Map-Reduce Algorithm for Mining Outliers in the Large Data Sets using Twister Programming Model

Subramanyam. RBV, Sonam. Gupta

Abstract—An important problem that appears often when analyzing data involves identifying irregular or abnormal data points called outliers. For example, in credit card transaction data, outliers might indicate potential fraud; in network traffic data, outliers might represent potential intrusion attempts. Outlier mining is the most important data mining task whose goal is to find the observation which is dissimilar from the remaining set of data set. Traditional outlier detection method assumes data is centralized at single location but this assumption is not true in today’s scenario because data set size is increasing day to day. Moreover new requirement of these methods must be applicable to data which might be distributed among different locations. Design of efficient parallel algorithms and frameworks are the key to meeting the scalability and performance requirements.

Map reduce is a framework for processing large data sets with a parallel, distributed algorithm on a cluster. Iterative Map reduce programming model has simplified the implementations of many distributed data mining applications. In this work we design and realize a parallel Outlier mining algorithm based on iterative Map Reduce framework. This algorithm uses Twister programming model which is a light weight map reduce runtime.

Keywords— Outlier mining, Iterative map Reduce, Twister Model, data mining, MRdriver.

I. INTRODUCTION

The need for data mining, i.e. extracting knowledge from the data in the form of useful and interesting models and trends, is of greater importance today than ever before, due to the massive amount of data existing in databases all over the world. There are various functionalities of data mining i.e. Association analysis, Classification, Prediction, Clustering and Outlier analysis.

A loose definition of outlier mining is “It is the important data mining task whose goal is to isolate the observation which is dissimilar from the remaining set of data set”. Outlier mining has wide applications; it can be used in fraud detection for example, by detecting unusual usage of credit card or telecommunication services. In addition it is useful in customized marketing for identifying the spending behavior of customers with extremely high incomes or low incomes or in medical analysis for funding unusual responses to various medical treatments.

“Outliers are data elements that cannot be grouped in a given class or cluster. Also known as exceptions or surprises, they are often very important to identify. While outliers can be considered noise and discarded in some applications, they can reveal important knowledge in other domains, and thus can be very significant and their analysis valuable”.

Outlier mining can be described as follows: Given a set of n data points or objects and k the expected number of outliers, find the top k objects that are considerably dissimilar from the remaining set of data [2].

Distance based outlier mining is one of the most important unsupervised learning method. Distance based outliers as those objects that do not have enough neighbors from the remaining set of data. The most popular distance measure is Euclidean distance. According to the Euclidean distance formula, the distance between two points in the plane with coordinates (x, y) and (a, b) is given by

\[ \text{dist}(x, y, (a, b)) = \sqrt{(x - a)^2 + (y - b)^2} \]

Important challenges faced by outlier detection methods and addressed in this dissertation include the following

(a) Large number of data sets - Currently, enormous amounts of data are created every day. With the rapid expansion of data, we are moving from the Terabyte to the Exabyte Age. At the same time, new technologies required to make it possible to organize and analyze the massive amounts of data currently being generated. Especially, theoretical and practical aspects of extracting knowledge from massive data sets have become quite important. Consequently, large-scale data mining has attracted tremendous interest in data mining community. One of the alternatives emerged for dealing with massive data sets that is map reduces based approach.

(b) Distributed data sets - A second issue is that data may be distributed among different sites belonging to the same or different organizations. Moreover, each of these sites might contain large amounts of data. This is an issue that has to be...
Outlier detection algorithms. Section 3 provides the problem data, the outlier detection algorithms must minimize communication overhead and synchronization overhead between the different sites, in which the data reside, as well as the scans over the data.

In Today's world many prominent data mining algorithm designed based on the map reduce framework. Map Reduce programming model is the distributed programming model which has simplified the implementations of many data parallel applications. The simplicity of the programming model and the quality of services provided by many implementations of Map Reduce attract a lot of enthusiasm among parallel computing communities.

