Egyptian Informatics Journal 22 (2021) 125–132

Contents lists available at ScienceDirect

Egyptian Informatics Journal

journal homepage: www.sciencedirect.com

Local dynamic neighborhood based outlier detection approach and its framework for large-scale datasets

Renmin Wang [a], Qingsheng Zhu [a,*], Jiangmei Luo [a], Fan Zhu [b]

[a] Chongqing Key Lab. of Software Theory and Technology, College of Computer Science, Chongqing University, Chongqing 400044, China
[b] Chongqing Institute, Chinese Academy of Sciences, Chongqing 400714, China

ARTICLE INFO

Article history:
Received 19 January 2020
Revised 14 May 2020
Accepted 9 June 2020
Available online 3 July 2020

Keywords:
Local outlier detection
Large-scale
Dynamic references nearest neighbors
Detection framework
k-means

ABSTRACT

Local outlier detection is a hot area and great challenge in data mining, especially for large-scale datasets. On the one hand, traditional algorithms often achieve low-quality detection results and are sensitive to neighborhood size. On the other hand, they are infeasible for large-scale datasets due to at least $O(N^2)$ time and space complexity. In light of these, we propose a new local outlier detection algorithm, which is designed based on a new stable neighborhood strategy-dynamic references nearest neighbors (DRNN). Meanwhile, we present a new detection framework by combining the proposed approach and k-mean for large-scale datasets. Experimental results demonstrate that the proposed algorithm can produce higher quality and robust detection results compared to several classic methods. Meanwhile, the new detection framework is able to significantly improve detecting efficiency without sacrificing accuracy.

1. Introduction

Based on basic definitions of an outlier [1,2], local outlier detection aims to capture the records that are deviant from their neighbors. Compare to global outliers, local outliers are more interesting and challenge in data mining and machine learning. At present, several strategies [3–7] have been proposed for outlier detection and many algorithms have been developed for various practical applications [8–13]. Particularly, local outlier detection includes two basic tasks: how to quantify neighborhood of a data object and how to estimate local outlier degree of a data object. In general, most local algorithms use k-nearest neighbors (kNN) to measure their neighborhoods and define their outlier factor according to different strategies, such as distance and density.

However, traditional local algorithms usually obtain low-quality results and are sensitive to neighborhood parameter, such as parameter k. Most traditional algorithms use kNN to quantify their neighborhood of data objects. As is well known, kNN quantifies a rounded or spherical local region, therefore, it is a rough neighborhood measurement and not feasible for datasets with non-spherical clusters. Moreover, it is hard to set k value without prior knowledge and the kNN-based detection algorithms are usually sensitive to k due to instability of kNN. Thus, many convenient local approaches often show poor detection performance and low robustness, especially for datasets with complex distributions.

Furthermore, nearest neighbors based algorithms generally take $O(N^2d)$ time and $O(N^2)$ space complexity to construct the affinity matrix, where N and d are the data size and the dimension, respectively. As the data size N increases, the computational and storage burden of them grows dramatically. For example, given a dataset with 1 million samples, the $N \times N$affinity matrix will need 7,450.58 GB of memory, which could lead to the memory bottleneck of a common computer, not to mention the next phase of detection process.

To alleviate the problems mentioned above,a novel Local Dynamic Neighborhood based Detection algorithm (LDNOD for short) which includes new definitions of neighborhood and outlier degree has been proposed. To measure a stable and accurate neighborhood for arbitrary data, a new nearest neighbors measurement, named Dynamic References Nearest Neighbors (DRNN) that searches neighbors based on dynamic references instead of a fixed and single reference, is designed first. Based on DRNN, a new outlier factor is defined to score each region rather than each data object. Toward large-scale data, a detection framework (LDNOD-km) by combining advantages of LDNOD and k-means is developed. Extensive experiments have been conducted on various

* Corresponding author.
E-mail address: qszhu@cqu.edu.cn (Q. Zhu).

Peer review under responsibility of Faculty of Computers and Information, Cairo University.

https://doi.org/10.1016/j.eij.2020.06.001

datasets, which have shown the advantages of the LDNOD and LDNOD-km compared with several the-state-of-art algorithms.

2. Related work

The local outlier factor (LOF) [14] is the most well-known local outlier detection algorithm and introduce the idea of local outlier first. LOF can be considered as a ratio of local densities, and a higher LOF value indicates more likely of a local outlier. Based on the idea of LOF, many variants of it have been proposed, such as connectivity-based outlier factor (COF) [15], local correlation integral (LOCI) [16], influenced outlierness (INFLO) [17], local outlier probability (LoOP) [18], local distance-based outlier factor(LDOF) [19], and so on. COF is similar to LOF, but the former estimates the local density of a data record using a set based nearest trail (SBN-trail) approach. Compared to LOF, COF indicates how far away a data instance shifts from a pattern. A interesting contribution of LOCI is the LOCI plot, which summarizes a wealth of information about the data in the vicinity of the point, and provides an intuitive understanding of why specific data points should be recognized as outliers. When a dataset contains clusters with different densities and they are close to each other, some traditional methods, such as LOF, would fail to score the points at the borders of the clusters. In INFLO,both the kNN and Reverse k-nearest neighbors (RkNN) [20,21] are combined to compute the outlier score. By using this strategy, outliers between clusters with different densities would be detected more accurately. To address the issue of threshold selection, LoOP uses a useful idea of outputting an anomaly probability instead of a outlier score for a data point. Due to implicit data patterns and parameter setting issues, existing outlier detection algorithms are ineffective on scattered real-world datasets. Zhang et al. proposed a novel LDOF method to measure the outlier scores of objects in scattered datasets, which uses the relative location of an instance to its kNN neighbors to determine the degree to which the instance deviates from its neighborhood. These methods mentioned above usually use kNN or RkNN to measure their neighborhoods. In recent years, Zhu et al. proposed natural neighbors (NaN) [22] and natural outlier factor (NOF) [23] to improve the robustness of local detection. The NaN is designed by integrating kNN and RkNN when the stable searching state is satisfied, and the NOF algorithm can detection outliers without parameter k.

