11-2015

LeSiNN: Detecting anomalies by Identifying least similar nearest neighbours

Guansong PANG
Singapore Management University, gspang@smu.edu.sg

Kai Ming TING

David ALBRECHT

LeSiNN: Detecting Anomalies by Identifying Least Similar Nearest Neighbours

Conference Paper · November 2015

DOI: 10.1109/ICDMW.2015.62

3 authors, including:

Guansong Pang
Singapore Management University
66 PUBLICATIONS **1,153** CITATIONS

SEE PROFILE

David W. Albrecht
Monash University (Australia)
72 PUBLICATIONS **1,299** CITATIONS

SEE PROFILE

LeSiNN: Detecting anomalies by identifying Least Similar Nearest Neighbours

Guansong Pang
Advanced Analytics Institute
University of Technology Sydney
Sydney, Australia
guansong.pang@student.uts.edu.au

Kai Ming Ting
School of Information Technology
Federation University Australia
Melbourne, Australia
kaiming.ting@federation.edu.au

David Albrecht
Clayton School of Information Technology
Monash University
Melbourne, Australia
david.albrecht@monash.edu

***Abstract*—We introduce the concept of Least Similar Nearest Neighbours (LeSiNN) and use LeSiNN to detect anomalies directly. Although there is an existing method which is a special case of LeSiNN, this paper is the first to clearly articulate the underlying concept, as far as we know. LeSiNN is the first ensemble method which works well with models trained using samples of *one* instance. LeSiNN has linear time complexity with respect to data size and the number of dimensions, and it is one of the few anomaly detectors which can apply directly to both numeric and categorical data sets. Our extensive empirical evaluation shows that LeSiNN is either competitive to or better than six state-of-the-art anomaly detectors in terms of detection accuracy and runtime.**

***Index Terms*—Least Similar Nearest Neighbours; *k*NN ; Anomaly Detection; Ensemble.**

I. Introduction

kNN (k-nearest neighbour) has been used as a core mechanism in many successful methods for anomaly detection. In density-based methods [1], [2], kNN is used as a density estimator to model density distribution, and identify anomalies as instances in low density regions. In distance-based methods [3]–[5], the underlying assumption is that anomalies have large distances to their nearest neighbours. So in these methods the distance to the k-th nearest neighbour or the average distance to the k nearest neighbours can be used as an anomaly score.

In this paper we introduce the concept of Least Similar Nearest Neighbour (LeSiNN) and use LeSiNN to detect anomalies directly, i.e., instances that have the least similar nearest neighbour are considered as anomalies. It is obvious that least similar nearest neighbours are all that is required to detect scattered anomalies, as scattered anomalies distribute far away from other instances in the feature space and thus have the smallest similarity to their nearest neighbours. However, least similar nearest neighbours are not sufficient to detect clustered anomalies. Clustered anomalies can have a very similar nearest neighbour and hence can be masked by other anomalies.

In order to reduce the masking effect of clustered anomalies, LeSiNN operates in small *random* samples of a data set instead of the full data set. Using small samples also substantially reduce the time complexity of nearest neighbour searching, from quadratic time complexity to linear time complexity.

Though based on distance, LeSiNN is different from existing kNN methods in three ways. First, LeSiNN only requires for each test instance to find the nearest neighbour. Whereas kNN methods often require to use $k > 1$. Second, most if not all, of the kNN methods focus on numeric data sets only. We show that LeSiNN performs well in both numeric and categorical data sets. Third, LeSiNN works best in an ensemble approach while kNN often operates as a single model. Though there are current distance-based methods (e.g., Sp [6]) which advocate the use of nearest neighbour. However, they operate as a single model too. We show that Sp is a special case of LeSiNN, and both Sp and LeSiNN can perform well in some situations using sample size of *one* instance!

LeSiNN has the following features:

- It is the first ensemble method which works well with models trained using samples of one instance!
- It is one of the few anomaly detectors which can apply directly to both categorical and numeric data sets.
- It has linear time complexity with respect to data size and the number of dimensions. Most state-of-the-art anomaly detectors have at least quadratic time complexity with respect to either data size or the number of dimensions.
- While kNN can be reduced to LeSiNN by using $k = 1$, this condition is rarely applied in practice because the theoretical analysis of kNN density estimators posits that $k > 1$ [7]. In contrast, LeSiNN operates using only nearest neighbours.

The rest of this paper is organised as follows. Section II reviews related work. We introduce LeSiNN in Section III, and the empirical evaluation results are presented in Section IV. Section V provides some discussions over the results, and Section VI concludes this paper.

II. Related work

Existing kNN-based anomaly detection methods, e.g., state-of-the-art distance-based methods [3]–[5] and density-based methods [1], [2], require distance computation, which has $O(n^2)$ time complexity, and thus cannot scale up to very large data sets. Although this cost can be reduced to $O(n \log n)$ when instances are preprocessed by an indexing scheme such as R^*-tree [8], most indexing schemes work in low dimensional data sets only; and they break down in high

dimensionality. Also, these indexing schemes are numeric data oriented only.

