Method of Detecting Resource-intensive Inquiries to Databases of Agro-industrial Resources Geo-ecological Monitoring Systems

Jalal Qais Jameel
Mustansiriyah University, College of Medicine Qadisaya Street, Baghdad, Iraq.
Belgorod State University, Pobedy Street, Belgorod, Russia.
E-mail: jalalalqaisy1@gmail.com

Tareq Nasser Mahdi
Mustansiriyah University, College of Pharmacy Qadisaya Street, Baghdad, Iraq.
Belgorod State University, Pobedy Street, Belgorod, Russia.
E-mail: tareq.nasser.m@gmail.com

Oleg R Kuzichkin
Belgorod State University, Pobedy Street, Belgorod, Russia.
E-mail: kuzichkin@bsu.edu.ru

Konstantin A. Polshchykov
Belgorod State University, Pobedy Street, Belgorod, Russia.
E-mail: polshchikov@bsu.edu.ru

Sergej A Lazarev
Belgorod State University, Pobedy Street, Belgorod, Russia.
E-mail: lazarev_s@bsu.edu.ru

Ilya K. Polshchikov
Belgorod State University, Pobedy Street, Belgorod, Russia.
E-mail: polshchykov@mail.ru

Received September 21, 2021; Accepted December 17, 2021
ISSN: 1735-188X
DOI: 10.14704/WEB/V19I1/WEB19259

Abstract

This article aims to achieve more efficient automated software systems use in the geo-ecological monitoring of agro-industrial sector resources. A method of detecting resource-intensive inquiries to agricultural resources databases is developed. Self-Organizing Map is used for clustering inquiries to databases in the method. Additionally, an algorithm is proposed to discover resource-intensive inquiries and the corresponding software.

The performance evaluation demonstrates that the suggested method considerably increases the correctness of detecting resource-intensive inquiries to databases compared to other

counterparts. Accordingly, in geo-ecological monitoring of agricultural objects and resources, the method is recommended for practical application.

Keywords

Self-Organizing Map, Databases, Resource-intensive Inquiries, Agro-industrial Sector geo-ecological Monitoring.

Introduction

Geo-ecological monitoring is an integrated system of routine long-term surveillance in time and space, assessing and forecasting the natural environment condition under natural and man-made factors. This is a routine surveillance system over ecological processes in various components of natural objects and technical objects in their interaction process. For example: 1) in the process of geo-ecological monitoring; 2) the condition of sedimentary rocks; 3) their composition and structure; 4) depth of soils; 5) chemical composition; and 6) aggressiveness of groundwaters are being monitored (Romanov et al., 2019, Vasilyev et al., 2018, Kuzichkin et al., 2020).

Geo-ecological monitoring is of particular importance in the agro-industrial sector, as evaluating and forecasting the condition of agricultural lands is performed based on it. Geo-ecological monitoring of agro-industrial resources requires transmitting, storing, and processing large amounts of information (Polshchykov et al., 2021). For this reason, automated software systems for geo-ecological monitoring are implemented.

In these systems, the databases are used to store the information of the objectives of agricultural resources. To perform needed operations, many users are intensively accessing the databases. The quality and promptness of these transactions are adversely affected by resource-intensive inquiries incoming to databases. A resource-intensive (problematic) inquiry to a database is an inquiry. The server's processing consumes a prohibitive amount of memory resources, disc, a processor and time. (Alghazali et al., 2021). Database Administrators (DBA) perform the conversion and search of resource-intensive inquiries.

The DBA specialists access database server software and hardware settings and the specialized summarized statistical information about query execution. According to such information analysis findings, database administrators correct resource-intensive inquiries either manually or by using special software applications – query optimizers. Inquiries to databases are formed by special features of SQL (Structured Query Language) (Jang, 2020, Wang, Cheungc & Bodik, 2017). The currently used methods of search and identification

of problematic inquiries, applied in system utility programs, do not always detect resource-intensive SQL operators or can miss some inquiries, categorized as resource-intensive. The circumstances mentioned above obtain the relevance of designing new smart tools for the prompt and correct discovery of resource-intensive inquiries to agricultural resources databases.

