RESEARCH ON ACCURATE RECOGNITION OF LOCAL OUTLIER DATA IN CLOUD COMPUTING DATABASE

Na Chen\textsuperscript{1}, Lijuan Liang\textsuperscript{2}
\textsuperscript{1}Hebei Vocational College of Rail Transportation, Shijiazhuang, Hebei, 050000, China
\textsuperscript{2}Xingtai University, Xingtai, Hebei, 054001, China
E-mail: chena12@163.com

ABSTRACT: In view of the low accuracy and low efficiency of traditional methods, a new method for outlier recognition is proposed based on the combination of grid unit and density. The local outlier recognition criteria are introduced. According to the pruning rule, the set of points in the cloud database that cannot be local outliers is deleted in advance, and the remaining candidate local outliers are analyzed, and the data points satisfying the conditions are selected as local outliers. Meshing is performed to divide the data in the cloud computing database into a number of disjoint small rectangular cell areas. Rough screening and fine screening are used to determine whether the data is a local outlier based on the density of these cell areas, the specified thresholds, and the data of neighboring cells. When the cloud computing database changes, the corresponding processing rules are given. The processing code is given when the cloud computing database distribution includes or intersects. The proposed method is parallelized by MapReduce to ensure the efficient operation of the method. The simulation data experiment and network intrusion data experiment verify the accuracy and efficiency of the proposed method.

KEYWORDS: Cloud computing; database; local outlier data; accuracy; recognition

1 INTRODUCTION

In practical applications, the data in the cloud computing database often comes from different information individuals, departments, companies, and countries. In these complex and heterogeneous data sets, there may be some data objects that are significantly different from other data, and is inconsistent with the general behavior or model of the data. These data objects are called outliers. The origin of outlier in cloud computing databases is often complex and diverse. Some of the outlier may be caused by mistakes of human input, device failure or noises in measurement, outdated old data or missing data. This kind of outlier data is usually considered to be erroneous and can be extracted from the dataset or transformed using other methods; another kind of outliers may be caused by changes in the data itself, reflecting distribution characteristics of the entire database to some extent, the data in such data sets are constantly changing with time. At certain time points, there will often be a large number or abnormal trading behavior to form outliers. Another possible reason for generating outlier data may be abnormal data sources, like the data generated when network invasion or disaster or disease happen, which are derived from abnormal behavior or state and have obvious differences from normal data. The outlier data of this type may contain very useful knowledge. It is a key research object for data analysis. Obviously, in-depth information of outliers should be analyzed to obtain interesting knowledge and rules that cannot be obtained from conventional data and patterns. How to accurately identify the local outlier data in the cloud database becomes an urgent problem.

Outlier data identification is accomplished in real, large-scale, noise-containing data. Outlier data recognition technology has been developed over a long period of time and has achieved relatively good research results. They can be roughly grouped into the following categories:

(1) Statistical-based recognition method. The statistical-based approach relies too much on whether the data set satisfies a certain distribution model, model parameters, or the number of outliers. However, the determination of these parameters is very difficult. In addition, this method is more suitable for univariate numerical data, and for most applications, the dataset is high-dimensional, so the statistical-based method is difficult to be applied in practice.

(2) Deviation-based recognition method. The deviation-based outlier recognition method can identify various types of data, but it is still necessary to understand main characteristics of the data. In real databases, it is difficult to identify main
characteristics of the data due to the large amount of data and attributes.

(3) Depth-based recognition method. The depth-based outlier identification method is also essentially a statistical method. The main idea is to first mark the data as point in the k-dimensional space, set a depth value for each data point according to the definition of depth, and then organize datasets based on this value. Data with smaller depth values are more likely to be outliers, so methods only need to detect low-level data. The depth-based recognition method can theoretically identify outliers in high-dimensional datasets, but in actual calculation, the efficiency is very low.

This paper aims at the low accuracy and low efficiency of traditional methods, starting from the idea of combining grid cells and density, and using the spatial distribution characteristics of grid cells in high-dimensional data, to propose a outlier data recognition method based on grid cells and density analysis. The MapReduce is used to program the model and parallelize the proposed method.