Recent research [6] has advocated the use of nearest neighbour in a method called Sp; and it is a special case of k-th nearest neighbour which employs a small sample. The authors [6] use a probabilistic analysis to suggest a plausible reason as to why a small sample is sufficient to detect anomalies using this method. We state their reason as follows: (i) Because anomalies constitute a small proportion in a data set, a small sample is likely to have normal instances only; thus, (ii) instances which have long distances to their nearest neighbours in this sample are likely to be anomalies. It shall be pointed out that [6] has emphasised point (i), but point (ii) was left implicit. We think the concept, i.e., the least similar nearest neighbours are anomalies, is the key. The concept can be applied by finding nearest neighbours in either the given data set or multiple random samples. Sp finds nearest neighbour in only one random sample and hence can perform very unstably.

Sp [6] has been reported to perform better than k-th nearest neighbour [3] and LOF [1] using numeric data sets. However, this comparison is unfair because these methods are known to be sensitive to k setting (see our results in Tables V and VII), and a fixed k was used in their experiments.

LOF-based ensembles [2] and iForest [9] are closely related to LeSiNN in the sense that they employ sampling techniques to build ensemble models for anomaly detection. An ensemble of LOF models, each built on a sample, was reported to improve the detection accuracy over single LOF model [2] but often at a cost of higher computation time than LOF, because it requires sufficiently large sampling size (e.g., 10% in [2]) to accurately approximate density using kNN [7]. iForest builds isolation trees on samples to isolate instances and detects anomalies using the average path length of isolated trees. iForest has similar time and space complexities as LeSiNN, but it is based only on numeric data, and there is no good method, so far, to extend it to categorical data.

Compared to methods for numeric data, significantly less research has been conducted for categorical data. Most anomaly detection methods [10]–[12] for categorical data are pattern-based algorithms which build on frequent or infrequent patterns. These methods, such as FPOF [10] and LOADED [12], need to search for frequent or infrequent itemsets to build detection models, which have time and space complexities exponential to data dimensionality. COMPREX [11] employs Minimum Description Length to avoid the costly frequent itemset search, but its time complexity is still quadratic in terms of data dimensionality. Compared to these methods, LeSiNN has linear time in terms of data size and data dimensionality. Also, at the core of these methods, is that they are based on density/frequency. In contrast, LeSiNN does not compute density, though a similarity measure is required.

III. Least Similar Nearest Neighbours: Concept and anomaly detector

This section is organised as follow: We provide our intuition of using least similar nearest neighbours (LeSiNN) to detect anomalies in Section III-A. The formal definitions of anomaly and anomaly score based on LeSiNN is given in Section III-B. The analyses of time and space complexities are described in Section III-C. The symbols and notations used are provided in Table I.

TABLE I
Symbols and notations

D	A data set with d attributes, where $\|D\| = n$
$\mathbf{x}$	An instance in D
$\eta_{\mathbf{x}}$	The nearest neighbour of $\mathbf{x}$.
$\mathcal{D}$	A sample of D, where $\|\mathcal{D}\| = \psi$
$sim(\cdot,\cdot)$	A similarity function
t	The number of samples

A. Intuition

We provide two examples to demonstrate the intuition of using LeSiNN to detect anomalies: one using real-world objects with categorical attributes and another using synthetic numeric data.

The first intuitive example is provided in Figure 1 (a). Using the characteristics of each object represented by four attributes: edible, colour, shape and texture, we can find the nearest neighbour of each object shown in Figure 1 (a). The nearest neighbour of the orange-coloured toy car is an orange because they share the same colour but nothing else, and all other fruits have no shared attributes with the toy car. In contrast, an orange is the nearest neighbour of another orange because they share four attributes, and all other fruits have less than four shared attributes with orange. Each of the other fruits and its nearest neighbour share two to four attributes. Using the number of shared attributes of objects as measure of similarity, we can to rank all objects in Figure 1 (a) in ascending order of least similar nearest neighbour. The toy car, which is intuitively the anomaly in this case, is ranked top because it has the least similar nearest neighbour, and oranges are ranked bottom since they have the most similar nearest neighbours.

The second example is based on a two-dimensional numeric data set with 200 instances, shown in Figure 1 (b). Figure 1 (c) shows the three curves of similarity to the nearest neighbour in three different subsets of the data set. The first curve uses the whole data set for the nearest neighbour search. The second curve uses only one instance, i.e., the instance at the centre of the distribution. While the third uses 10 random samples of size one. (The random sample method is to be described in Section III-B.) In each of the curves, $\mathbf{x}$ and $\mathbf{y}$, which are the outlying instances, have the smallest similarities to their nearest neighbours in all three curves. Whereas $\mathbf{z}$ is an instance inside the cluster and is more similar to its nearest neighbour than $\mathbf{x}$ and $\mathbf{y}$ are to their nearest neighbours.

B. Definitions of anomaly and anomaly score

We define anomaly and anomaly score as follows:

Definition 1: Anomalies are instances which are least similar to their nearest neighbours.