This work aims to develop a method of detecting resource-intensive inquiries to databases of agricultural resources.

Main Part

The analysis of scientific and technical literature has demonstrated that to solve the problem of detecting resource-intensive inquiries to databases it is advisable to use artificial intelligence methods that are successfully applied in various fields (Polshchykov et al., 2020, Ivaschuk et al., 2016, Polshchykov, Lazarev, & Zdorovtsov, 2017, Konstantinov et al., 2017, Polshchykov et al., 2019). Self-Organizing Map (SOM) is one of the neural network's types. SOM is characterized by resistance to noisy data, high learning rate, and the possibility of multivariable input data approximation by means of visualization (Girau, Torres-Huitzil, 2020, Guaman, Delgado, & Perez 2021, Chen, Chang, & Chang, 2018, Nguyen, 2019, Pen, Wang, Wang, 2021, Chushig-Muzo et al., 2020, Abarca-Alvarez et al., 2019, Chang, Wang, & Chang, 2020, Qu, Yang, & Guo, 2021, Alqudah et al., 2018, Chamundeswari, Varma, & Satyanarayana, 2018, Galutira, Medina, & Fajardo, 2019). The SOM algorithms are successfully used in practice and allow approximating any continuous functions. For creating and studying neural self-organizing maps, the Python versions of basic and modified SOM algorithms can be used and the TensorFlow open-source software library for machine learning. Based on neural network SOM-clustering, a method of detecting resource-intensive inquiries has been developed, as described below.

Let *M* the number of inquiries be incoming to databases. Each inquiry is presented with a parameter vector:

$$X_q = \{x_{q1}, x_{q2}, \dots, x_{qi}, \dots, x_{qn}\} \qquad (1)$$

Where X_q– parameter vector of inquiry number *q*, the number of inquiries accepts values.

$$q = 1,2, \dots, M;$$

X_{qi} – parameter number *i* of inquiry number *q*.

n – number (quantity) of parameters in the query vector.

In a self-organizing map, there are *N* neurons. Each neuron is presented with a weight vector:

$$W_k = \{w_{k1}, w_{k2}, \dots, w_{ki}, \dots, w_{kn}\} \qquad (2)$$

Where W_k – weight vector of neuron number *k*, the number of neurons accepts values.
W_{ki} – weight number *i* of neuron number *k*;
n – number (quantity) of weight values of the neuron.
Moreover, this self-organizing map is a trained one. Under query parameters of this or that resource-intensiveness class, the weight values of its neurons are set up.

To detect resource-intensive inquiries, it is suggested to calculate the Euclidean distance as a proximity measure of parameter vectors of incoming inquiries with weight vectors of each neuron in the self-organizing map:

$$d_{qk} = \sqrt{\sum_{i=1}^{n} \left(x_{qi} - w_{ki}\right)^2} \qquad (3)$$

To discover resource-intensive inquiries to databases, a method was developed, which involves performing the following main stages:

1. Input of data about parameter values of inquiries are to be classified in terms of resource-intensiveness to the self-organizing network.
2. Calculation of Euclidean distance between parameter vectors of inquiries and weights of each neuron in the self-organizing map;
3. Searching active neurons with the minimum Euclidean distance between its weight values and the inquiry parameter vector's values;
4. Searching of the self-organizing map's clusters, to which the active neurons belong;
5. Generating conclusions about assigning the incoming inquiries to corresponding resource-intensiveness classes;
6. Output of information about each inquiry (classified as resource-intensive) to the database administrator for deciding further actions for correcting (excluding) such inquiries.