Toward large-scale datasets, the partition-based strategy has been widely adopted as a powerful and efficient tool for local outlier detection [24,25]. In general, they usually partition the large-scale datasets into small and large clusters via a clustering method and then outliers are detected from each cluster. In fact, k-means [30] is a commonly partition method due to its low computational and space complexity. For example, clustering-based local outlier factor (CBLOF) is calculated by the distance of each data object to its respective cluster center after partitioning. Similar to CBLOF, LDCOF separates a dataset into clusters first by using k-means and then computes the LDCOF scores by dividing the distance of an object to its cluster center by the average distance. Additionally, the histogram-based outlier score (HBOS) [26] is a very fast anomaly detection algorithm to compute the feature probabilities of each data object. Zhao et al. [27] proposes the scalable unsupervised outlier detection (SUOD) for high-dimension large datasets by integrating several classical detection methods.

Despite the considerable efforts that have been made via various strategies in recent years, it remains a challenging task to improve accuracy and robustness of local detection. Moreover, traditional detection frameworks often have low-quality results for large-scale datasets because they heavily rely on the unstable clustering results and traditional low-precision detection algorithms. In light of this, we propose the LDNOD and its framework.

3. The proposed algorithm and its framework

In this section, the proposed algorithm (LDNOD) and its framework (LDNOD-km) are introduced in details. The LDNOD can produce high-quality and robust detection results. Meanwhile, the detection framework by integrating LDNOD with k-means can handle larger-scale datasets efficiently without sacrificing accuracy.

3.1. Local dynamic neighborhood based outlier detection

To address some problems presented above, we propose a new local neighborhood-based outlier detection approach-LDNOD. Unlike traditional algorithms, a new stable neighborhood method is presented as its neighborhood system. Moreover, it scores each local region instead of each data object for a dataset.

According to the two basic factors of a local detection algorithm, we define our own neighborhood and local outlier factor below.

Definition 1. (Dynamic references nearest neighbors of x_i) Let x_i be a data record for a dataset $X = \{x_1, x_2, \ldots, x_N\}$ and κ be the neighborhood parameter (like parameter k in kNN). The dynamic references nearest neighbors of x_i is defined as

$$DRNN(x_i, p) = DRNN(x_i, p-1) \longrightarrow NN(DRNN(x_i, p-1)), p \in [2, \kappa] \quad (1)$$

where $NN(x)$ denote a set of nearest neighbors of x and x is a substructure with a set of data objects. The initial structure is $DRNN(x_i, 0) = \{x_i\}$. Note that the number of nearest neighbors of $NN(x)$ is greater than or equal to 1. The '$\longrightarrow$' denotes point to respective nearest neighbor or neighbors, therefore, our DRNN neighborhood can be regarded as a directed and acyclic graph. Intuitively, the DRNN is constructed iteratively until the number of data objects reach to κ except x_i.

Fig. 1 illustrates how the DRNN method to construct a neighborhood. As is shown in Fig. 1, the DRNN neighborhood of point 1 $DRNN(x_1, 5)$ is constructed through 4 iterations. From A to D, each sub-neighborhood points to its nearest neighbor or neighbors. In Fig. 1 D, point x_9 is excluded due to $\kappa = 5$. From this simple example, we can intuitively observe the different searching strategy and characteristics of DRNN from traditional neighborhood methods.

Compare with traditional neighborhood methods, such as kNN and RkNN, the DRNN is feasible for measuring local neighborhood for arbitrary datasets. It is generally known that kNN always quantifies a rounded or spherical local region. In other words, kNN potentially assumes that a data point and its nearest neighbors are distributed with rounded or spherical shapes. In fact, a dataset is often complex and could be arbitrary distributions and shapes. Therefore, the neighborhood methods like kNN are inadequate for measuring arbitrary datasets, which would lead to low-quality detection results of traditional algorithms that use kNN. However, our DRNN strategy can quantify correct local neighborhood for arbitrary distributions due to its dynamic instead of a fixed reference objects.

