Flat line error,

Node side Data Lost error, Edge side Data Lost error, Node side Out of Bounds error, Edge side Out of Bounds error, Node side Spike error, Edge side Spike error, Aggregation and Fusion error.

C. Error Detection

At the phase of error detection, there are three inputs. The first is the graph of network. The second is the total collected data set D and the third is the defined error patterns p. The output of the error detection algorithm is the error set D'.

It can be concluded that this error detection techniques reduces the overall performance, and increase the time complexity. The proposed approach in this paper aim to address this issue by neural network algorithm.

III. PROPOSED SYSTEM

A. Error Detection Using Neural Network:

Neural network is used to train the data which is received from the sensors and then it detects incorrect information. It increase's its reliability and updates the training data. Error detection with new data and neural network's result is used to improve the accurate data collection.

Input: N training samples,

Output: predicted class

For each sample do

Input[i] =sample

For each i in neural network do

Output [i] = module. Forward Propagate(input[i])

Input [i+1] =output[i]

End

Predicted class = criterion (output)

If training then

Check error attributes

For each [k-i] in neural network do

Output = module. Backward Propagate

Input [i+1] =output[i]

End

End

End

Description

For the number of input training sample we have to perform the analysis of error detection. For each input sample the neuron network perform the output sample based on the prior knowledge. It is used to search the more accurate results along with hidden layer and this layer is used to map the error results for corresponding input sample. Hence it detects the error values also it progress the speed of process more efficiently.

B. Error Localization:

After error detection, it is important to locate the position and source of the detected error in the original WSN graph G(V, E). The inputs are the original graph of a scale-free network G (V, E), and an error data D. The output is G'(V', E') which is the subset of the G to indicate the error location and source.

C. Error Correction

Forward Error Correction (FEC) codes can detect and correct a limited number of errors without a feedback channel. Block codes and Convolution codes are its types. BCH code is the significant example of it. For the purpose of detecting and checking the errors, BCH encodes k data bits into n code bits by adding n-k parity checking bits. Given the length of the codes is $n = 2^m-1$ for any integer $m\geq3$, we will have t (where $t<2^m-1$), is the bound of the error correction. BCH can correct any combination of errors (burst or separate) fewer than t in the n-bit-codes. The number of parity checking bits is $n-k \leq mt$.

Galois Fields (GF), an important concept of BCH. GF is a finite set of elements on which two binary addition and multiplication can be defined. For any prime number p there is GF (p) and GF (p^m) is called extended field of GF (p). We often use GF (2^m) in BCH code. A GF can be constructed over a primitive polynomial such as $p(x) = x^4+x+1$. Usually, GF table records all the variables, including expressions for the elements, minimal polynomial, and generator polynomial. By referring to the table, we can locate a proper generator polynomial for encoder.

The BCH decoding is complicated because it has to locate and correct the errors. Suppose we have a received codeword $r(x)=r_0+r_1x+r_2x^2+\ldots+r_{n-1}x^{n-1}$, then $r(x)=v(x)+e(x)$, where, v(x) is correct codeword and e(x) is the error. First, we must compute a syndrome vector $s= (s_1,s_2,\ldots,s_{2t})$, which can be achieved by calculating $r.H^T$, where, H is parity-check matrix and can be defined as:

$$H = \begin{bmatrix} 1 & \alpha & \alpha^2 & \ldots & \alpha^{n-1} \\ 1 & \alpha^2 & (\alpha^2)^2 & \ldots & (\alpha^2)^{n-1} \\ . & . & . & \ldots & . \\ . & . & . & \ldots & . \\ 1 & \alpha^{2t} & (\alpha^{2t})^2 & \ldots & (\alpha^{2t})^{n-1} \end{bmatrix}$$

Here, α is the element of the GF field and can be located in the GF table. The location numbers for the errors will be achieved by finding roots of σ(x).

Algorithm

Encode

For i=1…n do

For j=1 …k do

$T_j [h_j ((x_i, i))]$. Keysum = (x_i, i)

$T_j [h_j((x_i, i))]$.valueSum = Check((xi; i))

Decode

for i = 1….. n do

for j = 1 …. k do

$T_j [h_j((y_i, i))]$:keySum ^= (y_i, i).

$T_j [h_j((y_i, i))]$:valueSum ^= Check((y_i, i)).

While ∃ a, j with (Tj [a]:keySum ≠ 0) and

(Tj [a]:valueSum == Check(Tj [a]:keySum)) do

(z, i) = Tj [a].keySum

if z ≠yi then

set y_i to z when decoding terminates

for j = 1 …. k do

$T_j [hj((z; i))]$.keySum ^= (z, i).

T_j [hj((z; i))].valueSum ^= Check((z, i)).

This error correction method is used to determine the error pairs more effectively and correct the errors using above procedures.