Figure 1 shows the developed algorithm for detecting resource-intensive inquiries where the steps are as follows:
Step 1. Setting the source of the data:

- Number of neurons in a self-organizing map;
- Number of inquiry's parameters;
- Number of weight values of neurons;
- Weight values of each neuron;
- Values of parameter vectors of inquiries, which are to be classified in terms of their resource-intensiveness.

Step 2. The parameter vector of inquiry number *q=1* is selected.

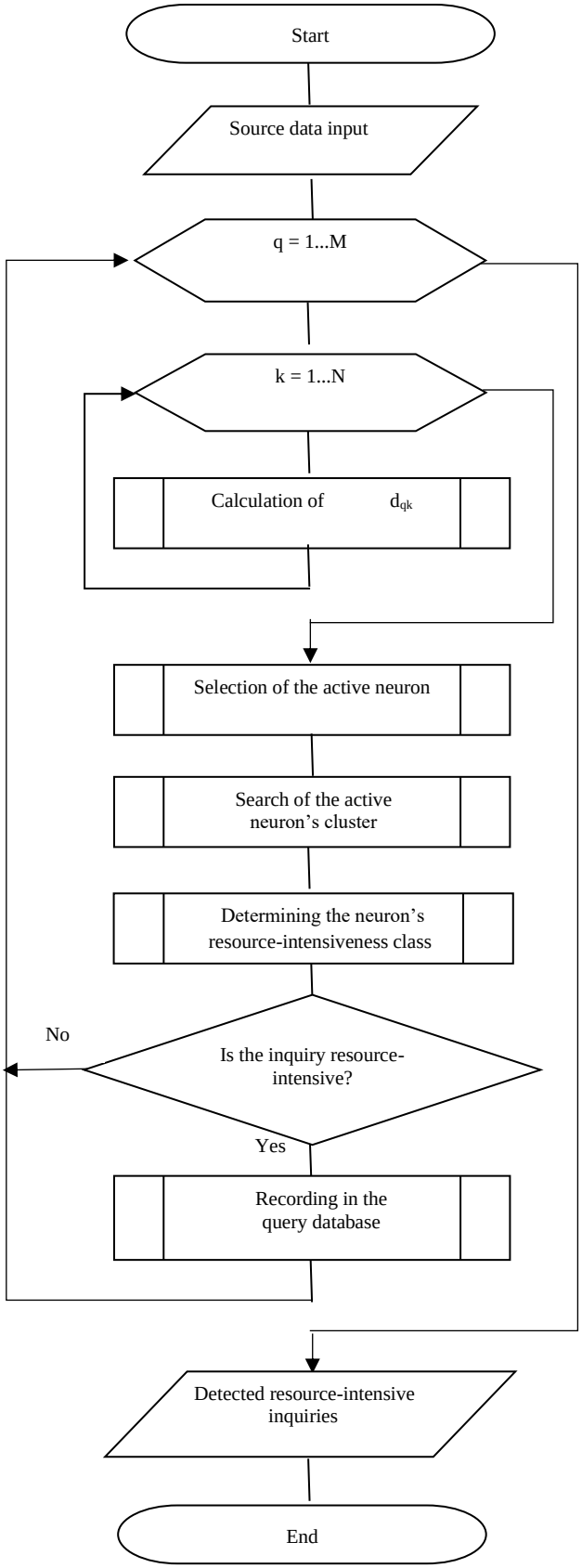

Figure 1 Block diagram of an algorithm for detecting resource-intensive inquiries to databases