Unlike other neighborhood methods, such as kNN and RkNN, our DRNN is also stable and insensitive to neighborhood parameter κ. The kNN and RkNN are constructed by using a fixed reference point. Therefore, they would add much different objects as their neighbors with increase of k. That is why they are unstable and insensitive to parameter k. However, the DRNN uses a dynamic and multiple reference data objects. When it searches neighbors, it can use similarity propagation between existing neighbors. From the perspective of social networks, DRNN considers both friends of a person and friends of him. Obviously, the new neighbors are as similar as possible with original data objects and their neighbors

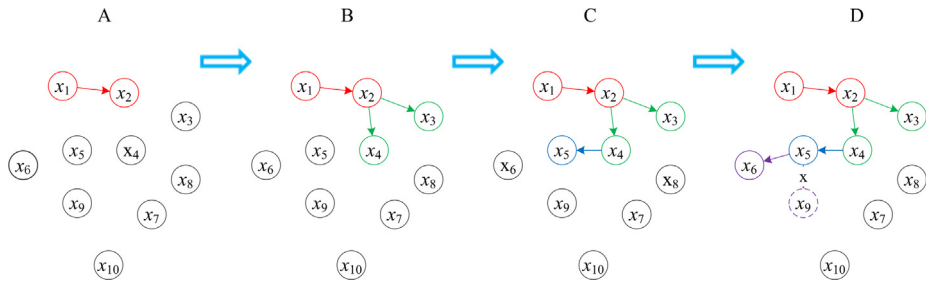

Fig. 1. Process of DRNN neighborhood for $\kappa = 5$. Arrows of different colors denotes different iteration results. Note that the reference points are gradually changed from $\{x_1\}$ to $\{x_1, x_2, x_3, x_4, x_5\}$.

with growth of κ. That is why our DRNN neighborhood is stable and insensitive to parameter κ.

Because close data objects have similar even same DRNN neighborhoods, we construct a common neighborhood for near data points. In other worlds, all data objects within a DRNN neighborhood has a common neighborhood. It is worth noting that different DRNN neighborhoods can share neighbor or neighbors to maintain natural features of them. Therefore, for a dataset X with N data records, we only need to construct $[N/(\kappa+1), N-\kappa]$ neighborhoods according to different sharing data points. Obviously, this manner is useful to improve efficiency of neighborhood construction, which usually takes majority running time of a nearest neighbors based detection algorithm.

With stable neighborhood method designed, we define our local outlier factor. Based on the definition of DRNN, we find that one and only one anomaly edge is included in any anomaly DRNN neighborhood. In light of this, we design the outlier degree below.

Definition 2. (Local neighborhood outlier factor, LNOF) given a local neighborhood sj, its outlier factor can be expressed as follow:

$$LNOF(s_j) = \frac{d_{max}(s_j)}{sum(s_j) - d_{max}(s_j)} \tag{2}$$

where

$$d_{max}(s_j) = max\{e_1, e_2, \ldots, e_\kappa\} \tag{3}$$

and

$$sum(s_j) = \sum_{e_\lambda \in s_j} dist(e_\lambda), \quad \lambda \in [1, \kappa] \tag{4}$$

Intuitively, $sum(s_j)$ denotes the sum of the edges within s_j; $d_{max}(s_j)$ is the length of the longest edge in s_j. Therefore, LNOF indicates that the degree which the potential anomaly edge deviates from the normal edges within the neighborhood s_j. It is easy to see that larger LNOF indicates greater the likelihood of an anomaly region. When LNOF approximately equal to 1 (but it should be larger than 1), it means that all data objects within the respective local neighborhood are similar to each other and no outlier is contained in this region. On the contrary, when $LNOF \gg 1$, it means s_j contains an outlier or outliers. Note that our LNOF does not measure outlier degrees of data objects directly.

Unlike traditional manners, the LNOF score each local neighborhood instead of each data object, that is why we named it local neighborhood based outlier factor. On the one hand, it deal with a local region at a time more than a data object, which can also potentially improve the efficiency of the proposed algorithm. In fact, near data objects have similar outlier degrees, thus there is no need to compute an outlier scores for all data points. On the other hand, it is able to recognize single outlier and multiple outliers (a group of outliers).

After scoring each local neighborhood, we only need to separate outliers from each anomaly neighborhoods. According to the characteristics of DRNN neighborhood and definition of LNOF, for any anomaly DRNN s_j, in which the outlier or outliers are the data objects located in front of the longest edge. Fig. 2 shows how to capture an outlier or outliers for an anomaly local neighborhood. As is shown in Fig. 2, points x_1 and x_2 are labeled as outliers by cutting off the longest edge. Note that our DRNN neighborhood is a directed graph.

Traditional detection algorithms usually use a top-n manner where they output the top scored n data objects as outliers. The parameter n is often not available without prior knowledge, and their detection quality heavily relies on it. However, for the LDNOD algorithm, we can intuitively set the LNOF threshold value, such as 3 or 5, although no consensus has been reached at present.

3.2. Process and complexity

We summary the process of the proposed algorithm as follows:

step 1: Construct DRNN neighborhoods for a dataset;
step 2: Score the DRNN neighborhoods and report anomaly DRNN neighborhoods according to a user-provided threshold value.
step 3: Capture outliers from the anomaly neighborhoods.