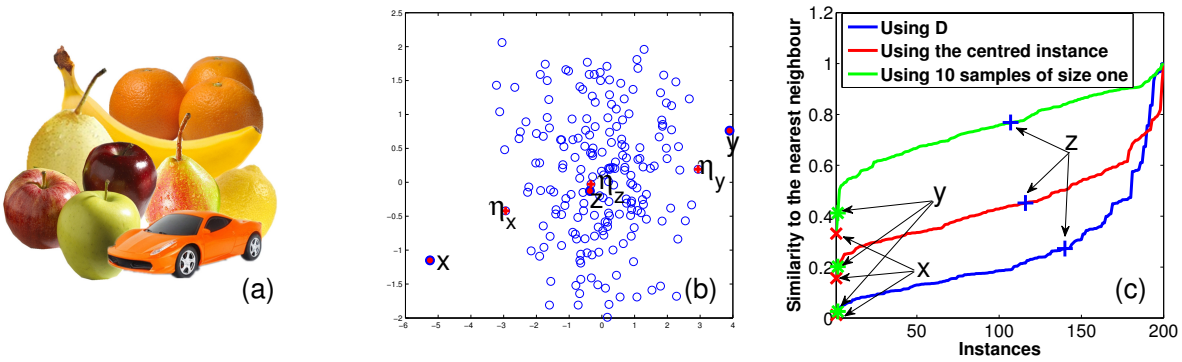

Fig. 1. Anomalies are less similar to their nearest neighbours than normal instances are to their nearest neighbours. (a) Intuition example using real-world objects. (b) A data set of a Gaussian distribution with 200 instances. $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ are example instances and $\eta_{\mathbf{x}}$, $\eta_{\mathbf{y}}$ and $\eta_{\mathbf{z}}$ are their nearest neighbours in D, respectively. (c) Three curves of instances sorted in ascending order according to the similarity to their nearest neighbours in D, the instance at the centre of the distribution, and 10 random samples of size one, respectively. This result reveals that $\mathbf{x}$ and $\mathbf{y}$, which are intuitively anomalies, are ranked as the top two instances which have the least similar nearest neighbours in all three sorted lists.

Definition 2: The similarity between $\mathbf{x} \in D$ and its nearest neighbour in $\mathcal{D}$ is defined as the nearest neighbour similarity of $\mathbf{x}$:

$$nn\_sim(\mathbf{x}|\mathcal{D}) = \max_{\mathbf{y} \in \mathcal{D}} \ sim(\mathbf{x}, \mathbf{y}) \tag{1}$$

Definition 3: The anomaly score for $\mathbf{x}$ is defined as the reciprocal of the average of its nearest neighbour similarity over t samples $\mathcal{D}_i$, $i = 1, 2, \cdots, t$:

$$score(\mathbf{x}) = (\frac{1}{t}\sum_{i=1}^{t} nn\_sim(\mathbf{x}|\mathcal{D}_i))^{-1} \tag{2}$$

Anomalies are instances which have the largest anomaly scores, i.e., they have the least similar nearest neighbours.

LeSiNN's ability to use small samples in an ensemble brings about three advantages. First, the ensemble significantly reduces its time complexity from n^2 to $n\psi$, where $\psi \ll n$. Second, small samples can reduce the masking effect of clustered anomalies; and this is a necessary step to detect clustered anomalies (this is the same requirement in iForest [9]). Third, with a sufficient large ensemble size, it produces a stable anomaly detector. In contrast, Sp [6] is an unstable detector because it uses only one small sample.

The similarity measure used depends on the attribute type. In this paper, we use the overlap similarity for data sets with categorical attributes, and (inverse) Euclidean distance for data sets with numeric attributes.

Overlap Similarity. The overlap similarity measure is defined as $sim(\mathbf{x}, \mathbf{y}) = \frac{1}{d}\sum_{i=1}^{d} I(x_i, y_i)$, where $I(x_i, y_i)$ is 1 when x_i is identical to y_i, and 0 otherwise.

Euclidean distance. Given two instances $\mathbf{x}$ and $\mathbf{y}$, their similarity based on Euclidean distance is defined as $sim(\mathbf{x}, \mathbf{y}) = \left(1 + \sqrt{\sum_{i=1}^{d}(x_i - y_i)^2}\right)^{-1}$.

C. Complexity analysis

In the training stage, LeSiNN needs t samples only, each having ψ instances. Thus, LeSiNN has time complexity $O(\psi t d)$. For each test instance, LeSiNN computes ψ pairwise similarities in each sample to find the nearest neighbour, which takes $O(\psi d)$, and so LeSiNN has time complexity $O(\psi t d)$ using t samples. To score n instances in a data set, LeSiNN has time complexity $O(n\psi t d)$. Therefore, the total time complexity of LeSiNN is $O(n\psi t d)$.

LeSiNN has space complexity $O(\psi t d)$ as it needs to store t samples, each having ψ instances.

TABLE II
TIME AND SPACE COMPLEXITIES OF LESINN AND SIX STATE-OF-THE-ART DETECTORS.

