Outlier Detection for Large Datasets

Kamlesh Kumar & Bidyut Kr. Patra

Department of Computer Science & Engineering DDIT Unnao, NIT Rourkela India.
E-mail : kamlesh4u2008@gmail.com

Abstract – **Finding outlier or anomaly in large dataset is an important problem in areas such as electronic commerce, credit card fraud, and even the analysis of performance statistics of professional athletes. The identification of outliers can lead to the discovery of truly unexpected knowledge. LOF (*local outlier factor*) is a classical density based outlier detection method, which is successfully used for detecting outliers in fields of machine learning, pattern recognition, and data mining. LOF has two steps. In the first step, it calculates k-nearest neighbors of each data point. In the next step, it assigns a score to each point called *local outlier factor (LOF)* using k-nearest neighbors information. However, LOF computes a large number of distance computations to calculate k-nearest neighbors of each point in the dataset. Therefore, it cannot be applied to large datasets. In this paper, an approach called TI-LOF is proposed to reduce the number of distance computations in classical LOF method. TI-LOF utilizes triangle inequality based indexing scheme to find k-nearest neighbors of each point. In the same line of classical LOF, TI-LOF assigns a score to each point using earlier computed information. Proposed approach performs significantly less number of distance computations compared to the classical LOF. We perform experiments with synthetic and real world datasets to show the effectiveness of our proposed method in large datasets.**

Keywords - Outlier Detection, Database Mining, triangle inequalities.

I. INTRODUCTION

Larger and larger amounts of data are collected and stored in data bases. This increases need of efficient and effective analysis methods to make use of the information contained implicitly in the data. *Knowledge discovery in databases* (KDD) has been defined as the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable knowledge from the data. [1]

Most studies in KDD focus on finding common patterns (association mining), grouping patterns (clustering), etc. However, for applications such as detecting criminal activities of various kinds (e.g. in electronic commerce), rare events, deviations from the majority, or exceptional cases may be more interesting and useful than the common cases [3].

"The data objects that do not comply with the general behavior or model of the data" and such data objects, which are grossly different from or inconsistent with the remaining set of data, are called outliers"[4].

A new method for finding outliers in a multidimensional dataset is a local outlier factor (*LOF*) for each object in the dataset, indicating its degree of outlier-ness. The developed concept of an outlier is quantifies how outlying an object is.LOF quantifies for an object that how much its value deviates from standard LOF value 1. The outlier factor is local in the sense that only a restricted neighborhood of each object is taken into account. Our approach is loosely related to density-based clustering. The remainder of this paper is organized as follows: we discuss related work in Section II. In Section III, We conduct extensive experiments and evaluation in Section IV and make concluding remarks in Section V.

II. RELATED WORK

Based on our limited survey, nearest neighbor techniques have been developed. This type of techniques has been used for finding outlier algorithm. Some of the techniques are reported in the section in brief. In this paper outliers are finding by two methods. One is conventional LOF, that nearest-neighbor was find by using k-nearest neighbor method which is expensive. Expensive in the sense of distance computational and time taken. By this method distance is compute for one point to all point in the dataset if there is *n* query point, then computational and time taken becomes more. So, in that way another method for finding nearest neighbor is called k-nearest neighbor by using triangle inequality. This method is efficient than conventional k-nearest neighbor speed-up and reduces computations.

For many KDD applications, such as detecting criminal activities in E-commerce, finding the rare instances or the outliers, can be more interesting than finding the common patterns. In outlier detection regards being an outlier as a binary property. It is more meaningful to assign to each object a *degree* of being an outlier. This degree is called the *Local Outlier Factor* (LOF) of an object. It is *local* in that the degree depends on how isolated the object is with respect to the surrounding neighborhood [8].