In this dissertation, we propose algorithms to address the challenges outlined above due to current data sets. Specifically, our contributions are as follows:

(a) We propose a fast and scalable distributed solving set algorithm that is based on Twister's iterative map reduce framework. Twister that is light weight iterative map reduce run time we used in distance based outlier mining algorithm. This algorithm automatically recovers from faults when fault Tolerance is enabled.

The organization of this dissertation is as follows the next chapter (Section 2) provides a thorough literature review of Outlier Detection algorithms. Section 3 provides the problem formulation of this paper. Section 4 is the design and implementation part of outlier analysis with the uses of map reduce strategy. Section 5 gives the experimental result of the algorithm. Section 6 provides the conclusion and future work.

II. RELATED WORK

In this section, we provide an overview of outlier mining methods is given, though different approaches have been proposed.

Outlier mining basically deals with individual objects such that each object consists of dissimilar or unrelated objects. Outlier mining has been studied in the last decades and numerous approaches have been developed. These approaches mainly classified into two categories one is classic outlier approach that is based on transaction data set which can be grouped into statistical-based approach, distance based approach, deviation based approach, and density based approach. The second one is spatial outlier approaches analyze outliers based on spatial data set which can be grouped into space based approach and graph based approach.

Distance-based Approach The concept of distance-based outlier relies on the notion of the neighborhood of a point, typically, the k nearest neighbors, and has been first introduced by Knorr and Ng [6]. Distance-based outliers are those points for which there are less than k points within the distance in the input data set. This definition does not provide a ranking of outliers and needs to determine an appropriate value of the parameter. Ramaswamy et al. [18] modified the definition of outlier introduced by Knorr and Ng and consider as outliers the top n point’s p whose distance to their k-th nearest neighbor is greatest.

Prior work on outlier detection using k-nearest neighbor’s algorithm

The concept of distance based outlier with k-nearest neighbors has been first introduced by Knorr and Ng [9][12]. Knorr and Ng present two algorithms, the first one is a nested loop algorithm that runs in O(dN^2) time, while the second one is a cell based algorithm that is linear with respect to N, the number of points of the, data set but exponential in d, the number of dimensions of the data sets. This last method is fast only if d ≤ 4. On the other hand, the nested loop approach is impractical when outliers in large data sets have to be mined. Thus, efforts for developing efficient algorithms that scale to large real data set, have been recently made.

Then after Ramaswamy et al. [23] present new algorithm that is partition based algorithm that, first, partitions the input points using a clustering algorithm and, then, prunes those partitions that cannot contain outliers. The experiments, up to 10 dimensions, show that the method scales well with respect to both data set size and dimensionality. This definition, however, does not take into account the information contained in the k-neighborhood of a point and, thus, it could not properly distinguish between a dense or sparse neighborhood.

Prior work on parallel and distributed outlier detection algorithm

The outlier detection task is very time consuming because one person’s noise can be another person’s signal. Now there has been increasing interest in parallel, distributed and scalable algorithm for outlier detection.

The outlier detection task can be very time consuming and recently there has been an increasing interest in parallel/ distributed methods for outlier detection. Hung and Cheung [9] presented a parallel version, called PENL, of the basic NL algorithm [10]. PENL is based on a definition of outlier employed in a distance-based outlier is a point for which less than k points lie within the distance in the input data set. This definition does not provide a ranking of outliers and needs to determine an appropriate value of the parameter. Moreover, PENL is not suitable for distributed mining, because it requires that the whole data set is transferred among all the network nodes.

Lozano and Acun’a [11] proposed a parallel version of Bay’s algorithm [12], which is based on a definition of distance-based outlier coherent with the one used here. However, the
method did not scale well in two out of the four experiments presented. Moreover, this parallel version does not deal with the drawbacks of the centralized version in [12], which is sensitive to the order and to the distribution of the data set. Otey et al. in [13] and Koufakou and Georgiopoulos proposed their strategies for distributed high-dimensional data sets. These methods are based on definitions of outlier which are completely different from the definition employed here, in that they are based on the concept of support, rather than on the use of distances.