Step 3. A neuron number k=1 of the trained self-organizing map is selected.
Step 4. For the selected neuron number k of the self-organizing map, a proximity measure of its weight vector with the parameter vector of the incoming inquiry is calculated by the formula (3).
Step 5. Increment the selected neuron number.
Step 6. Check the condition's fulfillment in below:

$$k \leq N. (4)$$

Jump to step 4 if the condition is fulfilled; otherwise, go to step 7.
Step 7. An active neuron with the minimum value d_{qk} is searched.
Step 8. A cluster, to which the active neuron belongs, is determined.
Step 9. On the basis of the data about the neuron's belonging to a certain cluster, a decision is made, to which resource-intensiveness class the incoming inquiry should be assigned.
Step 10. If the inquiry is classified as resource-intensive, it is put into the database of resource-intensive inquiries.
Step 11. Increment the selected neuron number. Then, check the condition's fulfillment in below:

$$q \leq N. (5)$$

Jump to step 3 if the condition is fulfilled; otherwise, go to step 12.
Step 12. Information about the detected resource-intensive inquiries is presented to the DBA administrator to decide further actions for correcting (excluding) such inquiries.
Algorithm end.
Using Self Organizing Map (SOM-clustering), the algorithms mentioned above are realized in software for clustering SQL inquiries. Using Python language based on tensor calculations through the TensorFlow library, The software applications were developed.

The number of inquiries is demonstrated with the conducted analysis. The incomings to databases are divided into the following clusters:

- The first– «heavy» resource-intensive inquiries. They are recognized by the heavily use of the processors, storage space, system memory, and output channels in the execution process.
- The second– «slow» resource-intensive inquiries. Frequent execution and compilation procedures characterize them.
- The third– spontaneous problematic inquiries. The occasional shortage of temporary resources leads to occur this cluster. The shortage is due to peak (maximum) in the

servers, network, processors, or resource shortages, appearing in the exacting other inquiries process.

- The fourth– other, not problematic (i.e., not resource-intensive inquiries).

In the developed method course of detecting resource-intensive inquiries was implemented, over 500 experiments were carried out to classify different kinds of inquiries. To detect resource-intensive inquiries, the determined outcomes were used to calculate the correctness indices.

The means of two indicators were used to assess the correctness of detecting inquiries to databases:

1. The detection probability.
2. The error detection probability.

The following expression is used to determine the value of the probability of detection of resource-intensive inquiries to databases.

$$P_{det} = \frac{Q_{det}}{Q_{\Sigma}} \tag{6}$$

Where Q_{det} and Q_{Σ} are the number of the detected and total number of resource-intensive inquiries incoming to databases, respectively.

The following formula is used to determine the probability of error detection of resource-intensive inquiries to databases.

$$P_{err} = \frac{Q_{err}}{Q_{det}} \tag{7}$$

Where Q_{err} denotes the incoming inquiries to databases. Erroneously, it is categorized as resource-intensive.

In detecting resource-intensive inquiries, in the literature, many experiments were conducted. For example: (Sharma, Singh, & Singh, 2019, Rahman, 2021) used Oracle Cost-Based Optimizer, whereas (Kuhn, Alapati, & Padfield, 2021, Cornejo, 2018) used up-to-date SQL query tuning and monitoring applications Tuning Advisor. Table 1 presents the research outcomes.

Table 1 Detection correctness outcomes of resource-intensive inquiries to databases evaluations

Applied Software	P_{det}	P_{err}
SOM-clustering	0.082	0.951
Oracle Cost-Based Optimizer	0.102	0.844
SQL Tuning Advisor	0.094	0.825

The data listed in Table 1 demonstrate the superiority of the suggested (SOM-clustering software) algorithm.

Compared to the currently applied SQL-inquiries tuning and optimization applications, the algorithm reduced the erroneous detecting probability with (12.20% to 24.39%) and increased the detecting probability with (12.68% to 15.27%).

Conclusions

A method of detecting resource-intensive inquiries to agricultural resources databases has been developed. To solve this problem, SOM has been used was substantiated based on analyzing scientific and technical literature.

The experiments demonstrated the superiority of the suggested (SOM-clustering software) algorithm. In comparison with using other similar applications, the algorithm showed a considerable reduction in erroneous detecting probability and increasing in detecting probability resource-intensive inquiries to databases. Consequently, practically, the proposed method is recommended to achieve more efficient automated software systems used in the geo-ecological monitoring of agro-industrial sector resources.