2 ACCURATE RECOGNITION OF LOCAL OUTLIERS IN CLOUD COMPUTING DATABASE

2.1 Local outlier recognition

Outlier data is a small amount of special data which obviously does not meet the general laws or patterns in a large number of datasets. These data may come from artificial measurement errors, machine execution errors, or occurrence of special events (Guo et al., 2016). In the daily transaction process, these very few outlier data have certain application value, because these data reflect some abnormal information. As in fraud detection, outliers may indicate fraud. Therefore, outlier data recognition can often find some real, but unexpected knowledge, and has been widely used in network intrusion detection, loan verification, credit card malicious overdraft and other fields.

The outliers are defined by $B(m, r)$. If the point have less than $m$ points in the range of radius $r$ is considered as outlier. But for some outliers, their neighbors are densely distributed on the edges. As shown in Figure 1, $O_1$ is an outlier, $O_2$ is not, $O_2$ is easy to be identified, but $O_1$ cannot be identified, which causes the covering of information. Covering means that the point is originally an outlier but fails to be identified. If we reduce $r$ in order to dig out $O_1$, then $O_2$ is identified as an outlier because $m$ does not reach the threshold, which leads to information submerging. Submerging refers to the normal data is judged as an outlier, so that the recognition accuracy of the outlier data is reduced.

![Fig.1. Edge distribution of neighbor points](image1)

![Fig. 2. Distribution of local outliers](image2)

In addition, there is also another type of outliers that are abnormal relative to their local neighbors, therefore these outliers are considered "local" outliers. As shown in Figure 2, the figure contains two clusters $C_1$, $C_2$ and two outliers $Q_1$, $Q_2$, where $C_2$ is dense, $C_1$ is sparse, $Q_2$ is a global outlier, and $Q_1$ is a local outlier.

2.2 Pruning of candidate outliers

Pruning refers to removing the point set that cannot be local outliers in advance, and analyzing the remaining candidate local outlier sets, and selecting the data points satisfying the conditions as local outliers. In order to describe the pruning strategy more clearly, the cloud computing database data structure class adds 3 attribute domains, node.pWigth, node.maxdnode and node.NNd in addition to the node identifier node.id, the data object node.obj, the neighbor node.nn of $K$, and the node.rad attribute domain. The node.pWigth represents the density of the data set object. In the candidate outlier set, each data object can be sorted according to the size of node.pWigth to determine whether it is a local outlier. Node.maxd reserves the maximum distance of the neighbors before the distance reaches the $K$ value. When the $K$ value is
reached, this value is assigned to node.rad; node.NNd is used to record the number of its neighbors and calculate the density. Depending on the size of node.NNd, each candidate local outlier is discussed in separate cases.

Referring to Figure 3, when examining the object \( S_i \), the candidate local outlier set is searched within a range where \( S_i \) is the center and \( r \) is the radius, and the distance \( d(S_i, S_w) \) is calculated for the returned arbitrary data object \( S_w \).

\[
\text{Fig.3. The pruning triangle inequality model}
\]

When 
\[
d(S_i, S_w) + \|S_w - S_i\| < r
\]
that is, 
\[
d(S_i, S_w) < r - \|S_w - S_i\| \Rightarrow S_i \text{ is a pruner, which is a pruning point. The pruning process only needs to calculate the distance between } S_i \text{ and } S_w \text{ without calculating the distance between } S_i \text{ and the neighbors of } S_w. \]

\[
d(S_i, S_w) + \|S_w - S_i\| \geq r \quad \text{if } d(S_i, S_w) < r \text{ is satisfied, the object to be investigated is added to the neighbors of the candidate set node object, and an equivalent cluster is constructed. As the number of neighbors increases, it may become a pruning point. According to the first pruning process, the reference point is the pruning point, so a part of the pruning point in the local outlier candidate set is kept. Therefore, the local outlier candidate set is re-scanned later, if } S_i, \text{rad} < r, \text{ rad, then pruning will be performed.}
\]

According to the pruning rule, the set of points in the cloud computing database that cannot become the set of local outliers is deleted in advance, and the remaining candidate local outlier sets are analyzed, and the data points satisfying the conditions are selected as local outliers, which provides the basis for accurate recognition of local outlier data.