Take a 2-dimension dataset as an example to illustrate the process of the LDNOD algorithm, as is shown in Fig. 3. In Fig. 3(a), only 236 DRNN neighborhoods are constructed for a dataset with 1380 points for $\kappa = 15$. Fig. 3(b) shows how to obtain anomaly DRNN neighborhoods based on threshold $thresh = 4$. Fig. 3(c) shows all anomaly DRNN neighborhoods according to (b), and points in the other neighborhoods are labeled as non-outliers. Finally, 61 outliers are separated from the 37 anomaly DRNN neighborhoods, as is shown in Fig. 3(d).

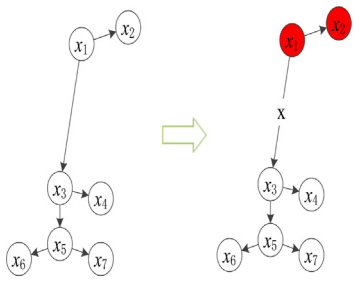

Fig. 2. Illustration of capturing an outlier or outliers from an anomaly neighborhood.

According to the process, the time complexity of the proposed algorithm is briefly analyzed as follows:

In Step 1: Before constructing DRNN neighborhoods, we need to compute a distance matrix first, which consumes $O(dN^2)$ time and $O(N^2)$space complexity, respectively. Each neighborhood need $(\kappa+1)log\kappa$ complexity, and $[N/\kappa+1, N-\kappa]$ neighborhoods cost $O((N-\kappa)(\kappa+1)log\kappa)$ time at worst.

In Step 2: At worst, $O(N-\kappa)$ time is required for scoring each DRNN neighborhood. And constant time complexity is needed to obtain anomaly neighborhoods.

In Step 3: Labeling outliers in very small set of neighborhoods also requires constant time complexity.

In summary, the time complexity is $O(dN^2+(N-\kappa)(\kappa+1)log\kappa+1)$ in total. As $N \gg \kappa$ and d, it can be rewritten as $O(N^2)$. Through the proposed algorithm is also a quadratic algorithm like traditional algorithms, some measures, such as sharing and scoring neighborhoods instead of data points, can improve efficiency to some extent.

3.3. Detection framework

To extend LDNOD to handle large-scale datasets, a detection framework by combining LDNOD and k-means (named LDNOD-km) is developed. The k-means is a popular partition method due to its linear time and space complexity, and it is widely used in large-scale outlier detection and clustering [28,29]. As is shown in Fig. 4, the framework mainly contains three phases:partition,detection, and aggregation.

First, we partition the entire data into m groups via k-means [31], where m is equal to $\sqrt{N}$. The k-means is a popular algorithm for partitioning data with high quality clusters and efficiency. Unlike the cluster-based outlier detection algorithms, the main purpose of this step is to get a small set of partitions with respect to original data instead of correct clusters. Generally speaking, m is much larger than the real number of clusters for a large-scale dataset. Therefore, pure partitions could be obtained by k-means, namely it contains only data objects from the same class besides outliers.

In the second phase, the LDNOD processes each partition of original data and output the outliers independently. Due to threshold manner, LDNOD does not need to gather and compare outlier scores of all data records.

Finally, we only need to aggregate outliers from each partition and output the final result. Note that, parameter κ should not be larger than the number of data objects of the smallest partition.

Note that we focus on local outliers in this paper, our divide-and-conquer strategy can work in the framework. It is true that extremely anomaly outliers (or global outliers) may affect the results of k-means and LDNOD, however, local outliers have little impact on k-means the LDNOD, and LDNOD-km.

4. Experiments

In this section, we conduct experiments on a variety of real datasets [32] to demonstrate the effectiveness, efficiency, and robustness of the LDNOD and its framework. Table 1 gives an overview of the datasets features. A widely used metric in outlier research, area under the ROC (AUC), is used to evaluate the detection performance. The real number of outliers is used as the top-n parameter for all methods except LDNOD. For LDNOD, the LNOF threshold value is set to 3 for all experiments. For the same dataset, parameters k and κ are set as the same values. Besides, other non-common parameters will be set as suggested by the corresponding papers.

All experiments are performed on a PC with an Intel i5-4460 CPU and 16.0 GB of RAM.