Detectors	Time complexity	Space complexity
LeSiNN	$O(n\psi t d)$	$O(\psi t d)$
iForest	$O(n\psi t)$	$O(\psi t)$
Sp	$O(n\psi d)$	$O(\psi d)$
kNN	$O(n^2 d)$	$O(nd)$
LOF	$O(n^2 d)$	$O(nd)$
FPOF	$O(n2^d)$	$O(2^d)$
COMPREX	$O(nd^2)$	$O(d^2)$

A comparison of time and space complexities between LeSiNN and six state-of-the-art detectors, including iForest [9], Sp [6], LOF [1], kNN [4], FPOF [10] and COMPREX [11], is provided in Table II.

Only LeSiNN and iForest are ensemble methods; and only LeSiNN, iForest and Sp employ samples. Since the sampling size and the ensemble size used are very small, i.e., $\psi \ll n$ and $t \ll n$, LeSiNN and Sp has linear time complexity with respect to both data size and dimensionality, and iForest has linear time complexity to data size. LeSiNN, iForest and Sp have constant space complexity with respect to data size. kNN, LOF, FPOF and COMPREX have much higher time and space complexities than LeSiNN. Though the time complexity of kNN and LOF can be reduced to $O(nd \log(n))$ when using some indexing scheme such as R^*-tree [8], most indexing schemes do not work on categorical data or data with high dimensionality. The time complexity of FPOF is linear to data size but exponential to dimensionality. The time complexity of COMPREX is linear to data size but quadratic to data dimensionality, though it becomes near-linear to the

dimensionality when many attributes are correlated [11].

IV. Experiments

A series of experiments were conducted to compare the detection performance of LeSiNN and its contenders on data sets with categorical attributes or numeric attributes. After a description of the experiment settings in Sections IV-A and IV-B, we compared LeSiNN with three state-of-the-art categorical data oriented detectors in categorical data sets in Section IV-C, and we then compared LeSiNN with three state-of-the-art numeric data oriented detectors in numeric data sets in Section IV-D. We examine the scalability of LeSiNN in Section IV-E and the effect of using different sampling sizes is presented in Section IV-F.

A. Contenders and their parameter settings

kNN [4], FPOF [10] and COMPREX [11] are selected as the contenders of LeSiNN in categorical data, and iForest [9], LOF [1] and Sp [6] are selected as the contenders in numeric data. These detectors are selected as our contenders, because they are state-of-the-art methods that are closely related to LeSiNN, as discussed in Section II.

Both LeSiNN and iForest employed $t = 50$ as the default ensemble size setting. Sp is equivalent to LeSiNN using $t = 1$. These three methods need to search for the best setting for ψ. The search for ψ was over the range 1, 2, 4, 8, 16, 32, 64, 128 and 256, except that Sp includes an additional $\psi = 20$ which was used as default in [6].

Following [10], FPOF employed 5 as the maximum length of itemsets by default, and we searched the minimum *support* threshold δ over the range 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8 and 0.9, and report the best results. COMPREX is a parameter-free method.

The size of neighbourhood or reference set (i.e., k) in kNN and LOF was searched over the range 1, 10, 20, 40, 60, 80, 150, 250, 300, 500, 1000, 2000, 3000 and 4000, and report the best results.

Note that LeSiNN and kNN used the same overlap similarity measure in handling categorical data, and LeSiNN and LOF used the same Euclidean distance in dealing with numeric data.

We implemented LeSiNN, kNN and Sp in WEKA [13]. FPOF was implemented using the *Apriori* algorithm in WEKA. iForest in the WEKA platform and LOF in the ELKI platform [14] were used in the experiments. COMPREX was implemented by [11] in Matlab.

In dealing with numeric data, R^*-tree indexing [8] was used by default in LOF; R^*-tree is not available in WEKA, and k-d tree [15] was used instead in LeSiNN and Sp. Note that tree based indexing methods do not work in categorical data because there are too many identical values in the attributes, so LeSiNN and kNN were employed without using indexing methods.

All the experiments were performed as a single-thread job at 2.27 GHz in a Linux cluster with 40GB memory.

B. Datasets and detection performance measure

Experiments were conducted on 18 real-world data sets from the UCI repository [16] and one synthetic data set having one normal cluster with two anomaly clusters, which was generated by the Mulcross data generator [17]. Following [2], [9], [11], the smallest class(es) or a small random subset of the smallest class was used as anomaly class against the largest class(es). A summary of the data sets is given in Table III. These data sets are from different application domains (e.g., health care, network security, image recognition and Internet advertising) and widely used in the literature [6], [9], [11]. The first seven data sets with mixed-type attributes were used as categorical attributes only or numeric attributes only in our two experiments. All numeric attributes are normalised before using in our experiments. We also generated multiple synthetic data sets for scaleup test in Section IV-E.

TABLE III
A summary of data sets used. *#num* and *#cate* denote the number of numeric and categorical attributes respectively. The *Anomaly class* column presents the anomaly class selected and its percentage in each data set. *Ad*, *Arrhy* and *Mammo* are for *Advertisements*, *Arrhythmia* and *Mammography*.