Acknowledgments

The theory of this article has been performed within the state task of the Russian Federation framework FZWG-2020-0029 “Development of theoretical foundations for building information and analytical support for telecommunications systems for geoecological monitoring of natural resources in agriculture”.

References

Abarca-Alvarez, F.J., Navarro-Ligero, M.L., Valenzuela-Montes, L.M., & Campos-Sánchez, F.S. (2019). European strategies for adaptation to climate change with the Mayors adapt initiative by self-organizing maps. *Applied Sciences*, *9*(18), 3859. https://doi.org/10.3390/app9183859

Alghazali, S.M., Polshchykov, K., Hailan, A.M., & Svoykina, L. (2021). Development of Intelligent Tools for Detecting Resource-intensive Database Queries. *International Journal of Advanced Computer Science and Applications, 12*(7). https://doi.org/10.14569/ijacsa.2021.0120704

Alqudah, A., Al-Zoubi, H., Al-Khassaweneh, M., & Al-Qodah, M. (2018). Highly Accurate Recognition of Handwritten Arabic Decimal Numbers Based on a Self-Organizing Maps Approach. *Intelligent Automation and Soft Computing*, *24*(3), 493–505. https://doi.org/10.31209/2018.100000005

Chamundeswari, G., S. Varma, G.P., & Satyanarayana, C. (2018). Contact Distribution Function based Clustering Technique with Self-Organizing Maps. *International Journal of Image, Graphics and Signal Processing*, *10*(3), 9–17. https://doi.org/10.5815/ijigsp.2018.03.02

Chang, L.C., Wang, W.H., & Chang, F.J. (2021). Explore training self-organizing map methods for clustering high-dimensional flood inundation maps. *Journal of Hydrology*, *595*. https://doi.org/10.1016/j.jhydrol.2020.125655

Chen, I.T., Chang, L.C., & Chang, F.J. (2018). Exploring the spatio-temporal interrelation between groundwater and surface water by using the self-organizing maps. *Journal of Hydrology*, *556*, 131–142. https://doi.org/10.1016/j.jhydrol.2017.10.015

Chushig-Muzo, D., Soguero-Ruiz, C., Engelbrecht, A.P., De Miguel Bohoyo, P., & Mora-Jimenez, I. (2020). Data-Driven Visual Characterization of Patient Health-Status Using Electronic Health Records and Self-Organizing Maps. *IEEE Access*, *8*, 137019–137031. https://doi.org/10.1109/access.2020.3012082

Cornejo, R. (2018). Performance tuning basics. *Dynamic Oracle Performance Analytics*, 3–16. https://doi.org/10.1007/978-1-4842-4137-0_1

Galutira, E.F., Fajardo, A.C., & Medina, R.P. (2019). A novel kohonen self-organizing maps using exponential decay average rate of change for color clustering. *Intelligent and Interactive Computing*, 23–33. https://doi.org/10.1007/978-981-13-6031-2_28

Girau, B., & Torres-Huitzil, C. (2018). Fault tolerance of self-organizing maps. *Neural Computing and Applications*, *32*(24), 17977–17993. https://doi.org/10.1007/s00521-018-3769-6

Guaman, D., Delgado, S., & Perez, J. (2021). Classifying Model-View-Controller Software Applications Using Self-Organizing Maps. *IEEE Access*, *9*, 45201–45229. https://doi.org/10.1109/access.2021.3066348

Ivaschuk, O.A., Polschykov, K.A., Lazarev, S.A., Ivaschuk, O.D., & Fedorov, V.I. (2016). Integral estimate of terrestrial compartment condition in management of biotechnosphere of rural and urban areas. *International Journal of Pharmacy and Technology*, *8*(4), 27032–27038.