Problems of basic k-nearest neighbor LOF algorithm

A drawback of the basic nearest neighbor based LOF technique is the large dimension and large dataset. Time taken is more due to distance computation for each point to all point of dataset. The conventional LOF such techniques do not scale well as the number of attributes increases and also when increases number of instances. Several techniques have directly optimized the outlier detection technique under the assumption that only top few outliers are interesting. If an outlier score is required for every test instance, such techniques are not applicable. Hence it is not well suited for large number of attributes. In basic k-nearest neighbor time complexity is more because for each pattern in dataset finds the distance to all pattern. Conventional LOF have main problem is high dimension and large instances of dataset. Due to high dimensions, large instance its efficiency is decreases and becomes more time taken. Due to this problem VA-file is used.VA-file performance is better when dataset is not so large. TI-neighborhood method performs well rather than VA-file for large dataset [9].

SOME NOTATION

Distance between two points p and q is denoted as distance(p,q). This can be use as variety of distance metrics. Depending on an application, one metric may be more suitable than the other. In particular, if Euclidean distance is used, a neighborhood of a point has a spherical shape; when Manhattan distance is used, the shape is rectangular.

Positivity

distance(p, q)$\geq$ 0 for all p and q,

distance(p, q)=0 only if p=q.

Symmetry

distance(p, q)= distance(p, q) for all p and q.

Triangle Inequality

distance(p,r)$\leq$ $\mathrm{distance(p,q)} + \mathrm{distance(q,r)}$ for all point p, q and r. Measures that satisfy all three properties are known as *metrics.*

Proximity measurement quantifies the similarity or dissimilarity (in terms of distance) between two data objects. The following notation has been used in the description of the metrics: p and q denote points in n-dimensional space, and the components of the points are p_1, p_2,…, p_n and q_1, q_2, …, q_n. There are various methods of measuring the distance quantifiers among spatial data.

distance(p, q) = $\boldsymbol{(\sum_{k=1}^{n}|p_k - q_k|)1/r}$

Where, r is a parameter, the following are the three most common examples of Minkowski distances

- r = 1. City block (Manhattan, taxicab, L1 norm) distance. A common example is the Hamming distance.
- r = 2. Euclidean distance (L2 norm).
- r = ∞. Supremum (Lmax or L∞) distance.

Euclidean distance, have some well-known properties. If distance(p, q) is the distance between two points, p and q, then the following properties hold.

Eps-neighborhood of a point p (denoted by NEps(p)) [12]

It is defined as the set of points q in dataset D. That are different from p and distant from p by no more than Eps, where, *Eps* is the *radius* at point p and Eps$\geq$ distance(p, q) that is,

NEps(p) = {qϵD | q $\neq$p $\wedge$ distance(p, q) $\leq$Eps}.

Let p be a point in D. The set of all points in D that are different from p and closer to p than q will be denoted by CloserN(p, q) that is,

CloserN(p, q) = {q'ϵD| q'$\neq$p $\wedge$ distance(q', p) < distance(q, p)}.

Clearly,

Closer(p, p) = $\emptyset$

k-neighborhood of a point p (kNB(p))

It is defined as the set of all points q in D such that the cardinality of the set of points different from p that are closer to p than q does not exceed k-1, that is,

kNB(p) = {q ϵD| q $\neq$p $\wedge$ |CloserN(p, q)| $\leq$ k-1}.

Note that for each point p, one may determine a value of parameter Eps in such a way that NEps(p) = kNB(p). In particular, NEps(p) = kNB(p) for Eps = distance(q, p), where q is most distant neighbor of point p in kNB(p).

III. METHODOLOGY

A. Basic k-nearest neighbor LOF

Nearest neighbor based techniques require a distance or similarity measure defined between two data instances. Distance (or similarity) between two data instances can be computed in different ways. For continuous attributes, Euclidean distance is a popular choice but other measures can be used.

Distance (p, q) = $\sqrt{\sum_{k=1}^{n}{}_{i}\ (p_k - q_k)2}$

Where, n is the number of dimensions and p_k and q_k are, respectively, the k_{th} attributes.