Data set	n	#num	#cate	Anomaly class
Linkage	5749132	2	5	match(0.36%)
Census	299285	7	33	50K+(6.20%)
CoverType	286048	10	44	cottonwood (0.96%)
Probe	64759	34	7	attack(6.43%)
U2R	60821	34	7	attack(0.37%)
Ad	3,279	3	1,555	ad(0.14%)
Arrhy	452	206	73	arrhythmia(14.60%)
Nursery	4648	0	8	very_recom (7.06%)
Chess	4580	0	6	zero(0.59%)
Mushroom	4429	0	22	poisonous(5.00%)
SolarFlare	1066	0	10	flare X(0.47%)
Http	567497	3	0	attack(0.39%)
Mulcross	262144	4	0	two clusters(10.00%)
Smtp	95156	3	0	attack(0.03%)
Shuttle	49097	9	0	classes 2,3,5,6,7 (7.15%)
Mammo	11183	6	0	class 1(2.32%)
Satimage	6435	36	0	crop(10.92%)
Isolet	730	617	0	class Y (1.37%)
Mfeat	410	649	0	digit 0 (2.44%)

The area under ROC curve (AUC) [18] was used as a measure of effectiveness for the anomaly ranking produced by an anomaly detector. Higher AUC indicates better detection performance. We also recorded the runtime to compare their efficiency. The AUC and runtime results were averaged over 10 runs for all randomised methods. In this paper, confidence intervals are based on the average AUC over 10 runs and its two standard errors; so the statistical significance statement is at 95% confidence level.

We employed a commonly used performance evaluation method for unsupervised anomaly detection techniques [19]. Specifically, we trained and evaluated detection models on the same data set, but it is assumed that class labels are unavailable in the training stage. The class labels are only used to compute AUC in the evaluation stage.

C. Detection performance on categorical data

The best detection performance of LeSiNN, kNN, FPOF and COMPREX is shown in Table IV. LeSiNN is significantly better than kNN in seven data sets, draws in two and loses in one. LeSiNN outperforms FPOF significantly in four data sets, with four draws and one loss. Note that FPOF cannot work in the high dimensional data set, *Arrhythmia* and *Advertisements*, due to an out-of-memory exception error resulting from its high space complexity, even though they are small data sets, e.g., *Arrhythmia* has no more than 500 instances. LeSiNN outperforms COMPREX significantly in 6 out of 10 data sets, with two draws and two losses. Note that we cannot obtain the results of kNN and COMPREX in *Linkage* and *Advertisements*, respectively, within three weeks, due to their high time complexity with respect to data size and data dimensionality respectively.

TABLE IV
AUC DETECTION PERFORMANCE OF LESINN, kNN, FPOF AND COMPREX ON CATEGORICAL DATA.

Data set	LeSiNN	kNN	FPOF	COMPREX
Linkage	0.9974±0.0005	na	0.9978	0.9973
Census	0.6225±0.0029	0.6209	0.6148	0.5046
CoverType	0.9960±0.0022	0.9446	0.9965	0.9936
Probe	0.9856±0.0014	0.9522	0.9867	0.9790
U2R	0.9916±0.0006	0.9880	0.9203	0.9893
Ad	0.8008±0.0019	0.7924	na	na
Arrhy	0.6921±0.0008	0.6667	na	0.6848
Nursery	1.0000±0.0000	1.0000	1.0000	1.0000
SolarFlare	0.9772±0.0010	0.9642	0.9791	0.9793
Mushroom	0.9972±0.0004	0.9982	0.9400	0.9359
Chess	0.9900±0.0024	0.9318	0.9290	0.9943
#na	0	1	2	1
#wins/draws/losses		7/2/1	4/4/1	6/2/2

The runtimes and the best parameters of the four detectors are reported in Table V. LeSiNN runs three to five orders of magnitude faster than kNN in large data sets, i.e, *Census*, *CoverType* and *Probe*. LeSiNN is significantly faster than FPOF in all three data sets with more than 20 attributes, *Census*, *CoverType* and *Mushroom*; LeSiNN is faster by a factor of more than 1,900, 348,000 and 50, respectively. LeSiNN runs slower than FPOF (but in the same order of magnitude) in the largest data set, *Linkage*, which has only five dimensions, as FPOF has lower linear time complexity than LeSiNN with respect to data size. Because LeSiNN and COMPREX were implemented in different programming languages. we will compare them in equal footing in the scaleup test in Section IV-E.

LeSiNN obtains its best performance with $\psi \leq 16$ in 7 out of the 11 data sets, while kNN requires to search for a wide range of values to get its best performance. FPOF works best in most data sets with $\delta \leq 0.1$, as a small δ is normally needed in order to obtain a sufficient number of frequent patterns.

D. Detection performance on numeric data

The AUC performance detection comparison of LeSiNN, iForest, LOF and Sp is presented in Table VI. LeSiNN is competitive to or significantly better than iForest in 14 out of 15 data sets. LeSiNN performs comparably to or significantly better than LOF in 9 out of 14 data sets. We cannot obtain the results of LOF in the two largest data sets, *Linkage* and *Http*, due to an out-of-memory exception error, resulting from high space complexity of R^*-tree indexing. LeSiNN outperforms Sp significantly in six data sets, with nine draws and no loss; LeSiNN performs stably and has small standard errors over different runs in each data set, whereas the detection performance of Sp fluctuates a lot in many data sets. Note that LeSiNN, iForest and Sp can detect the two anomaly clusters in *Mulcross*, while LOF fail to detect them. This is because the neighbourhood size of LOF requires to be larger than the size of anomaly cluster in order to detect clustered anomalies, but the size of the two clusters in *Mulcross* is far larger than the k search range used in LOF.

