Using the triangle inequality to accelerate Density based Outlier Detection Method

Bidyut Kr. Patra

Department of Computer Science & Engineering
National Institute of Technology Rourkela, Rourkela, Odisha, INDIA

Abstract. Discovering outliers in a collection of patterns is a very well known problem that has been studied in various application domains. Density based technique is a popular one for finding outliers in a dataset. This technique calculates outlierness of each pattern using statistics of neighborhood of the pattern. However, density based approaches do not work well with large datasets as these approaches need to compute a large number of distance computations inorder to find neighborhood statistics. In this paper, we propose to utilize triangle inequality based in- dexing approach to speed up the classical density based outlier detection method LOF. Proposed approach computes less number of distance computations compared to the LOF method. Exper- imental results demonstrate that our proposed method reduces a significant number of distance computations compared to the LOF method.

Keywords: Outlier detection, LOF, triangle inequality, large datasets.

1 Introduction

Finding outliers in a collection of patterns is a very well known problem in the data mining field. An outlier is a pattern which is dissimilar with respect to the rest of the patterns in the dataset. Depending upon the application domain, outliers are of particular interest. In some cases presence of outliers adversely affect the conclusions drawn out of the analysis and hence need to be eliminated beforehand. In other cases, outliers are the center of interest as in the case of intrusion detection system, credit card fraud detection. There are varied reasons for outlier generation in the first place. For example, outliers may be generated due to measurement impairments, rare normal events exhibiting entirely different characteristics, deliberate actions etc. Detecting outliers may lead to the discovery of truly unexpected behavior and help avoid wrong conclusions etc. Thus irrespective of the underlying causes for outlier generation and insight inferred, these points need to be identified from a collection of patterns. There are number of methods proposed in the literature for detecting outliers [1] and are mainly of two types as distance based and density based.

Distance based: These techniques count the number of patterns falling within a selected threshold distance R from a point x in the dataset. If the count is more than a preset number of patterns then x is considered as normal and otherwise outlier. DB-Outlier [2], DOLPHIN [3] are of this type.

Density based: These techniques measure density of a point x within a small region by counting number of points within a neighborhood region. Breunig et al. [4] introduced a concept of local outliers which are detected based on the local density of points. Local density of a point x depends on its K nearest neighbor points. A score known as Local Outlier Factor is assigned to every point based on this local density. All data points are sorted in decreasing order of LOF value. Top m points are chosen as outliers from this sorted order. However, LOF does not work well with large datasets as it computes a large number of distance calculations in order to find K nearest neighbors of each pattern in the dataset.

In this paper, we propose an approach called *TI*-LOF to accelerate the LOF outlier detection method for large datasets. Proposed *TI*-LOF uses an indexing method called TI-k-Neighborhood-Index [5] to quickly find the K nearest neighbors of each pattern in the data. TI-k-Neighborhood-Index uses triangle inequality property of metric space to reduce the search space. Subsequently, this information is used to calculate LOF of each point in the dataset. *TI*-LOF performs less distance computations and takes less time compared to the classical LOF method. Experimental results validate our claim.

Rest of the paper is organized as follows. Section 2 discusses background of the proposed approach. Section 3 discusses the proposed speeding up approach *TI*-LOF. Experimental evaluations are reported in section 4. Finally, we conclude with section 5.

2 Background of Proposed Approach

Our proposed approach is a combination of two schemes i.e, TI-k-Neighborhood-Index and LOF, which are discussed briefly.

2.1 TI-k-Neighborhood-Index Approach

Our proposed speeding up approach *TI*-LOF exploits an indexing scheme TI-k-Neighborhood-Index [5] which is based on triangle inequality property. In recent years, triangle inequality property of the metric space has been used to reduce the distance computations in the clustering methods [6–8]. Elkan [6] used triangle inequality to reduce the distance computation in k-means clustering method. Nassar [7] used for speeding up summarization scheme (data bubble). Recently, Marzena et al. [8] proposed to speed up DBSCAN method using triangle inequality. TI-k-Neighborhood-Index is also proposed by Marzena et al. in [5]. It works as follows.