2.3 Recognition method for local outliers

(1) Mesh generation

Mesh generation methods can be divided into two categories, namely the bottom-up mesh method and the top-down mesh method (Wang, Noda, 2017). The bottom-up mesh method divides the data space evenly into mesh cells of equal size based on the user-defined partitioning parameters. The mesh cells containing a certain number of data points are called high-density mesh cells. This method identifies all connected high-density mesh cells as clusters, and assumes that all data points falling within the same mesh cell belong to the same cluster, and each mesh cell only stores statistical information of data that falls within it. For example, the number of data points, the sum of data points, etc. The advantage is that the data can be compressed into a mesh data structure by scanning the data once, and an arbitrarily shaped cluster is found based on this mesh data structure. The accuracy of the cluster depends on the size of the mesh unit. If the mesh unit has a smaller granularity, and the accuracy is higher, but the computational complexity is greater. The bottom-up mesh method is not suitable for processing high-dimensional data. In high-dimensional space, the data distribution is very sparse, and the mesh method loses its compression effect, and the high density mesh cells belonging to the same cluster may not be connected, resulting in a failure to find a reasonable number of clusters.

In summary, when the partitioning parameters are selected properly, the bottom-up mesh method has high precision, but they cannot handle high-dimensional data. The top-down mesh method can handle high-dimensional data quickly, but with low precision. If we can combine the advantages of these two methods, a high-precision, high-efficiency mesh generation method can be obtained, the detailed analysis is carried out below.

This section formalizes the problem as follows. Let \( H = B_1 \times B_2 \times \cdots \times B_d \) be a \( d \)-dimensional data space in a cloud computing database, where \( B_1, B_2, \cdots, B_d \) is the \( d \) data dimensions of \( H \). The stream data is point \( W = W_1, W_2, \cdots, W_n \) in this \( d \)-dimensional space, each point is \( W_i = (w_{i1}, w_{i2}, \cdots, w_{id}) \). The \( j \)-th attribute value \( w_{ij} \) is the value of \( W_i \) in dimension \( B_j \), and \( w_{ij} \) is in the range of \( [\min_j, \max_j] \).

First, the data space \( V \) is evenly divided to obtain a mesh data structure \( G \). Each dimension of
$H$ is evenly divided into $c$ segments. Segments from each dimension intersect to form a mesh cell. The mesh cell $A_i = (a_{i1}, a_{i2}, \ldots, a_{id})$, where $a_y = [l_{ij}, h_{ij}]$ is a semi-open interval of dimension $B_j$, where $1 \leq \lambda \leq c$, $l_{ij}$ denotes the length of the mesh, and $h_{ij}$ denotes the width of mesh. The number of data points that fall into each mesh cell is recorded by the attribute D of the mesh cell, and D is also called the density of mesh cell. There are a total of $c^d$ mesh cells in the mesh $G$. When the dimension $d$ of the data space is large, $c^d$ is large. In order to be able to store the entire mesh data structure in memory, only the mesh cells of $count > 0$ are saved for meshing. This method can not only handle high-dimensional data, but also ensure accuracy.

(2) Local outlier recognition under mesh generation

The method proposed in this paper divides the data in the cloud computing database into disjoint small rectangular cell regions. Based on the density of these cell regions, the specified thresholds, and the data of neighboring cells, if the data is local outlier data is determined. The proposed method only scans the original data set once and performs efficiently.

Combination of the advantages of the two methods based on analysis and mesh together generates local outlier data recognition based on cell density. A cell is an important data structure in the proposed method. It is a triple $F(P', D, I)$. Among them, $P' = (p_1', p_2', \ldots, p_d')$ describes location of the cell; $D$ describes density of the cell, which is the number of data within the cell; $S$ is the data set within the cell. The proposed method used to identify the local outlier data includes two stages: rough screening and fine screening. The detailed process is as follows:

(1) Rough screening. First, $D$ (the number of data) is set to zero and $S$ is set to null for each cell. Then each data in the cloud database is mapped to a cell, $h(p = (p_1, p_2, \ldots, p_d)) \rightarrow F$. Among them:

$$U.D = U.D + 1$$
$$U.S = U.S + \{g\}$$

Again, each cell is processed as follows: if the cell is DU, $U.D > \xi$, then the data in the cell is not local outlier data; otherwise, the data in the cell is temporarily marked as local outlier data.