TABLE VI
AUC PERFORMANCE COMPARISON OF LESINN, IFOREST, LOF AND SP ON NUMERIC DATA.

Data set	LeSiNN	iForest	LOF	Sp
Linkage	0.9975±0.0000	0.9974±0.0000	na	0.9577±0.0389
Census	0.7808±0.0334	0.7906±0.0247	0.6690	0.6965±0.0049
CoverType	0.9275±0.0032	0.8726±0.0173	0.9781	0.8231±0.0560
Probe	0.9976±0.0001	0.9747±0.0081	0.6321	0.9979±0.0005
U2R	0.9878±0.0005	0.9860±0.0015	0.8873	0.9871±0.0008
Ad	0.7389±0.0056	0.6957±0.0069	0.7435	0.6623±0.1500
Arrhy	0.8256±0.0025	0.8164±0.0106	0.8296	0.7993±0.0329
Http	1.0000±0.0000	0.9997±0.0001	na	1.0000±0.0000
Mulcross	1.0000±0.0000	0.9979±0.0012	0.6035	0.9411±0.0996
Smtp	0.8873±0.0026	0.8834±0.0058	0.9535	0.8582±0.0333
Shuttle	0.9897±0.0006	0.9962±0.0007	0.9809	0.9104±0.1039
Mammo	0.8464±0.0061	0.8557±0.0101	0.8644	0.8113±0.0175
Satimage	0.9973±0.0006	0.9836±0.0027	0.9934	0.8386±0.0626
Isolet	1.0000±0.0000	0.9997±0.0003	1.0000	0.9996±0.0004
Mfeat	0.9808±0.0013	0.9401±0.0124	0.9800	0.8533±0.0943
#na	0	0	2	0
#wins/draws/losses		8/6/1	6/3/4	6/9/0

The runtimes and best parameters of the four detectors are reported in Table VII. LeSiNN runs significantly faster than LOF in the two large data sets, *Census* and *Probe*, by a factor of more than 800 and 300, respectively. LeSiNN is slower than iForest by a factor of between 10 and 50 in larger data sets (e.g., *CoverType*) or high-dimensional data sets (e.g., *Isolet* and *Mfeat*); LeSiNN is slower than Sp by a factor of between 10 and 80 in larger data sets. LeSiNN and Sp obtain their best performance using $\psi \leq 16$ in 14 and 12 data sets, respectively; while iForest requires much larger ψ, e.g., 256, in most data sets. LOF requires to search for a wide range of values to get its best performance.

It is interesting to note that both LeSiNN and Sp using $\psi = 1$ can obtain the best results in data sets with different characteristics, e.g., data sets with clustered anomalies like *Mulcross*, large data sets *Http* and *Probe*.

We have a more direct comparison between LeSiNN and Sp such that $t\psi$ used in LeSiNN and Sp is equal (i.e., the number of instances they use in the training stage is the same): LeSiNN using $\psi = 1$ with varying t values, and Sp (equivalent

TABLE V
RUNTIMES AND BEST PARAMETERS FOR LESINN, kNN, FPOF AND COMPREX ON CATEGORICAL DATA.

	Runtime (in seconds)				Best parameter		
Data set	LeSiNN	kNN	FPOF	COMPREX	LeSiNN (ψ)	kNN (k)	FPOF (δ)
Linkage	129	na	50	10,442	8	na	0.5
Census	45	68,722	89,168	640	16	3,000	0.1
CoverType	3	143,958	1,044,166	3,609	1	3,000	0.1
Probe	0.6	93,470	1.8	826	4	3,000	0.2
U2R	8	58,359	5	456	64	2,000	0.05
Ad	269	62	na	na	256	10	na
Arrhy	0.02	0.16	na	494	4	150	na
Nursery	0.05	6.25	0.34	18.83	2	500	0.1
SolarFlare	0.02	0.29	0.35	6.30	2	250	0.1
Mushroom	10	6	520	69	256	10	0.05
Chess	0.56	2.87	0.17	22.76	64	60	0.05

TABLE VII
RUNTIMES AND BEST PARAMETERS FOR LESINN, IFOREST, LOF AND SP ON NUMERIC DATA

	Runtime (in seconds)				Best parameter			
Data set	LeSiNN	iForest	LOF	Sp	LeSiNN (ψ)	iForest (ψ)	LOF (k)	Sp (ψ)
Linkage	328	141	na	8	8	256	na	4
Census	8	2	6,787	2	1	8	80	256
CoverType	185	7	6,866	3	128	256	3,000	128
Probe	4	2	1,351	0.1	1	128	4,000	1
U2R	16	2	753	0.2	8	256	500	2
Ad	0.3	0.1	4	0.02	16	16	1,000	1
Arrhy	0.2	0.1	0.1	0.02	2	128	80	1
Http	12	14	na	1	1	256	na	1
Mulcross	6	1	542	0.3	1	16	40	1
Smtp	30	3	350	0.5	256	256	1,000	256
Shuttle	4	1	418	0.1	8	128	4,000	2
Mammo	0.4	0.1	16	0.1	2	64	150	128
Satimage	1.4	0.1	16	0.05	8	64	2,000	16
Isolet	1	0.05	2	0.07	1	256	20	4
Mfeat	2	0.05	1	0.05	16	256	80	4