Initially, data points are arranged in a sorted (ascending) list with respect to the magnitudes (*norm*) of the points. For each point $x \in D$, it identifies K points q preceding and following x in the sorted list such that difference between $||x||$ and $||q||$ is smallest. These q points determines a radius EPS of a sphere centered at x, which covers all K nearest neighbors of x. The radius EPS is the maximum of distances between x and q. Then, TI-K-Neighborhood-Index uses theorem 1 to check whether each point p (starting from closest point from x in the list) preceding and following x can be the potential K neighbor or not.The whole approach is depicted in Algorithm 1.

Theorem 1 *[5] Let D be a set of points ordered in a non-decreasing way with respect to their magnitudes ($||.||$) of the points. Let p be any point in D, and $EPS \in \mathcal{R}^+$ be a value such that*

$|N_{EPS}(p)^1| >= K$. Let q_f and q_b be preceding and succeeding points of p in the ordered set D, respectively such that $||q_f|| - ||p|| > EPS$ and $||p|| - ||q_b|| > EPS$.Then

1. *q_f and all points following q_f do not belong to K neighborhood of p.*
2. *q_b and all points preceding q_b do not belong to K neighborhood of p.* □

Algorithm 1 $TI\text{-}k\text{-}Neighborhood\text{-}Index(\mathcal{D}, K)$

```
for each pattern x ∈ D do
    Calculate ||x||
end for
Make a list T of all sorted patterns, which are ordered in ascending order with respect to | | . | |
for each pattern x ∈ T  do
    Pick K patterns q preceding and succeeding x such that ||x|| − ||q|| is minimum.
    Calculate EPS = max_{i..K} ||x − q_i||
    KNN Set (x) = ∅
    Let p_i be the closest pattern preceding x in list T.
    while (||x|| − ||p_i|| ≤ EPS) do
        if (||x − p_i|| ≤ EPS) then
            KNN Set(x) = KNN Set ∪{p_i} and store distance between x and p_i
            if (There exists an immediate preceding point p_{i+1} in T) then
                p_i = p_{i+1}
            end if
            EPS = maximum of computed distance between x and p ∈ KNN Set(x).
        end if
    end while
    Let q_i be the closest pattern succeeding x in list T.
    while (||x|| − ||q_i|| ≤ EPS) do
        if (||x − q_i|| ≤ EPS) then
            KNN Set(x) = KNN Set ∪{q_i} and store distance between x and q_i
            if (There is an immediate succeeding point q_{i+1} in T) then
                q_i = q_{i+1}
            end if
            EPS = maximum of computed distance between x and q ∈ KNN Set(x).
        end if
    end while
    Sort all points in KNN Set and output first K points as K nearest neighbors of x
end for
```

2.2 LOF:Identifying Density-Based Local Outliers

M.M.Breunig [4] proposed a density based outlier detection method called LOF. LOF introduces a factor called $local\ outlier\ factor\ (lof)$ to measure the degree of outlierness of a pattern in the

[1] $N_{EPS}(p) = \{q \in D \mid q \neq p,\ ||p - q|| \leq EPS\}$

dataset. LOF uses the K nearest neighbor information of points to calculate *lof* of data points. To make our article more convenient to reader, we recall definitions of $reachability$-$distance$ $(reach$-$dist)$, $local\ \ reachability\ density$ (lrd) and finally, $local\ outlier\ factor$ (lof).

Definition 1 (Reachability distance (reach-dist) of a point) *Let x, y be two arbitrary points in $\mathcal{D}$. and K-$NN(y)$ be K nearest points of y. reachability distance of x with respect to y is defined as follow.*

$$reach\text{-}dist(x, y) = \begin{cases} ||x - y|| & \text{if } x \notin K\text{-}NN(y) \\ \max\{||y - q|| \quad | \ q \in K\text{-}NN(y)\} & \text{if } x \in K\text{-}NN(y), \end{cases}$$

Definition 2 (Local reachability density (lrd) of a point) *Let K-$NN(x)$ be the K- nearest neighbors of $x \in \mathcal{D}$. Local reachability density (lrd) is defined as follow.*

$$lrd(x) = 1/\frac{\sum_{o \in K\text{-}NN(x)} reach\text{-}dist(x, o)}{|KNN(x)|}$$