(2) Fine screening. Sparse units temporarily marked as local outliers in each roughing process are processed according to the following rules:

a. If the neighbors of the sparse cell $U$ are all empty cells, the data in the sparse cell is local outlier data.

b. If the sparse cell $U$ has non-empty neighboring cells, the deviation index of the data object is calculated for the data $g$ in the sparse cell. If the deviation index is less than the specified threshold, the data is not local outlier data. Otherwise, $g$ is local outlier data.

This algorithm only scans the original data once, and maps the data to the corresponding cell, and processes each cell to improve the I/O processing speed. Assuming that the number of data in the data set is $M$, the number of sparse cells is $\alpha$, the number of neighbors of the sparse cell is $\beta$, and the number of data in the sparse cell is $\delta$, then the time complexity of the algorithm is:

$$O_t = f(M + \alpha* \beta* \delta* \log (\alpha* \beta* \delta))$$

3 RESULTS AND ANALYSIS

3.1 Simulation data experiment

In order to verify the accuracy and efficiency of the proposed method, simulation data experiments and network intrusion data experiments are performed in turn.

The data set used in the simulation experiment is a cloud computing two-dimensional database generated by Matlab software. The database is scanned first for outliers recognition, and 8 clusters and 22 outliers are obtained. The environment of this experiment is Core i5 CPU 2.40 GHz, 2GB memory, operating system is Windows 7. The method is performed in Matlab environment.

All clusters are marked and outliers are labeled. The data distribution is shown in Figure 4.
After the 22 outliers shown in the above figure have been marked, outliers are identified using the proposed method. The recognition results obtained are shown in Table 1.

Table 1. Recognition results of local outliers by the proposed method

<table>
<thead>
<tr>
<th>Outlier data types</th>
<th>Recognition results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ordinary outliers</td>
<td>4;13;20;8;16</td>
</tr>
<tr>
<td>Local outliers</td>
<td>1;2;17</td>
</tr>
<tr>
<td>Strong outliers</td>
<td>3;5;14;6;7;11;12;15;18;19;21;22</td>
</tr>
<tr>
<td>Noise</td>
<td>9;10</td>
</tr>
</tbody>
</table>

According to the outlier properties of outliers, all 22 outliers labeled are divided into four categories: ordinary outliers, local outliers, strong outliers and noise. It can be seen from Figure 4 and Table 1 that although outliers 4, 5, and 14 are very close in space, their formation causes, that is, the outlier properties are different. The outlier data 4 is an outlier in the x dimension with respect to all clusters, that is, difference of the x attribute value of the outlier data 4 from others is the main reason becoming the outlier. The outlier data 5 and 14 are not outliers in one dimension, so the x and y attribute values jointly determine the 5 and 14 outliers. So in the final recognition results, 4 is the ordinary outlier, and 5 and 14 are strong outliers.

Similarly, the attribute values of the outlier data 11 and 20 are basically same on the Y coordinate axis, and they all fall in the projection of the cluster D. However, on the x axis, the projection of the outlier data 11 falls in the projection of the cluster A, and the outlier data 20 is the outlier on the x axis with respect to all the clusters. Therefore, the outlier data 20 is an ordinary outlier, and the outlier data 11 is a strong outlier.

The outlier data 8 and 16, although the two outliers are far apart in space, both are outliers in the Y dimension with respect to all clusters. That is, the outlier data 8 and 16 have the same reason that their Y attribute values are different from other conventional data. So the outlier 8 and 16 are all divided into ordinary outliers in the end.

According to the outlier recognition results, the local outliers are outliers from a certain cluster. In summary, the proposed method can accurately recognize the local outlier data in the cloud computing database.

### 3.2 Network intrusion data experiment

The experimental data selected in this section is a cloud computing database that contains multiple types of attack data and normal data. There are nearly 5 million connection records. Each connection record is marked as normal or a specific type of attack. For each TCP/IP connection provided, in addition to some basic attributes, some properties are also extended using domain knowledge. Some attributes are obtained based on the calculation of information within the past 2 seconds. The intrusion detection database contains four types of attacks: Denial of Service (DOS), User to Root (U2R), Remote to Local (R2L), and Probe (Probe).

