Journal of Computer Science 4 (2): 129-132, 2008
ISSN 1549-3636

Fast Algorithms for Outlier Detection

[1]Fawaz A.M. Masoud, [1]Moh'd Belal Al- Zoubi, [2]Imad Salah and [3]Ali Al-Dahoud
[1]Department of Computer Information Systems, University of Jordan, Amman 11942, Jordan
[2]Department of Computer Science, University of Jordan, Amman, Jordan
[3]Department of Computer Science, Al-Zaytoonah University, Amman, Jordan

Abstract: Finding fast algorithms to detect outliers (as unusual objects) by their distance to neighboring objects is a big desire. Two algorithms were proposed to detect outliers quickly. The first was based on the Partial Distance (PD) algorithm and the second was an improved version of the PD algorithm. It was found that the proposed algorithms reduced the number of distance calculations compared to the nested-loop method.

Key words: Outlier detection, K-Nearest Neighbour (KNN), partial distance, data mining

INTRODUCTION

Outlier detection has many practical applications in different domains. In many data mining applications, identifying exceptions or rare events can often lead to the discovery of unexpected knowledge in areas such as fraud detection[1], identifying computer network intrusions and bottlenecks[2] and criminal activities in E-commerce and detection of suspicious activities[3].

Outliers are data points (vectors) with values much different from those of the remaining set of data[4]. Outliers may represent errors in the data or could be correct data values that are simply much different from the remaining data. Outliers can be described as follows[5]. Given a set of N data points or objects and an expected number of outliers, n, find the top n objects that are considerably dissimilar, exceptional, or inconsistent with respect to the remaining data.

One of the most popular approaches for detecting outliers is the distance-based approach[6-9]. In this approach, the distance of a point from its k nearest points (or neighbors) is calculated. If the neighboring points are relatively close, then the point is considered normal; however, if the neighboring points are far away, then the point is considered outlier. The advantages of this approach are that no explicit distribution needs to be defined to detect outliers and can be applied to any feature space for which a distance measure can be defined[6-8].

Given a distance measure on a feature space, there are many different definitions for the distance-based outliers. Knor and Ng[4] present the following definition. A point p in a data set is an outlier with respect to the parameters k and d, if at least k points in the dataset lie greater than distance d from p.

Ramaswami *et al.*[10], proposes a new formulation for distance-based outliers, based on the distance of a point, p, to its k^{th} nearest neighbor, denoted with $D^k(p)$. Given a k and n, a point p is an outlier if no more than n-1 other points in the data set have a higher value for D^k than p. This means that the top n points, having the maximum D^k values, are considered outliers.

Most recently, Angiulli and Pizzuti[8]) propose a new definition of outliers. In this definition, for each point, p, the sum of the distances from its k nearest neighbors is considered. This sum is called the weight of p, $w_k(p)$ and is used to rank the points of the data set. Outliers are those points having the largest values of w_k. Thus, given n, the expected number of outliers in the data set and an application dependent parameter k, specifying the size of the neighborhood of interest, the outlier detection problem consists of finding the n points of the data set scoring the maximum w_k values. The problem with the distance-based approach is its high computational complexity.

Distance-based approaches are simple to implement. However, they suffer exponential computational growth as they are founded on the calculation of the distances between all objects in the dataset. The computational complexity is directly proportional to both the dimensionality of the data and the number of objects. Hence, techniques for efficiently calculating distance with a lower runtime are required[7,11].

Researchers have tried a variety of methods to find these outliers efficiently. In[6], the authors propose the

Corresponding Author: Fawaz A.M. Masoud, Department of Computer Information Systems, University of Jordan, Amman 11942, Jordan Tel: +962-5355000 Fax: +962-5354070

nested-loop algorithm for finding distance-based outliers. In this method, one compares each data point in the data set with every other point to determine its k nearest neighbors. Given the neighbors for each data point in the data set, simply select the top n candidates according to the outlier definition. This method has quadratic complexity as we must make all pairwise distance computations between the data points.

Another method for finding outliers efficiently is to use a spatial indexing structure such as a KD-tree, R-tree, or X-tree to find the nearest neighbors of each candidate point[6]. One queries the index structure for the closest k points to each data point and as before, one simply selects the top candidates according to the outlier definition. For low-dimensional data sets, this approach can work very well and potentially scales as N logN if the index tree can find a point's nearest neighbors in logN time. However, index structures can lead to poor performance as the dimensionality increases[7,10].