Definition 3 (local outlier factor of a point) *The local outlier factor of $x \in \mathcal{D}$ is defined as*

$$lof(x) = \frac{\sum_{o \in K\text{-}NN(x)} \frac{lrd(o)}{lrd(x)}}{|K - NN(x)|}$$

The LOF start searching K nearest neighbors of all points in the dataset and subsequently calculates local reachibility density of the points. This information is used to compute *lof* of all points. Finally, all points are sorted with respect to their lof values and top N data points are declared as outlier points. However, LOF is not computationally efficient method. It needs to compute a large number of distance calculation for finding K nearest neighbors of all data points. To overcome this shortcoming of the LOF, we propose a speeding up approach which is discussed below.

3 TI-LOF:Proposed Speeding up Approach

The proposed *TI-LOF* is a hybrid approach with a combination of TI-k-Neighborhood-Index and classical LOF methods. The method works as follow. At first TI-k-Neighborhood-Index is applied to the dataset $\mathcal{D}$ to collect K nearest neighbors of all data points quickly. From this statistics, TI-LOF calculates maximum of distances between a point x and its K neighbors. In the last step, lof of all points are calculated and arranged them in decreasing order. The whole method is depicted in Algorithm 2.

4 Performance Evaluations

To evaluate *TI-LOF*, we implemented it using C language on Intel(R) Core 2 Duo CPU(2.90GHz) Desktop PC with 2 GB RAM. We tested our method with classical LOF method. Detailed results are reported in this section.

Algorithm 2 $TI - LOF(\mathcal{D}, K, N)$

/* $\mathcal{D}$ is a dataset, K is the value of K nearest neighbor and N desired number of outliers. */
Apply *TI-k-Neighborhood-Index* $(\mathcal{D},\ K)$ as given in Algorithm 1
for each pattern $x \in \mathcal{D}$ **do**
  Calculate $MAX_K(x)$ =$\max\{||x - q||\ \ |\ q \in K - NN(x)\}$
end for
for each pattern $x \in \mathcal{D}$ **do**
  Calculate lrd(x) using Definition 1.
end for
for each pattern $x \in \mathcal{D}$ **do**
  Compute $lof(x)$
end for
Sort all points in decreasing order in a LIST with respect to lof values.
Output top N points in the LIST.

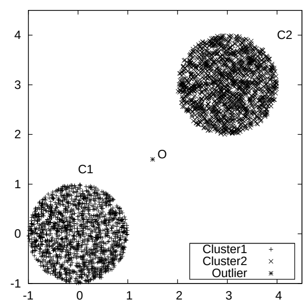

Fig. 1. Uniform Dataset.

4.1 Synthetic Dataset

One synthetic dataset called **Uniform dataset** is designed to evaluate the *TI-LOF* method. This is a two dimensional dataset having two circular shaped clusters filled with highly densed points. There is a single outlier (say O) placed exactly in the middle of the two densed clusters as shown in the Figure 1. We ran *TI-LOF* method along with classical LOF method with Uniform dataset. Obtained results for different values of K are reported in Table 1. From experimental results, it can be observed that *TI-LOF* performs 6 millions less distance computations compared to the classical LOF method. Our method takes less time (with $K = 30$, 0.57 seconds) than classical LOF method (with $K = 30, 1.45$ seconds).

4.2 Real Dataset

Shuttle Dataset: This dataset has 9 integer valued attributes of $58,000$ patterns distributed over 7 classes (after merging training and test sets). Class labels are eliminated from the all patterns. With $K = 30$, TI-LOF computes more than 138 millions less distance calculations compared to the LOF method (Fig. 2).