The distance of each point p in data set D until last point is not found, for each dataset point (D,p). Choose k-point from distance (D, p), and find max value of them k-point. Check for rest (N-k) points. If max distance value is greater than the rest (N-k) point distance, if it happen then change their position among them. Again find max among k-point then checking process continue until k-point have not minimum distance than the (N-k) points. Continue this process for all point p in data set D [7].

The k-point will be the k_{th} – nearest neighbor.

For finding outliers firstly we have to find the reachable distance of point p to the k_{th} neighbor point o1, o2…ok. If p is the neighbor of k_{th} point (o1, o2,….ok) then reachable distance will be k-distance. Otherwise will be actual distance of point. Equation 1 is finding *lrd* is as follow:

$$lrd(p) = \frac{MinPts}{\sum_{p} reach\_dist_{0MinPts}(p,o)}$$

For each p in D Reach-dist(p)=max(k-distance(p),distance(p,o)). So *lrd* is calculated by above equation.

Reverse of summing all reachable distance of point p w.r.t. k_{th} neighbor and dividing k_{th} point then we find *lrd*(local reachable distance) of point p. Find *lrd* of each k_{th} neighbors and summing, after that dividing by *lrd* of p point. Then again dividing by no. of k_{th} neighbor and then find the LOF value of point p. Equation 2 is finding *LOF* is as follow for point p:

$$LOF(p) = \frac{1}{MinPts} \cdot \sum_{p} \frac{lrd(o)}{lrd(p)}$$

For each p in D, LOF is calculated by the above equation [2] [5] [8].

B. Nearest Neighbor method by using Triangle Inequality LOF

Density based method is classical method in data mining. The bottleneck for k-nearest neighbor algorithms is high dimensional data. In density based method which requires the calculation of a neighborhood within a given radius *Eps* (*Eps-neighborhood*) for each data point. A new solution was based on properties of the triangle property. In this algorithm, examine a problem of an efficient calculation of the set of all *k* nearest neighboring points (*k-neighborhood*) for each data point. In the new task, the value of a neighborhood radius is not restricted. The theoretical solution is also based on properties of the triangle inequality. Based on this solution, a new TI-k-Neighborhood-Index algorithm that calculates *k*-neighborhoods for all points in a given data set is discussed here. The usefulness of this method is verified by using it in the NBC (Neighborhood Based Clustering) clustering algorithm [4].

Some property of triangle inequality is as follow and by using property can say something about the point which is neighbor or not.

Property 1 : (Triangle inequality property) for any three point's p, q, and r:

Distance (p, r)$\leq$ $\mathrm{distance}(p, q) + \mathrm{distance}(q, r)$

Property 2 : bove property is equivalent to

Distance (p, q) ≥ distance (p, r) - distance (q, r).

The property 2 says that, if we know the distance(p, r) and distance(q, r) from a reference point r then it conclude that distance(p, q) can be find without calculating the actual distance of point p to point q [11].

Let D be a set of points. For any two point's p, q in D and any point r:

From the above properties we find one conclusion that distance (p, r) - distance (q, r) > Eps ⤇q∉N Eps(p) ∧p ∉N Eps(q).

If the difference of distance(p, r) and distance(q, r) is greater than Eps(radius) then we can say p will not be neighbor of q and, q will not be neighbor of p without calculating the distance of p to q. it is short method for finding the neighbor.