Dutta et al. proposed algorithms for the distributed computation of principal components and top-k outlier detection. In their approach, outliers are objects that deviate from the correlation structure of the data: A top-k outlier is an object having at most the kth largest sum of squared values in a fixed number of the lowest order principal components, where each component is normalized to its deviation. This definition neither implies nor is implied by the definition employed in this work. For example, if all clusters are located far from the mean of the data set, distance-based outliers close to the mean are not necessarily exceptional in the correlation structure. On the other hand, objects having large values in the first principal components need not have smaller weight than objects which deviate from the correlation structure in the low-order components.

III. PROBLEM FORMULATION

In this section, we formally give the definition of the problems that will be treated. We are given a large amount of data sets called objects and each object is represented by set of measurements called attributes or features and constraint is that data sets are distributed in different location.

Our main goal is to find abnormal data points in this large data set with map reduce strategy According to the input of the data set, we perform the algorithm in iterative way that is different from traditional map reduce. In each iteration step the set of n pairs of outliers is updated.

IV. ALGORITHMS

In this section we describe algorithm named map reduce strategy for mining outliers in the large data using Twister iterative map reduce programming model. This algorithm is based on distributed solving set algorithm[1].First of all we define the mapper, reducer and combiner function of this algorithm which is based on Twister iterative map reduce computation.

4.1 Outlier Mining using iterative map reduce

The first step in designing the Map Reduce routines for Outlier mining is to define and handle the input and output of the implementation. In Map Reduce input is given as a <key, value> pair, in algorithm where ‘key’ is the Global candidate set and ‘value’ is the serializable implementation of data set.

For implementation of Map Reduce routine two files are necessary one is initial global candidate set file and another is local data set files for mappers. In the configuration step data is distributed to mappers. Once the configuration step is defined then calls Mapper routine then after Reducer routine then combiner routine. Fig.1 shows the Map Reduce routine for outlier analysis this is based on Twister iterative map reduce programming model.

Various stages of outlier mining algorithm.

4.2. Initialization stage

The first step of outlier mining – Map Reduce is to start the Map reduce workers and configure both the map and reduce tasks using configuration method. This is used for the configuring once and used many times approach with the aim of supporting iterative map reduce computation efficiently. The configuration step, which occurs only once, can be used to load any static data necessary for the map tasks.

![Fig 1 Map Reduce routine for mining outlier analysis](image)

4.3 Map stage

After the configuration step is over, the user program start the MRDriver to start the map reduce computation by passing the key, value pairs to the map task. In that each map function gets the variable data i.e. global candidate set as a key, value pairs that is calculated during the previous iteration and used as the input value for mapper function and load static data set.

All the map function gets the same selected global candidate at each iteration and computes local nearest neighbors to all global candidates with respect to local data set. And for next iteration map function will select local candidate set and pass local nearest neighbors and local candidate set to Reducer function.

Compute Local nearest neighbors- The weight of the objects in local candidate set are computed by comparing each Object in Ci with each object in the local data set. This operation is split into three steps in order to avoid duplicate distance computation. The first double cycle compares each
object in Ci with all other objects in Ci and updates the heap. The second double cycle compares the objects of Ci with the other objects of C. And finally third double cycle compares the objects of Di with the object of C. See third double cycle algorithm [3].

**Algorithm to Update LNNC**

```java
for each p in Di{
    for each q in C{
        if max(Sum(NN[p]), Sum(LNNC[q])) > minOut{
            Dist = dist(p,q);
            UpdateMin(NN[p], <q, Dist>);
            UpdateMin(LNNC[p], <q, Dist>);
        }
    }
}
```

**Mapper Routine for Outlier analysis**

**Algorithm 1: Mapper Design for Outlier mining algorithm**

Input: Local data set Di, Global candidate set C, Local candidate set Ci, minOut.

Output: Local nearest neighbor's w.r.t each global candidate set and for next iteration global candidate.