Bay and Schwabacher[7] present an algorithm which is based on the nested-loop algorithm, using randomization and pruning rule, with near linear time performance. However, the algorithm depends on the data ordering which can lead to poor performance, as the authors reported. In addition, the algorithm may perform poorly if the data does not contain outliers.

In this study, we propose two algorithms to detect outliers quickly. The first is the Partial Distance (PD) algorithm and the second is a proposed (improved) version of the PD algorithm.

PARTIAL DISTANCE

The Partial Distance (PD) algorithm[12,13,14] has been proposed to reduce computation complexity of the LBG algorithm of[15] within the area of Vector Quantization (VQ).

The PD logic first calculates the distance (squared) between a query point, p and an arbitrary data point and takes this distance as the current initial minimum distance. Then it continuously compares the accumulative partial distance between the query point and each candidate data point with the current minimum distance. If the accumulative partial distance exceeds the current minimum distance, the candidate data point is eliminated (rejected) before completing the total distance calculation. If a total distance is obtained, then the current minimum distance is updated by choosing the minimum of the current minimum distance and the calculated distance.

Let $X = \{x_i, i = 1,\ldots,N\}$ be a set of data points (vectors) of size N, where $(x_{ij}, j = 1,\ldots,K)$ is a K dimensional data point. For a given query point $P = (p_j, j = 1,\ldots,K)$, it is required to find the data point with the minimum distance from the set X under the squared-error distance measure defined as:

$$d(P, x_i) = \sum_{j=1}^{K} (p_j - x_{ij})^2$$

The basic structure of the partial distance (PD) search algorithm[12,13] is as follows:

$d_{min} = \infty$
Loop A: For i = 1 to N
d = 0
Loop B: For j = 1 to K
$d = d + (p_j - x_{ij})^2$
if $d > d_{min}$ Next i (exit condition)
Next j
$d_{min} = d$
min = i
Next i

It can be observed that the PD search algorithm gains computation saving over the full search algorithm because of the provision for a premature exit from Loop B, on satisfying the condition $d > d_{min}$ (called the exit condition) before the completion of the distance computation, $d(P, x_i)$.

IMPROVED PARTIAL DISTANCE

The performance of the PD algorithm is sensitive to the choice of the initial minimum distance, d_{min}[16]. This might degrade the performance of the PD algorithm.

Instead of choosing an arbitrary data point, which is the case in the PD algorithm, one might think of choosing the mean value of the data set. However, choosing the mean value might lead to wrong results in cases where the mean value is the closest nearest neighbor to a given query point. This is because the mean value might not be one of the points in the data set.

Our approach is based on finding the data point nearest to the mean value (termed Nmean) and then computing the distance d_{Nmean} between each data point and the Nmean. The resulting distances are used as initial minimum distances. In this case, we achieve an improved performance over the PD algorithm. The Improved Partial Distance (IPD) algorithm is as follows:

Table 1: Computations based on the percentage CPU time of the nested-loop method with a different number of points (No. points) and different dimensions, D, for the random data set

	2D		4D		8D	
No. of points	PD	IPD	PD	IPD	PD	IPD
10000	75	63	60	48	75	66
20000	69	56	60	48	80	64
30000	64	54	55	45	89	73
40000	63	38	57	50	68	63
50000	61	52	53	47	63	54

$d_{min} = d_{Nmean}$
Loop A: For i = 1 to N
$d = 0$
Loop B: For j = 1 to K
$d = d + (p_j - x_{ij})^2$
if $d > d_{min}$ Next i (exit condition)
Next j
$d_{min} = d$
min = i
Next i

The results of applying the IPD algorithm show some improvement over the PD algorithm, as discussed next.

RESULTS AND DISCUSSION

We will investigate the efficiency of the two proposed algorithms (PD and IPD) when applied to detect outliers. The proposed algorithms described in this study generate identical outputs and so are not assessed on the basis of accuracy. The performance of the algorithms is primarily determined by the number of distance calculations carried out.

In order to test the efficiency of the proposed algorithms, two data sets have been tested. The first set represents random data with the dimensions (2D, 4D and 8D). The second set represents data extracted from 1 min of speech with three different dimensions (2D, 4D and 8D). The performance of the proposed method is reported in terms of percent of the nested-loop method.