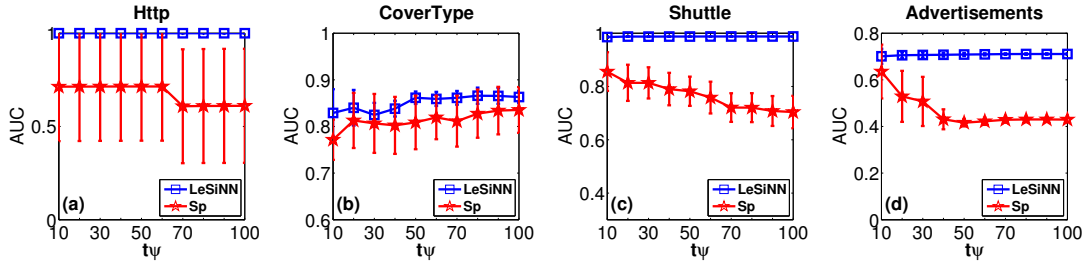

Fig. 2. LeSiNN ($\psi = 1$ with varying t) versus Sp (with varying ψ) such that $t\psi$ is the same.

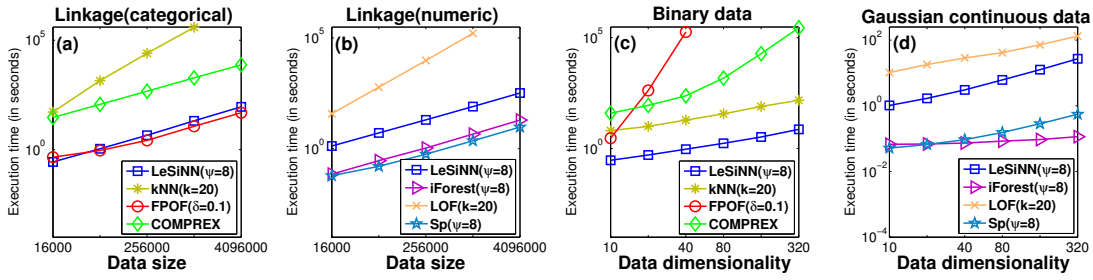

Fig. 3. Scaleup test of LeSiNN, using kNN, FPOF and COMPREX as baselines in categorical data; iForest, LOF and Sp as baselines in numeric data. Figures (a) and (b) are the results of the data size scaleup tests. Figures (c) and (d) are the results of the dimensionality scaleup tests.

to LeSiNN with $t = 1$) with varying ψ values. The results in Figure 2 shows that LeSiNN performs comparably to or significantly better than Sp using different combinations of $t\psi$ in all four data sets; and LeSiNN is much more stable too with lower standard errors. Similar results can also be found in other data sets.

E. Scalability examination

We examined the scalability of LeSiNN with respect to data size using five subsets of the largest data set *Linkage*. The smallest subset contains 16,000 instances, and subsequent subsets are increased by a factor of four, until the largest subset which contains 4,096,000 instances. Only one attribute type is used when examining the categorical data oriented methods or the numeric data oriented methods.

The scalability of LeSiNN with respect to dimensionality was tested using synthetic binary and Gaussian continuous data sets in categorical data and numeric data, respectively. The data sets contain the same number of instances, i.e., 10,000 instances. The data set with the lowest dimensionality contains 10 attributes, and subsequent data sets are increased by a factor of two, until the data set with the highest dimensionality contains 320 attributes.

The scaleup test results are presented in Figure 3. Figure 3 (a) shows that LeSiNN, FPOF and COMPREX have runtime linear to the data size, and run two to three orders of magnitude faster than kNN. Figure 3 (b) shows that LeSiNN, iForest and Sp have linear time complexity to data size, and run at least two orders of magnitude faster than LOF. LeSiNN runs slower than iForest because it needs to search for the nearest neighbours; LeSiNN is slower than Sp because LeSiNN uses multiple samples while Sp uses only one sample. Note when the data size reaches 4,096,000, we cannot get the runtime of kNN within three weeks, and LOF runs out of memory.

For scalability with respect to data dimensionality, Figure 3 (c) shows that both LeSiNN and kNN have runtime linear to the data dimensionality, and run up to five orders of magnitude faster than FPOF. Note that the space complexity of FPOF increases exponentially with increasing dimensions, and FPOF runs out of memory when the number of dimensions was increased to 80. The runtime of COMPREX increases by a factor of more than 7,000 when the dimensionality increases by a factor of 32; while that of LeSiNN increases by less than 30. Therefore, though LeSiNN and COMPREX were implemented in different programming languages, the runtime increasing ratio indicates that LeSiNN runs significantly faster than COMPREX by a factor of more than 200. In numeric data, Figure 3 (d) shows that LeSiNN, LOF and Sp have linear time complexity to data dimensionality, while the time complexity of iForest is constant to data dimensionality since it works on a few randomly selected attributes only.