Jang, Y.S. (2020). Detection of SQL Injection Vulnerability in Embedded SQL. *IEICE Transactions on Information and Systems*, *E103.D*(5), 1173–1176. https://doi.org/10.1587/transinf.2019edl8143

Konstantinov, I., Polshchykov, K., Lazarev, S., & Polshchykova, O. (2017). Model of Neuro-Fuzzy prediction of confirmation timeout in a mobile ad hoc Network. *CEUR Workshop Proceedings. Mathematical and Information Technologies*, 174–186.

Kuhn, D., Alapati, S.R., & Padfield, B. (2012). SQL tuning advisor. *Expert Indexing in Oracle Database 11g*, 205–231. https://doi.org/10.1007/978-1-4302-3736-5_9

Kuzichkin, O.R., Romanov, R.V., Dorofeev, N.V., & Grecheneva, A.V. (2020). Organization and application of information and analytical support for geological monitoring of water use. *IIOAB*, *11*(1), 20–26.

Nguyen, T.H. (2019). *Metagenome-based disease classification with deep learning and visualizations based on self-organizing maps*, 307–319.

Pen, H., Wang, Q., & Wang, Z. (2021). Boundary precedence image inpainting method based on Self-organizing Maps. *Knowledge-Based Systems*, *216*. https://doi.org/10.1016/j.knosys.2020.106722

Polshchykov, K.A., Lazarev, S.A., Konstantinov, I.S., Polshchykova, O.N., Svoikina, L.F., Igityan, E.V., & Balakshin, M.S. (2020). Assessing the Efficiency of Robot Communication. *Russian Engineering Research*, *40*(11), 936–938. https://doi.org/10.3103/s1068798x20110155

Polshchykov, K., Lazarev, S., Polshchykova, O., & Igityan, E. (2019). The algorithm for Decision-Making supporting on the selection of processing means for big arrays of natural language data. *Lobachevskii Journal of Mathematics*, *40*(11), 1831–1836. https://doi.org/10.1134/s1995080219110222

Polshchykov, K., Lazarev, S., & Zdorovtsov, A. (2018). Neuro-fuzzy control of data sending in a mobile ad hoc network. *Journal of Fundamental and Applied Sciences*, *9*(2S), 1494. https://doi.org/10.4314/jfas.v9i2s.856

Polshchykov, K., Shabeeb, A.H.T., Lazarev, S., & Kiselev, V. (2021). Justification for the decision on loading channels of the network of geoecological monitoring of resources of the agroindustrial complex. *Periodicals of Engineering and Natural Sciences (PEN)*, *9*(3), 781–787.

Qu, X., Yang, L., Guo, K., Ma, L., Sun, M., Ke, M., & Li, M. (2019). A Survey on the Development of Self-Organizing Maps for Unsupervised Intrusion Detection. *Mobile Networks and Applications*, *26*(2), 808–829. https://doi.org/10.1007/s11036-019-01353-0

Rahman, H. (2020). *Oracle database design and development in ador composite ltd*. Composite Ltd. http://dspace.daffodilvarsity.edu.bd:8080/handle/123456789/5484

Romanov, R., Kuzichkin, O., Vasiliev, G., & Mikhaleva, E. (2019). The use of the geoelectric method of the express control during the geoecological monitoring of the decentralized water supply. *2019 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon)*. https://doi.org/10.1109/fareastcon.2019.8934213

Sharma, M., Singh, G., & Singh, R. (2018). A review of different cost-based distributed query optimizers. *Progress in Artificial Intelligence*, *8*(1), 45–62. https://doi.org/10.1007/s13748-018-0154-8

Vasilyev, G. (2018). Analysis of the Combined Transfer Functions for Geotechnical Control. *18th International Multidisciplinary Scientific GeoConference SGEM2018, Science and Technologies in Geology, Exploration and Minining*. https://doi.org/10.5593/sgem2018/1.2/s02.006

Wang, C., Cheung, A., & Bodik, R. (2017). Synthesizing highly expressive SQL queries from input-output examples. *ACM SIGPLAN Notices*, *52*(6), 452–466. https://doi.org/10.1145/3140587.3062365