Table 1 shows a summary of the results. The Table 1 shows the performance of the proposed algorithms is better than those obtained from the nested-loop in all cases. The Table 1 also shows that the IPD algorithm performs the best.

Table 2 is a summary of the results for the speech data set. The Table 2 shows that the performance of the proposed algorithms is better than the performance of the nested-loop in all cases. The Table 2 also shows that the IPD algorithm has some improvements over the PD algorithm.

Table 2: Computations based on the percentage CPU time of the nested-loop method for the speech data with a different number of points (No. points) and different dimensions, D

	2D		4D		8D	
No. of points	PD	IPD	PD	IPD	PD	IPD
10000	73	60	60	48	55	49
20000	80	66	60	54	53	45
30000	59	50	55	45	52	46
40000	66	50	57	50	45	42
50000	58	50	52	45	55	47
60000	66	59	55	48	54	48

The two tables show that the results obtained from the IPD algorithm are the best in all cases. The tables also show that a better performance is obtained for higher dimensions, particularly on the real data.

CONCLUSION

In this study, we have proposed two algorithms to detect outliers quickly. The first algorithm is the Partial Distance (PD) algorithm and the second algorithm is an improved version of the PD algorithm proposed in this study. The results offer a significant increase in efficiency over the nested-loop method when applied to both random and real data sets, particularly with the increase of the number of data points and dimensions. It is also noticed that the proposed algorithms gave better performance when a real data set was applied.

REFERENCES

1. Bolton, R. and D. Hand, 2002. Statistical fraud detection: A review (with discussion). Stat. Sci., 17: 235-255.
2. Lane, T. and C. Brodley, 1999. Temporal sequence learning and data reduction for anomaly detection. ACM Trans. Inform. Syst. Secur., 2: 295-331.
3. Chiu, A. and A. Fu, 2003. Enhancement on local outlier detection. In: 7th International Database Engineering and Application Symposium (IDEAS03), pp: 298-307.
4. Knorr, E. and R. Ng, 1998. Algorithms for mining distance-based outliers in large data sets. In: Proceeding the 24th International Conference on Very Large Databases (VLDB), pp: 392-403.
5. Han, J. and M. Kamber, 2006. Data Mining: Concepts and Techniques. 2nd Edn. Morgan Kaufmann.
6. Knorr, E., R. Ng and V. Tucakov, 2000. Distance-based outliers: Algorithms and applications. VLDB J., 8: 237-253.

7. Bay, S. and M. Schwabacher, 2003. Mining distance-based outliers in near Linear Time with Randomization and a Simple Pruning Rule. SIGKDD '03, Washingtno, DC, USA.
8. Angiulli, F. and C. Pizzuti, 2005. Outlier mining in large high-dimensional data sets. IEEE Trans. Knowl. Data Eng., 17: 203-215.
9. Shrestha, M., H. Hamilton and Y. Yao, 2006. The PDD framework for detecting categories of peculiar data. In: Proceeding 6th International Conference on Data Mining (ICDM06), pp: 562-571.
10. Ramaswami, S., R. Rastogi and K. Shim, 2000. Efficient algorithm for mining outliers from large data sets. In: Proceeding ACM SIGMOD, pp: 427-438.
11. Hodge, V. and J. Austin, 2004. A survey of outlier detection methodologies. Artificial Intel. Rev., 22: 85-126.
12. Bei, C. and R. Gray, 1985. An improvement of the minimum distortion encoding algorithm for vector quantization. IEEE Trans. Commun., 33: 1132-1133.
13. Paliwal, K. and V. Ramasubramanian, 1989. Effect of ordering the codebook on the efficiency of the partial distance search algorithm for vector quantization. IEEE Trans. Commun., 37: 538-540.
14. Venkateswarlu, N. and P. Raju, 1993. A new fast classifier for remotely sensed images. Int. J. Remote Sens., 14: 383-390.
15. Linde, Y., A. Buzo and R. Gray, 1980. An algorithm for vector quantizer design. IEEE Trans. Commun., 28: 89-95.
16. Chen, S. and W. Hsieh, 1991. Fast algorithm for VQ codebook Design. IEE Proc., 138: 357-362.