Procedure OutlierMiningMapDesign:
1. Load global candidate set file C
2. Remove Global candidate in local data set
3. Create two Hash map LNNC, LG
4. Calculate LNNC
5. Update LG
6. Calculate active number of objects
7. Call OutlierMiningReduce(LNNC, LG)

**Fig.2 Mapper design for Outlier analysis**

**4.3 Reduce function**

Reduce workers are initialized in the same manner as the map works. Once initialized the reduce workers wait for the map output. MRDriver instruct the reduce workers to start the reduce task when all map tasks are completed. Here, Reduce function computes the global nearest neighbors with respect to full data set. Reduce function also generate the union of all local candidate set i.e. called global candidate set. Reduce function write the result into output directory and then main program will call the combine function if it’s necessary.

**Reducer Routine for Outlier analysis**

**Algorithm 2: Reducer Design for Outlier mining algorithm**

Input: Local candidate sets, Local nearest neighbors w.r.t each global candidate set

Output: Global nearest neighbors w.r.t each global candidate set and for next iteration union of all local candidate set i.e. called Global candidate.

Procedure OutlierMiningReduceDesign:
1. Collect LNNC w.r.t each mapper
2. Calculate GNCC
3. Union of LG
4. WRITE result into output directory

**Fig.3 Reducer Design for Outlier analysis**

**4.4 Combine function**

Once the user program receives all the outputs of the reduce computation, it may perform a combine operation. This step can be used to combine the result of the reduce task to produce the final result or if iteration is there then this step is used to compute the deciding value to continue the iterations. Here, Combine function takes the global nearest neighbors and global candidate set. Combine function will update the heap of n-pairs i.e. OUT.OUT is the heap of n-pairs <p, w> where p an object and w is the associated true weight i.e. calculated by the sum of k-nearest neighbors. Then combiner calculates the min out value i.e. the lower bound to the weight of the top-n outliers. Minout value I is used to decide whether the objects are active or non-active.

**Combine routine for outlier analysis**

**Algorithm 3: Combine function for Outlier mining algorithm**

Input: Global nearest neighbors and set of Global candidates

Output: Global candidate set, minOut and heap of n-pairs of outliers.

Procedure OutlierMiningCombineDesign:
1. LOAD OUT
2. Update OUT w.r.t GNCC
3. Calculate minOut i.e. Min(OUT)
4. Send OUT, global candidate set and minOut to main program

**Fig.4 Combine function for outlier analysis**

**4.5 Main program with termination stage**

Main program will get the output from the combine function and check whether to continue the iteration or not. This is decided by the set of global candidate if global candidate set is not null then continue the iteration.

In termination stage user program informs the MRDriver to complete the map reduce computation. MRDriver terminates the set of workers used for Map Reduce computation and output is in the output directory.
5. EXPERIMENTAL RESULT

5.1 Experimental Data Set

We empirically evaluated the mining algorithm by taking real life data sets from Uci repository. The experiment result indicate that our algorithm can effectively identify the outliers in the large data sets.

Dataset1 Our experiment is conducted on the Poker hand data sets. Poker is obtained from the real data set Poker Hands available at UCI repository, by removing the class labels then Poker consist of 1,000,000 instances of 10 attributes. This study aims to identify outliers from these records. For the convenience of calculation, we have randomly selected 80,000 from the data set and removed their redundant attributes, only leaving ten attributes.

Dataset2 KEGG metabolic reaction network real data set downloaded from Uci repository contains Multivariate text. Data set contains 65,554 instances of 29 attributes after removing some attributes we have taken 10 attributes with 64,000 instances.

Dataset3 Magic gamma Telescope real data set downloaded from Uci repository contains Multivariate real data set, number of instances is 19020 and number of attributes is 11. We have randomly selected 16000 instances contains 11 attributes.

5.2 Experimental setup

In order to verify the performance of the algorithm proposed in this paper, we have implemented the algorithm. The algorithm are compiled with jdk 7 and its running environment is Twister programming model that is map reduce programming paradigms in Ubuntu.

5.3 Result

Analysis 1 - Now, we present the result of the experiments. We have performed analysis of the algorithm w.r.t parameters n, k and m. Firstly by varying parameters n and k with m =80 taken as constant. As for the varying parameters n, with k = 10 algorithms time is directly proportional to n.