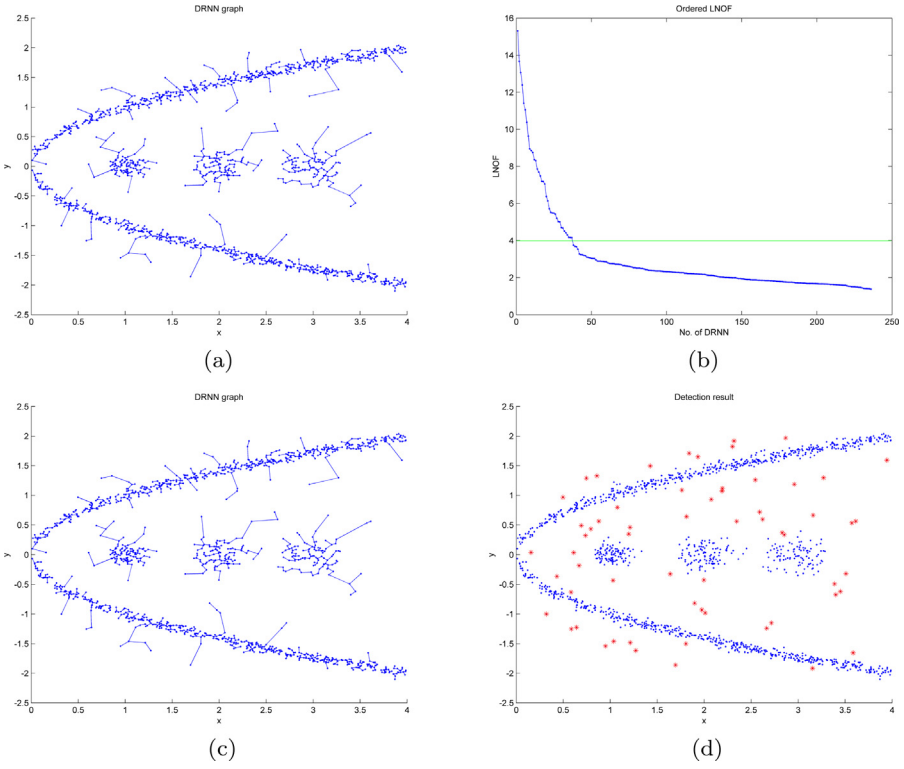

Fig. 3. An example in 2-dimension space. (a) DRNN graph. 236 DRNN neighborhoods are constructed for a dataset with 1380 points for $\kappa = 15$; (b) Ordered LNOF scores. The green line denotes the LNOF threshold value equals to 4; (c) 37 anomaly DRNN neighborhoods are obtained according to the threshold; (d) Detection result. 61 data points are detected as outliers (marked with red stars).

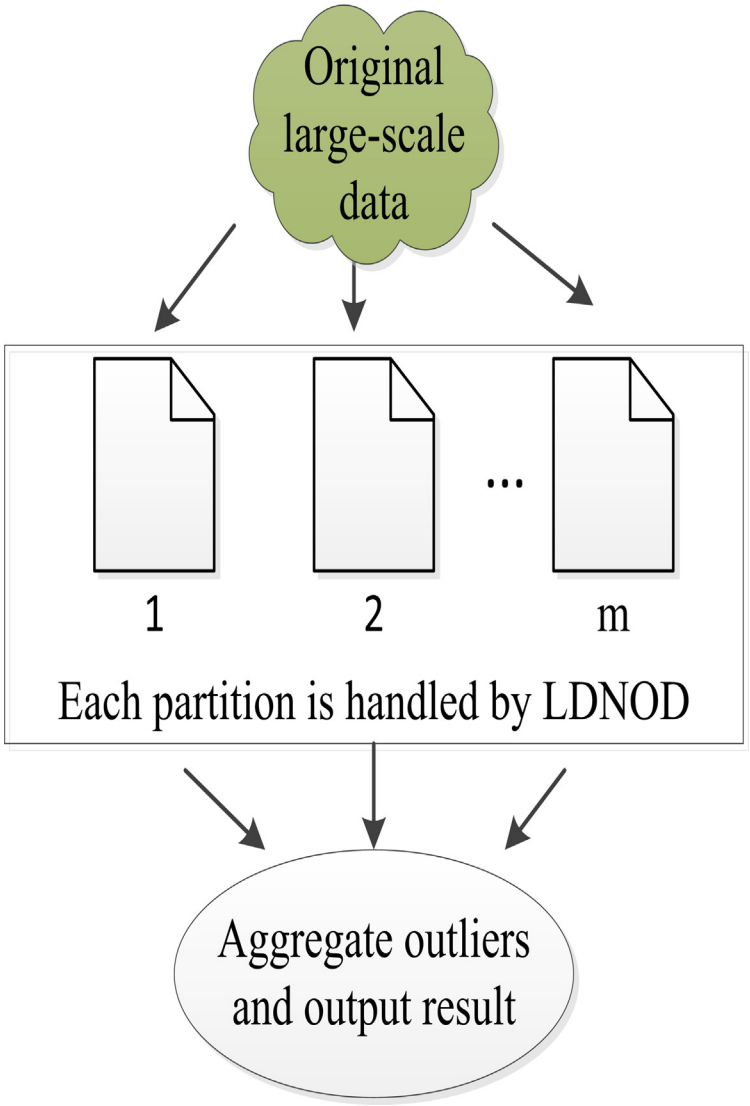

Fig. 4. Flowchart of LDNOD-km.

Table 1
Descriptions of 8 real datasets.

Datasets	Size	No. of Dim.	No. of Outliers
Shuttle	1,013	9	13
Cardiotocography	2,126	21	471
Waveform	3,443	21	100
Wilt	4,839	5	261
PageBlocks	5,473	10	560
Annthyroid	7,200	21	534
PenDigits	9,868	16	20
*KDDCup*99	60,632	38	246

4.1. Comparison with neighborhood based algorithms

In this experiment, we first compare LDNOD algorithm against five classical detection algorithms, which include LOF,COF, KNN [33], FastABOD [34], and LDOF.