1. **DOS attacks.** The attacker uses some means to exhaust resources and prevent cloud computing legitimate users from using the system or crashing the system. DOS attacks include many types: DOS attacks such as Mailbomb, Neptune, and Smurf are accomplished by abusing legitimate user features; DOS attacks such as Teardrop, PingofDeath construct malformed packets to block the TCP/IP stack from packet reconfiguration, so that normal connection cannot be completed; pache2, Back, Syslogd use a certain network daemon software vulnerability to achieve the attack.

2. **U2R attacks.** It refers to an attack that first obtains a certain account password through Sniffer, Dictionary, and other means, possesses or obtains access rights of the system common user, and further utilizes certain system vulnerabilities to obtain super user rights. The most common is a buffer overflow attack: stack overflow is caused by well-designed data, and the attacker can thus execute the command provided by the operating system; there are some U2R attacks that use some software carefully considering the issue of temporary file management; some U2R attacks obtain super user rights by attacking some local flawed suid programs and competition conditions.

3. **R2L attacks.** R2L refers to the behavior that an attacker who does not have a legitimate account
of an affected host sends packets to the host through the network, and then uses certain vulnerabilities of the system to obtain the user's access rights to the host. There are many ways in which a remote attacker can gain unauthorized access to a local account: some use buffer of network service software for overflows; other attacks such as Dictionary, ftp-write, Xsnoop, and guesses use misconfigured system security strategy.

(4) Probe attacks. Probe attacks detect the known system vulnerabilities or collect specific information by scanning computers in the cloud computing network. Probe is the first step for an attacker to launch an attack.

This section takes 2% of the data in the cloud computing database as test data. Each data object is a TCP data frame that occurs during the cloud computing network connection process. By preprocessing these TCP data frames, the currently described data objects that record various network connection activities are obtained. The important parameters for evaluating intrusion recognition are recognition rate, false positive rate, and recognition time. The recognition rate is the ratio of the actual number of recognized attacks to the number of attacks contained in the database. The false positive rate is the misidentification of normal behavior as an attack behavior, which is the ratio of the number of false positives to the number of normal behaviors. The ideal outlier recognition method should have high recognition rate, low false positive rate and low recognition time for intrusion detection, but in reality this is impossible. Relevant studies have shown that false positive rate increases with the increase of recognition rate. Simply increasing the recognition rate cannot effectively improve intrusion recognition performance. In the case of high recognition rate, false positive rate should be reduced as much as possible. In order to verify the effectiveness of the proposed method, the machine learning method, the statistical recognition method and the deep learning method are compared and tested. The results are described in Table 2.

<table>
<thead>
<tr>
<th>Type of attack</th>
<th>False positive rate/%</th>
<th>Recognition rate of the proposed method/%</th>
<th>Recognition rate of the statistical method/%</th>
<th>Recognition rate of the deep learning method/%</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOS</td>
<td>2.5</td>
<td>87.6</td>
<td>71.2</td>
<td>75.5</td>
</tr>
<tr>
<td>Probe</td>
<td>3.2</td>
<td>82.5</td>
<td>60.3</td>
<td>62.9</td>
</tr>
<tr>
<td>U2R</td>
<td>3.6</td>
<td>81.3</td>
<td>56.8</td>
<td>55.4</td>
</tr>
</tbody>
</table>

Through the above experimental results, we can know that the outlier data recognition method proposed in this paper can achieve a high recognition rate under a high false positive rate. In addition, recognition time of the proposed method is 16.9s, and recognition time of the statistical method is 25.3s, recognition time of the deep learning method is 29.2s, which verifies the accuracy and high efficiency of the proposed method.

4 CONCLUSION

The main research results of this paper are as follows:

(1) An outlier data recognition method based on mesh cell and density analysis is given. The method firstly divides each dimension of the cloud computing database and divides the mesh cell, and filters the outlier data and the normal data mesh cell. For mesh cells that contain both outlier data and normal data, density analysis methods are used to measure and determine outlier data, thereby further improving the accuracy of outlier data recognition. Finally, the feasibility of this method is verified by experiments.

(2) The MapReduce programming model is adopted to parallelize the proposed method. First, three pairs of Map and Reduce functions are used to count the maximum and minimum values of each dimension, and the number of objects in each cell and neighbor cells is counted and some outlier data is selected; then, the density values of candidate outliers are calculated; finally, a parallel outlier recognition method based on mesh cell and density analysis under MapReduce is implemented. Experiments verify the accuracy and efficiency of the proposed method.

REFERENCES