The algorithm starts with calculating the distance for each point in D (dataset) to a reference point r, e.q. to the point with all coordinates equal to 0. The points are sorted w.r.t. their distance to r. For each point p in D (point sorted according to distance), the function identifies first those k points q following and preceding point p in D for which the difference between distance (p, r) and distance (q, r) is smallest. These points are considered as candidates for k nearest neighbors of p. Radius Eps is calculated as the maximum of the real distances of these k-closest relative points to p. It is guaranteed that real k-nearest neighbors lie within this radius from point p. The remaining points preceding and following point p in D (starting from points closer to p in the ordered set D) are checked as potential k nearest neighbors of p until the conditions specified are fulfilled. If so, no other points in D are checked as they are guaranteed not to belong to k-Neighborhood Based (p). The remaining points preceding and following point p in D for which the difference between distance (p, r) and distance (q, r) is less than Eps find actual distance of that point. Sort that actual distance in non decreasing order and top k-point will be k-nearest neighbor of point p. [12] For each p in D Reach-dist(p)=max(k-distance(p),distance(p,o)). So *lrd* is calculated by equation 1. After finding the lrd, we have to find LOF as reverse of summing all reachable distance of point p w.r.t. k_{th} neighbor and dividing k_{th} point then we find *lrd*(local reachable distance) of point p. Find *lrd* of each k_{th} neighbors and summing, after that dividing by *lrd* of p point. Then again dividing by no. of k_{th} neighbor and then find the LOF value of point p. [8]

IV. EXPERIMENTAL RESULTS

We present our experiments to evaluate the effectiveness of TI-LOF method. In this chapter we have used two different dataset and run on machine.

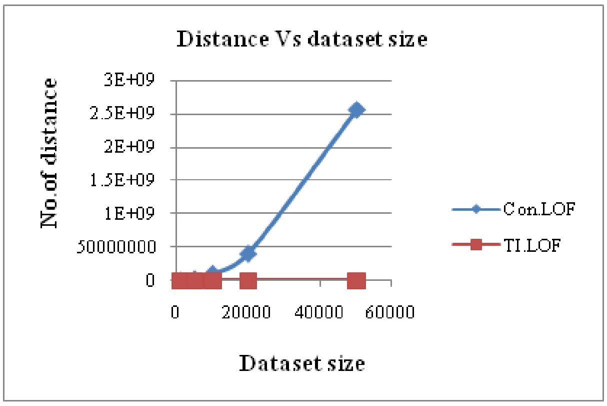
1. *Uni_2 Data set (Synthetic)*

This data set is synthetic dataset. In this data set two clusters and four outliers are present. This dataset is successfully implemented. Uni_2 data set (synthetic) contains number of instances 3139, number of attributes 2. Attribute information are numerical and missing attributes Values is none.

In this section we report conventional LOF and TI-LOF by using different datasets. We used multidimensional dataset for large pattern for different dataset.

No. of distance Vs k-distance Uni_2 dataset, Pattern=3139, dimension=2

S.N.	K	(No. of distance) Con.LOF	(No. of distance) TI -LOF	Extra Distance
1	10	9855413	131555	9723858
2	15	9856885	193695	9663190
3	20	9858204	255649	9602555
4	25	9859372	317249	9542123
5	30	9860638	378667	9481971



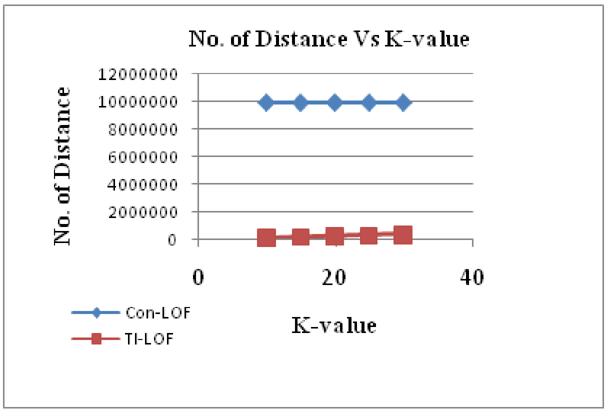
The above graph when the value of k is increasing then LOF distance is increasing almost 75 times than the TI-LOF. So by the graph we can say that distance computation will be less in newly method (TI- LOF).

2. *Shuttle Data set*

SHUTTLE Dataset (Stat Log Version), Dataset is multivariate, introduced by Jason Catlett of Basser Department of Computer Science, University of Sydney, N.S.W., and Australia. These data have been taken from the UCI Repository of Machine Learning Databases at Number of instances 50556, attributes 9, attribute information the shuttle dataset contains 9 attributes all of which are numerical and missing attributes values are none.