Table 2 shows the comparison results of average AUC values between LDNOD and five baseline algorithms for neighborhood size from 10 to 50, and the best AUC values are highlighted in bold. As is shown in Table 2, our LDNOD method achieves the best AUC values on most of the ten datasets, even the other detection algorithms are performed with use of the correct number of outliers. Besides, LDOF and kNN only perform best on *Waveform* and *KDDCup*99, respectively. Especially for *Shuttle* dataset, our LDNOD outperforms the competing approaches.

Table 3 reports the comparison of average running time of the 6 approaches for a wide range of neighborhood parameter, and the best time are highlighted in bold. In Table 3, we can see that the LDNOD is the second fastest algorithm of the 6 methods for all tasting datasets. LOF is the fastest method in the evaluated methods, but LDNOD come close—there is a slightly difference between the two approaches. Despite theoretical computational complexity of all the 6 algorithms is $O(N^2)$, the actual running times are quite different from each other. We observed that the running times of the 6 methods vary widely with growth of data in volume, the fastest method (LOF) is faster from several to several dozen times than the slowest method (COF).

Fig. 5 shows the curves of AUC values of all methods for the 8 datasets for $10 \leqslant k$ (or κ) $\leqslant 50$. According to these graphs, we can see that the AUC curves of the proposed algorithms are flat for a wide bound of κ. However, the other methods are fluctuating with increases of k for most datasets. Although some approaches are insensitive to parameter k for some datasets, their AUC values

Table 2
Average AUC values of 6 methods for $10 \leqslant (kor\kappa) \leqslant 50$.

Datasets	LDNOD	LOF	COF	FastABOD	LDOF	KNN
Shuttle	**0.9295**	0.5727	0.6457	0.6524	0.6142	0.7855
Cardiotocography	**0.8756**	0.7258	0.7533	0.7459	0.7225	0.6785
Waveform	0.5822	0.5314	0.5516	0.5834	**0.6027**	0.5538
Wilt	**0.7345**	0.6257	0.6206	0.6112	0.6211	0.6102
PageBlocks	**0.7683**	0.6980	0.7128	0.7659	0.7017	0.6875
Annthyroid	**0.6754**	0.6357	0.6515	0.6458	0.6554	0.5956
PenDigits	**0.7248**	0.5857	0.6245	0.6435	0.6674	0.6218
*KDDCup*99	0.7825	0.6124	0.5764	0.5934	0.6211	**0.8275**

Table 3
Comparison of average running times of 6 methods for $10 \leqslant (kor\kappa) \leqslant 50$ (Unit: second).

Datasets	LDNOD	LOF	COF	FastABOD	LDOF	KNN
Shuttle	0.7517	**0.6765**	1.7863	1.5798	0.6789	0.8657
Cardiotocography	0.9454	**0.8965**	5.2654	4.5759	1.4487	1.3596
Waveform	2.4782	**2.1577**	12.1891	11.5425	3.1632	2.1578
Wilt	2.9517	**2.3324**	18.5679	15.8697	4.0784	4.5684
PageBlocks	3.6095	**2.5785**	21.7484	17.0728	4.8713	6.2157
Annthyroid	4.6648	**4.5279**	36.2735	28.7235	12.3791	18.4558
PenDigits	8.6584	**7.4125**	54.9494	44.5245	29.7606	33.4587
*KDDCup*99	110.87	**102.48**	8548.84	7485.65	1226.59	1854.54

are relatively small. Therefore, our approach is obviously robust than the competing methods.

In short, the proposed algorithm is effective and robust compared with several classic approaches. Moreover, its efficiency is also competitive among the 6 methods.

4.2. Comparison with large-scale detection algorithms

We also compare our LDNOD-km framework against several large-scale outlier detection algorithms, which are listed as follows:

(1) CBLOF [24]: cluster-based local outlier factor.
(2) LDCOF [25]: local density cluster-based outlier factor.
(3) MOA [35]: massive online analysis.
(4) SUOD [27]: scalable unsupervised outlier detection.

In CBLOF, LDCOF, and LDNOD-km, k-means is included and plays an important role. The MOA is an open source framework for big data stream mining, and SUOD is an acceleration framework for large-scale unsupervised outlier detector training and prediction. For LDNOD-km, CBLOF, and LDCOF, the number of clusters is set to $\sqrt{N}$ for k-means, and each approach is performed 20 trials on each dataset. Due to robustness of LDNOD, we set κ to 30 for all datasets for convenient. The LNOF threshold is also set to 3. The other parameters in the baseline methods are set as suggested by the corresponding papers or codes.

Tables 4 and 5 illustrate the performance and running times of each algorithm on different datasets, respectively. As is shown in Tables 4 and 5, the LDNOD-km significantly reduces the running times of the original LDNOD without significant loss of accuracy, even increased on some datasets. From Table 4, the LDNOD-km achieves 3 best AUC scores among 8 datasets, and the SUOD performs best on the remaining 5 datasets. Despite SUOD has better AUC values than our detection framework on average, it consumes more running times than ours, as is shown in Table 5. Moreover, we can see that the LDNOD-km is the fastest method and faster than the other 4 methods, especially for larger datasets. Thus, our detection framework is competitive on both accuracy and efficiency compared with several large-scale detection methods.