As an example, the table below reports the execution time on the Poker hand, KEGG metabolic reaction network and MAGIC gamma Telescope data sets for different parameters of n, k and m with number of mappers is 8.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>n = 10</th>
<th>k = 25</th>
<th>k = 50</th>
<th>n = 10</th>
<th>n = 25</th>
<th>n = 50</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data set 1</td>
<td>43.45</td>
<td>50.072</td>
<td>70.035</td>
<td>50.072</td>
<td>71.34</td>
<td>80.56</td>
</tr>
<tr>
<td>Data set 2</td>
<td>40.34</td>
<td>48.56</td>
<td>65.456</td>
<td>49.567</td>
<td>67.56</td>
<td>73.34</td>
</tr>
<tr>
<td>Data set 3</td>
<td>19.56</td>
<td>25.45</td>
<td>32.45</td>
<td>28.46</td>
<td>39.45</td>
<td>45.34</td>
</tr>
</tbody>
</table>

Now, varying parameter is m with some n and k values randomly selected. We have seen that it is also directly proportional to the execution time. This table suggest us to set m to a reasonably small value as so we have selected m = 80, that is the value used in our experiments.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>m = 80</th>
<th>m = 240</th>
<th>m = 800</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data set 1</td>
<td>n=k=10</td>
<td>n=k=10</td>
<td>n=200 k=50</td>
</tr>
<tr>
<td></td>
<td>56.762</td>
<td>57.024</td>
<td>111.937</td>
</tr>
<tr>
<td>Data set 2</td>
<td>28.46</td>
<td>37.323</td>
<td>58.34</td>
</tr>
<tr>
<td>Data set 3</td>
<td>49.567</td>
<td>55.34</td>
<td>107.34</td>
</tr>
</tbody>
</table>

Analysis 2 For this analysis we have taken data set1 i.e. poker hand data set with n =10, m=80, k = 10 Fig.6 shows the nth outlier weight with respect to each iteration with 10 iterations considered. It is clear that the value of nth outlier weight increase as the iteration increases.

Analysis 3 Same data set as analysis 2. Fig.7. shows active number of object w.r.t each iteration with 10 iterations. As the iteration increases the active number of object decreases and reaches to zero at last iteration.
We presented the Map Reduce based outlier mining algorithm that is based on distance based outlier detection solving set [3] to compute the outliers.

The volume of information increases day to day in today's world requires huge quantity of data processing. In this paper we introduce the implementation of Outlier mining algorithm over a Map Reduce network. This algorithm provides a robust and efficient system and also reduces the implementation costs of processing such huge volume of data.

Map Reduce infrastructure has provided data mining researchers with a simple programming interface for parallel scaling up of many data mining algorithms on large data sets. This certainly makes it possible to mine large data sets without the constraints of memory limit. This algorithm also solves the problem of scalability to support processing of high volume of data sets.

Parallel implementation of outlier mining Algorithm is an area of research that is aimed to improve efficiency, accuracy, scalability and fault tolerance and research work is going on in this area.

REFERENCES

[2] Jiawei Han and Micheline Kamber , " Data Mining : concepts and technique" 2nd edition.
[5] Jaliya Ekanayake1,2, Hui Li1,2, Bingbing Zhang1,2, Thilina Gunaratne1,2, Seung-Hee Bae1,2, Judy Qin2, Geoffrey Fox1,2 “ Twister : A Runtime for iterative MapReduce ” ACM conference 1-58113 2010

Dr RBV Subramanyam obtained his M.Tech. and Ph.D. degrees from Indian Institute of Technology Kharagpur (IIT) and currently faculty in Department of Computer Science & Engineering, National Institute of Technology Warangal, Telangana State, India. He is member of IEEE, IE (India). His research interests include Data Mining, Fuzzy Systems, Soft Computing, and Big Data Analytics. Ms. Sonam Gupta was student of M.Tech. (Computer Science & Engineering) during 2012-14 at Department of Computer Science & Engineering, National Institute of Technology Warangal. Her research interests are Data Mining, Big Data Analytics.