IV. EXPERIMENT RESULTS

A comparative experiment between existing and proposed algorithm is conducted. As shown in Fig. 1, the accuracy, precision, recall and F-Measure are high in proposed algorithm when compared to existing algorithm. Based on the above result, it can be concluded that our proposed error detection and correction approach for big data can dramatically improve the performance of the system and provides higher accuracy and Fast computation in detection rates.

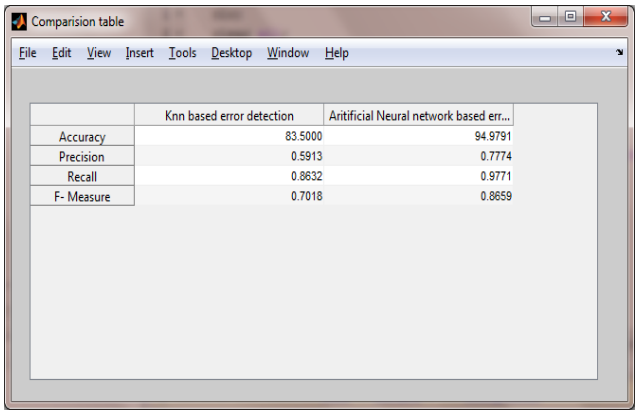

Fig. 1. Comparison of two error detection strategies.

V. CONCLUSION AND FUTURE WORK

Error occurrence in the big sensor data becomes the greatest issue in the real world application which affect the original behaviour of the data's that are collected. Detecting errors that are present in the large volume would consume more time complexity. In the existing work it is resolved by comparing the source error patterns with the gathered data entries to find and locate the errors that are present in the environment. The existing system is focused on only detection errors which cannot correct the errors that are detected. This is resolve by using forward errors correction mechanism which can correct the errors that are present in the environment. Also in the proposed system, we propose neural network algorithm to significantly detect the errors and to increases the accuracy. In future, we can develop an advanced algorithm to progress the optimal performance in the given scenario and reduces the error rates greatly. It will improve the higher accuracy in big sensor data applications.

REFERENCE

[1] S. Tsuchiya, Y. Sakamoto, Y. Tsuchimoto, and V. Lee, "Big Data Processing in Cloud Environments," FUJITSU Science and Technology J., vol. 48, no. 2, pp. 159-168, 2012.
[2] D.J. Wang, X. Shi, D.A. Mcfarland, and J. Leskovec, "Measurement Error in Network Data: A Re-Classification," Social Networks, vol. 34, no. 4, pp. 396-409, Oct. 2012.
[3] D. Xiong, M. Zhang, and H. Li, "Error Detection for Statistical Machine Translation Using Linguistic Features,"Proc. 48th Ann. Meeting of the Association for Computational Linguistics (ACL'10),pp. 604-611, 2010.
[4] S. Mukhopadhyay, D. Panigrahi, and S. Dey, "Model Based Error Correction for Wireless Sensor Networks," IEEE Trans. Mobile Computing, vol. 8, no. 4, pp. 528-543, Sept. 2008.
[5] M.C. Vuranand and I.F. Akyildiz, "Error Control in Wireless Sen-sor Networks: A Cross Layer Analysis," IEEE Trans. Networking, vol. 17, no. 4, pp. 1186-1199, Aug. 2009.
[6] "Big Data: Science in the Petabyte Era: Community Cleverness Required," Nature, vol. 455, no. 7209, p. 1, 2008.

[7] A. Sheth, C. Hartung, and Richard Han, "A Decentralized Fault Diagnosis System for Wireless Sensor Networks,"Proc. IEEE Sec-ond Conf. Mobile Ad-hoc and Sensor Systems (MASS '05), Nov. 2005.
[8] N. Laptev, K. Zeng, and C. Zaniolo, "Very Fast Estimation for Result and Accuracy of Big Data Analytics: The EARL System," Proc. IEEE 29th Int'l Conf. Data Eng. (ICDE), pp. 1296-1299, 2013.
[9] E. Elnahrawy and B. Nath, "Online Data Cleaning in Wireless Sensor Networks,"Proc. First Int'l Conf. Embedded Networked Sensor Systems (ACM Sensys '03), pp. 294-295, 2003.
[10] Chi Yang, Chang iu, xuyun Zhang, Surya Nepal and Jinjun Chen " A Time Efficient Approach for Detecting Errors in Big Sensor Data on cloud", IEEE Trans. Networking, vol. 26, no. 2, pp. 1045-9219, Feb. 2015
[11] "Sensor Cloud ,http://www.sensorcloud.comaccessed on 30,Aug. 2013
[12] M.H. Lee and Y.H. Choi, "Fault Detection of Wireless Sensor Networks,"Computer Comm., vol. 31, no. 14, pp. 3469-3475, 2008.
[13] M. Yuriyama and T. Kushida, "Sensor Cloud Infrastructure,"Proc.13th Int'l Conf. Network-Based Information Systems (NBiS), pp. 1-8, 2010