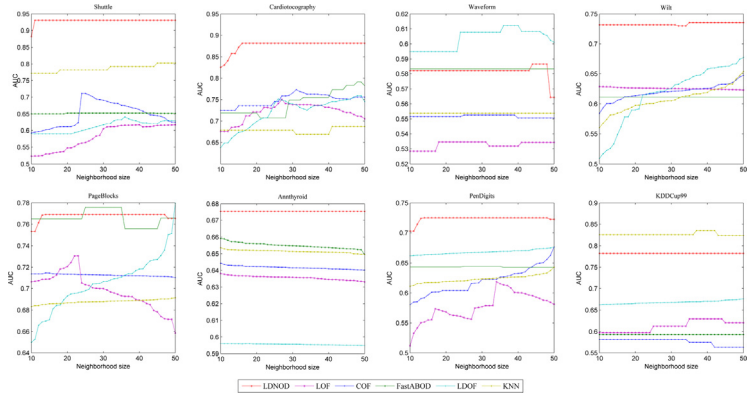

Fig. 5. AUC curves of the 6 algorithms for the 8 datasets for $10 \leqslant k(or\kappa) \leqslant 50$.

Table 4
Average AUC values of 5 algorithms on 8 datasets.

Datasets	LDNOD-km	CBLOF	LDCOF	MOA	SUOD
Shuttle	0.9197	0.8364	0.8066	0.7573	**0.9342**
Cardiotocography	**0.8622**	0.6155	0.6838	0.6529	0.8256
Waveform	0.6212	0.5235	0.5647	0.5891	**0.6896**
Wilt	**0.7125**	0.5563	0.6249	0.6347	0.6715
PageBlocks	0.7853	0.6457	0.6784	0.6658	**0.8568**
Annthyroid	0.6647	0.6146	0.6847	0.6755	**0.7584**
PenDigits	0.7186	0.6247	0.8465	0.7252	**0.8572**
*KDDCup*99	**0.8025**	0.5684	0.6253	0.6618	0.7675

Table 5
Average running times of 5 algorithms on 8 datasets (Unit: second).

Datasets	LDNOD-km	CBLOF	LDCOF	MOA	SUOD
Shuttle	**0.2234**	0.3474	0.3324	0.8224	0.7547
Cardiotocography	**0.3453**	0.8543	0.8475	1.2587	1.7487
Waveform	**0.8732**	1.5254	1.4547	3.1572	4.2574
Wilt	**0.9192**	2.1847	2.2054	3.5174	4.5895
PageBlocks	**1.2038**	3.2585	3.2147	4.4515	5.0145
Annthyroid	**1.2684**	4.2474	4.1214	5.2124	6.7648
PenDigits	**1.8529**	5.2475	5.2271	6.5749	7.3356
*KDDCup*99	**15.8723**	82.5484	82.4843	112.5548	151.6542

5. Conclusions

In this paper, a new local detection algorithm (LDNOD) and its framework (LDNOD-km) have been proposed. The LDNOD is insensitive to neighborhood parameter due to the stability of DRNN, which constructs neighborhood of an instance based on dynamic reference objects. Moreover, sharing neighborhoods of close objects and scoring each local region are designed, which can potentially reduce the running time of LDNOD. Because the LDNOD-km combines the benefits of both LDNOD and k-means, it is able to handle large-scale datasets efficiently without sacrificing accuracy. Finally, experimental results have demonstrated the effectiveness of LDNOD and its framework. In the future, we will further improve LDNOD-km and apply it to handle larger scale and high dimensional datasets.

Declarations of Interest

None.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61802360) and Chongqing Science and Technology project (KJZH17104 and CSTC2017rgun-zdyfx0040) and Chongqing Research Program of Basic Research and Frontier Technology (CSTC2019jcyj-msxmX0033).