Table 1. Experimental Results with Uniform Dataset

Value of (K)	Method	Time (in Sec.)	Number of Distance Computation (in Millions)
15	LOF	0.98	9.85
	TI-LOF	**0.38**	**2.98**
20	LOF	1.14	9.86
	TI-LOF	**0.43**	**3.10**
25	LOF	1.29	9.86
	TI-LOF	**0.49**	**3.21**
30	LOF	1.45	9.87
	TI-LOF	**0.57**	**3.30**

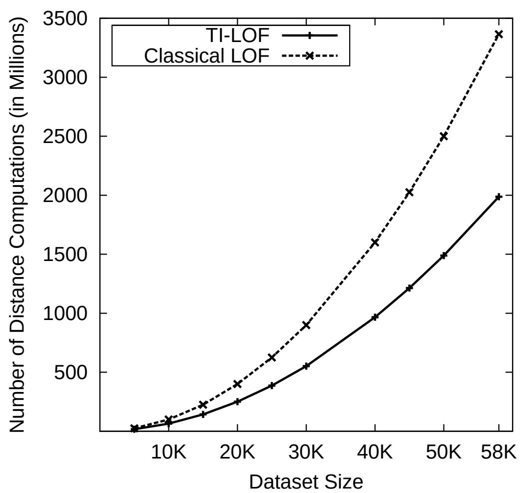

Fig. 2. Number of Distance Computation varies with size of the dataset for Shuttle Data

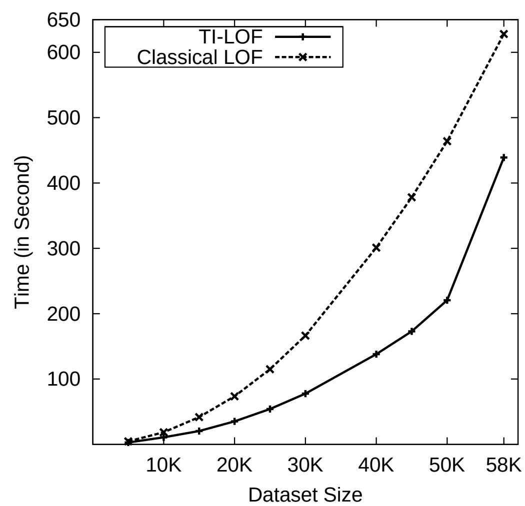

Fig. 3. Execution time of LOF and *TI-LOF* for Shuttle Data

Fig. 2 shows the number of distance computed by *TI-LOF* and LOF as data size varies from 5000 to 58000 for Shuttle dataset with $K = 30$. It may be noted that with the increase of dataset size, number of distance computations are reduced compared to LOF method significantly.

Fig. 3 shows the execution time of *TI-LOF* and LOF methods as data size varies from 5000 to 58000 for Shuttle dataset with $K = 30$. It can be observed that *TI-LOF* method takes less time compared to the LOF method.

5 Conclusion

TI-LOF is a speeding up approach for classical outlier detection method LOF. Metric space property is used to reduce the number of distance computations in LOF method. Experimental results demonstrate that our proposed approach computes significantly less distance calculations compared to LOF method and the TI-LOF is faster than LOF method.

References

1. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM Comput. Surv. **41**(3) (2009)
2. Knorr, E.M., Ng, R.T.: Algorithms for mining distance-based outliers in large datasets. In: Proceedings of the 24rd International Conference on Very Large Data Bases. VLDB '98 (1998) 392–403
3. Angiulli, F., Fassetti, F.: Dolphin: An efficient algorithm for mining distance-based outliers in very large datasets. ACM Trans. Knowl. Discov. Data **3** (March 2009) 4:1–4:57
4. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: Lof: identifying density-based local outliers. In: Proceedings of the 2000 ACM SIGMOD international conference on Management of data. SIGMOD '00 (2000) 93–104
5. Kryszkiewicz, M., Lasek, P.: A neighborhood-based clustering by means of the triangle inequality. In: Proceedings of the 11th international conference on Intelligent data engineering and automated learning (IDEAL 2010). (2010) 284–291
6. Elkan, C.: Using the triangle inequality to accelerate k-means. In: Proceedings of the Twentieth International Conference on Machine Learning, ICML'03). (2003) 147–153
7. Nassar, S., Sander, J., Cheng, C.: Incremental and effective data summarization for dynamic hierarchical clustering. In: Proceedings of SIGMOD Conference, SIGMOD'04. (2004) 467–478
8. Kryszkiewicz, M., Lasek, P.: Ti-dbscan: Clustering with dbscan by means of the triangle inequality. In: Proceedings of the 7th International Conference on Rough Sets and Current Trends in Computing RSCTC, 2010. (2010) 60–69