Distances Vs dataset size for shuttle dataset at k=20

S.N.	Data size	(No. of distance) Con.LOF	(No. of distance) TI -LOF	Extra distance
1	1000	1005251	29103	976148
2	5000	25026694	144631	24882063
3	10000	100055197	273228	99781969
4	20000	400112189	607857	399504332
5	50556	557216927	1534654	2555682273



The above graph is different dataset size Vs no. of distance is calculated. The average distance of Con.LOF is multiple of almost 580 times greater than TI-LOF.

V. CONCLUSIONS AND FUTURE WORK

In this paper, LOF is a classical density based outlier detection method. However, it is not scalable with the large size of the dataset. In this paper, a speeding up approach is proposed to reduce the number of distance computation in LOF method. Our proposed method is effective in large dataset. The primary advantage of our method is that its distance computation is very less than existing. It is faster method for detecting outliers for large dataset. Due to less computation, time has been taken less. So it is faster than Conventional LOF.

VI. FUTURE WORK

LDOF (local distance outlier factor) is not suitable for finding outliers for large dataset. It finds outlier globally. The point which not actually outlier, it declares as outlier. I will improve this method for finding outliers in clear way.

VII. REFERENCES

[1] Fayyad U., Piatetsky-Shapiro G., Smyth P.: “Knowledge Discovery and Data Mining: Towards a Unifying Framework”, Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining, Portland, OR, 1996, pp. 82-88.

[2] Ester M., Kriegel H.-P., Sander J., Xu X.: “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”, Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining, Portland, OR, AAAI Press, 1996, pp. 226-231.

[3] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: A new data clustering algorithm and its applications. Data Mining and Knowledge Discovery, 1(2):141{182, 1997.

[4] Knorr E. M., Ng R. T.: “Algorithms for Mining Distance- Based Outliers in Large Datasets”, Proc. 24th Int. Conf. on Very Large Data Bases, New York, NY, 1998, pp. 392-403.

[5] Kryszkiewicz, M., Lasek, P.: TI-DBSCAN: Clustering with DBSCAN by Means of the Triangle Inequality. In: Szczuka, M. (ed.) RSCTC_2010. LNCS, vol. 6086, pp. 60–69.Springer, Heidelberg (2010).

[6] Agrawal R., Gehrke J., Gunopulos D., Raghavan P.: “Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications”, Proc. ACM SIGMOD Int. Conf. on Management of Data, Seattle, WA, 1998, pp. 94-105.

[7] Knorr E. M., Ng R. T.: “Finding Intensional Knowledge of Distance-based Outliers”, Proc. 25th Int. Conf. on Very Large Data Bases, Edinburgh, Scotland, 1999, pp. 211-222.

[8] Markus M. Breunig†, Hans-Peter Kriegel†, Raymond T. Ng‡, Jörg Sander†LOF: Identifying Density-Based Local Outliers. MOD 2000, Dallas, TX USA © ACM 2000 1-58113-218-2/00/05 . . .$5.00.

[9] D. R. Heisterkamp and J. Peng, “Kernel Vector Approximation Files for Relevance Feedback Retrieval in Large Image Databases,” Multimedia Tools and Applications, vol 25., N° 2, pp. 175-189, June,2005.

[10] TR 07-017 Anomaly Detection: A Survey Varun Chandola, Arindam Banerjee, and Vipin Kumar August 15, 2007

[11] Marzena Kryszkiewicz and Piotr Lasek , (Neighborhood Based Clustering) by means triangle Inequality clustering algorithm. A C. Fyfe et al. (Eds.): IDEAL 2010, LNCS 6283,pp. 284-291,2010.Springer-Verlog Berlin Heidelberg 2010.

[12] Kryszkiewicz, M., Lasek, P.: TI-DBSCAN: Clustering with DBSCAN by Means of then Triangle Inequality. In: Szczuka, M. (ed.) RSCTC 2010. LNCS, vol. 6086, pp. 60–69. Springer, Heidelberg (2010).