References

[1] Hawkins D. Identification of outliers. Chapman and Hall; 1980.
[2] Barnett V, Lewis T. Outliers in statistical data. 3rd ed, 1994.
[3] Hodge V, Austin J. A survey of outlier detection methodologies. Artif Intell Rev 2004;22(2):85–126.
[4] Chandola V, Banerjee A, Kumar V. Anomaly detection: a survey. ACM Comput Surveys 2009;41(3):15.
[5] Campos GO, Zimek A, Sander J, Campello RJ, Micenkov B, Schubert E, Assent I, Houle ME. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. Data Min Knowl Discov 2016;30(4):891–927.
[6] Goldstein M, Uchida S. A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. PloS One 2016;11(4):e0152173.
[7] Domingues R, Filippone M, Michiardi P, Zouaoui J. A comparative evaluation of outlier detection algorithms: experiments and analyses. Pattern Recogn 2018;74:406–21.
[8] Weller-Fahy DJ, Borghetti BJ, Sodemann AA. A survey of distance and similarity measures used within network intrusion anomaly detection. IEEE Commun Surveys Tutor 2015;17(1):70–91.
[9] Ahmed M, Mahmood AN, Islam MR. A survey of anomaly detection techniques in financial domain. Future Gen Comput Syst 2016;55:278–88.
[10] Djenouri Y, Zimek A. Outlier detection in urban traffic data. In: Proceedings of the 8th international conference on web intelligence, mining and semantics. ACM; 2018.
[11] Yu R, He X, Liu Y. GLAD: group anomaly detection in social media analysis. ACM Trans Knowl Discov Data 2015;10(2):18.
[12] Riazi M, Zaiane O, Takeuchi T, Maltais A, G++nther J, Lipsett M. Detecting the onset of machine failure using anomaly detection methods.
[13] Zhou JT, Du J, Zhu H, Peng X, Liu Y, Goh RSM. AnomalyNet: an anomaly detection network for video surveillance. IEEE Trans Inf Forensics Secur 2019.
[14] Breunig MM, Kriegel HP, Ng RT, Sander J. May. LOF: identifying density-based local outliers. ACM SIGMOD Record 2000;29(2):93–104.
[15] Tang J, Chen Z, Fu A, Cheung D. Enhancing effectiveness of outlier detections for low density patterns. In: Chen MS, Yu P, Liu B, editors. Advances in knowledge discovery and data mining. vol. 2336 of Lecture Notes in Computer Science. Springer: Berlin/Heidelberg; 2002. pp. 535–548.
[16] Papadimitriou S, Kitagawa H, Gibbons PB, Faloutsos C. LOCI: Fast Outlier Detection Using the Local Correlation Integral. In: Proceedings of the 19th International Conference on Data Engineering. Los Alamitos, CA, USA: IEEE Computer Society Press; 2003. p. 315–326.
[17] Jin W, Tung A, Han J, Wang W. Ranking Outliers Using Symmetric Neighborhood Relationship. In: Ng WK, Kitsuregawa M, Li J, Chang K, editors. Advances in Knowledge Discovery and Data Mining. vol. 3918 of Lecture Notes in Computer Science. Springer, Berlin/ Heidelberg; 2006. p. 577–593.
[18] Kriegel HP, Krger P, Schubert E, LoOP Zimek A. Local outlier probabilities. In: Proceeding of the 18th ACM conference on information and knowledge management (CIKM-09). New York, NY, USA: ACM Press; 2009. p. 1649–52.
[19] Zhang K, Hutter M, Jin H. A new local distance–based outlier detection approach for scattered real-world data. Pacific-Asia Conference on Knowledge Discovery and Data Mining; 2009. pp. 813–822.
[20] Yiu ML, Mamoulis N. Reverse nearest neighbors search in ad hoc subspaces. IEEE Trans Knowl Data Eng 2007;19(3):412–26.
[21] Wang Chai. A pruning based continuous RKNN query algorithm for large k. Chin J Electr 2012;3:523–7.
[22] Zhu Qingsheng, Feng J, Huang J. Natural Neighbor: a self-adaptive neighborhood method without parameter K. Pattern Recogn Lett 80; 2016.
[23] Huang J, Zhu Q, Yang L, Feng J. A non-parameter outlier detection algorithm based on natural neighbor. Knowl-Based Syst 2016;92(C):71–7.
[24] He Z, Xu X, Deng S. Discovering cluster-based local outliers. Pattern Recogn Lett 2003;24(9–10):1641–50.
[25] Amer M, Goldstein M. Nearest-neighbor and clustering based anomaly detection algorithms for RapidMiner. In: Simon Fischer IM, editor. Proceedings of the 3rd RapidMiner Community Meeting and Conferernce (RCOMM 2012). Shaker Verlag GmbH; 2012. p. 1–12.
[26] Goldstein M, Dengel A. Histogram-based outlier score (HBOS): a fast Unsupervised Anomaly Detection Algorithm. In: Wlfl S, editor. KI-2012: Poster and Demo Track. Online; 2012. p. 59–63.
[27] Zhao Y, Ding X, Yang J, Bai H. Toward scalable unsupervised outlier detection. Workshops at the Thirty–Fourth AAAI Conference on Artificial Intelligence, 2020.

[28] Cai D, Chen X. Large scale spectral clustering via landmarkbased sparse representation. IEEE Trans Cybern 2015;45(8):1669–80.
[29] Huang D, Wang C-D, Wu J, Lai J-H, Kwoh CK. Ultra-scalable spectral clustering and ensemble clustering. IEEE Trans Knowl Data Eng 2019. doi: https://doi.org/10.1109/tkde.2019.2903410. 1–1.
[30] MacQueen J. Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics. Berkeley, Calif: University of California Press; 1967. p. 281–97.
[31] Cai D. 'Litekmeans: the fastest matlab implementation of kmeans', Available at:http://www.zjucadcg.cn/dengcai/Data/Clustering.html, 2011.
[32] Datasets:https://www.dbs.ifi.lmu.de/research/outlier-evaluation/DAMI/.
[33] Angiulli F, Pizzuti C. August. Fast outlier detection in high dimensional spaces. In: European Conference on Principles of Data Mining and Knowledge Discovery. p. 15–27.
[34] Kriegel HP, Schubert M, Zimek A. Angle-based outlier detection in high–dimensional data. In: ACM SIGKDD international conference on knowledge discovery and data mining.
[35] Bifet Albert, Holmes Geoff, Kirkby Richard, Pfahringer Bernhard. MOA: massive online analysis. J Mach Learn Res 2010;11:1601–4.