F. The effect of varying ψ

The detection performance of LeSiNN with varying ψ values is shown in Figure 4. It shows that LeSiNN performs stably and obtains the best results using $\psi \leq 16$ in three out of four data sets. A larger ψ is needed in order to detect anomalies in data sets with multiple normal clusters, e.g., *Mushroom* with multiple edible mushroom species. Due to the space limitation, we only present some typical results here. Similar results can also be found in other data sets.

V. Discussion

Parameter ψ. The probability for the inclusion of a scattered anomaly into a sample is $1 - \binom{n-1}{\psi}/\binom{n}{\psi}$, which is close to zero as long as $\psi \ll n$. Thus, scattered anomalies have the least similar nearest neighbours in each of these samples. The occurrence probability of clustered anomalies in the samples increases with the cluster size. To detect clustered anomalies, especially large clustered anomalies, ψ is required to be sufficiently small in order to avoid sampling clustered anomalies into the samples. As illustrated in Figure 4 (c), the detection performance of LeSiNN in *Mulcross* drops quickly when ψ gets larger.

For data sets with unimodal distribution, ψ can be as small as one to detect all the anomalies, such as the results on *Http*, *Mulcross* and *Probe* in Tables VI and VII. ψ is required to be larger in order to perform well in multi-modal data sets.

CoverType produces two interesting results: In the data set version of 44 categorical attributes, LeSiNN with $\psi = 1$ produces AUC=0.9960; and in the data set version of 10 numeric attributes, LeSiNN needs to use $\psi = 128$ to produce AUC=0.9275. This indicates that the categorical version has a single mode and the numeric version has multiple modes.

In practice, based on our results in Sections IV-C and IV-D, in order to obtain favourable detection accuracy in data sets with different types of anomalies, LeSiNN is suggested to employ $\psi \leq 16$.

Parameter t. Our results have shown that the performance of LeSiNN converges quickly with respect to the ensemble size t, and there is normally no statistically significant difference in detection accuracy when $t \geq 30$. $t = 50$ is recommended for LeSiNN. As expected with any ensemble method, a small ensemble size (i.e., small t value) will have high variance. This is the reason why Sp is very unstable.

LeSiNN versus kNN methods. Compared to LeSiNN, kNN methods, such as kNN and LOF, can also detect scattered anomalies successfully using $k = 1$, i.e., the nearest neighbour. However, in order to detect clustered anomalies, k needs to be as large as the size of anomalous clusters in kNN methods, while only small ψ is required in LeSiNN. Therefore, kNN methods are much more sensitive to the setting of k than LeSiNN to ψ in real-world data sets having scattered anomalies or/and clustered anomalies.

As discussed in Section II, some effort have been made to using kNN methods in an ensemble approach, e.g., LOF-based ensemble using sampling techniques in [2]. As shown in [2], such ensembles could gain some improvement in terms of effectiveness but often at a cost of higher computation time, because large sample sizes are required to accurately approximate density using kNN. We have also tried to use k nearest

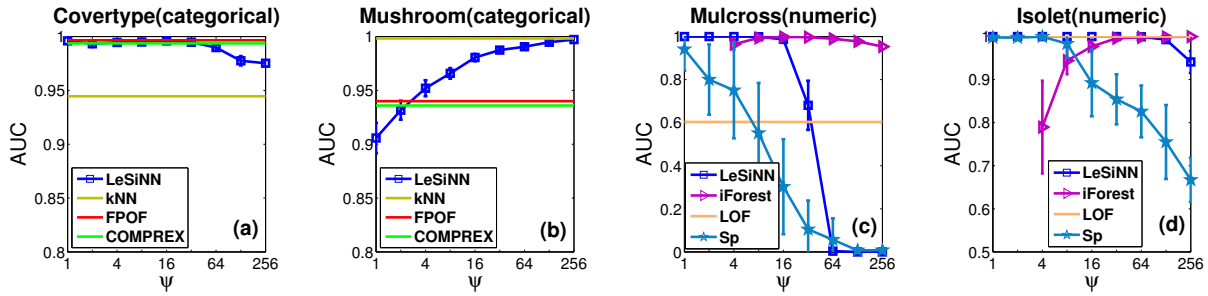

Fig. 4. Detection performance of LeSiNN with varying ψ values in 1, 2, 4, 8, 16, 32, 64, 128, 256. In categorical data sets, the best performance of kNN, FPOF and COMPREX are used as baselines. In numeric data, the best performance of LOF, and the performance of iForest and Sp with varying ψ values are used as baselines. Note that iForest must use sample size larger than two in order to build its isolation trees.

neighbours to replace nearest neighbours in LeSiNN, but it worked significantly worse than using nearest neighbours.

Relative measure. Note that LeSiNN is in a disadvantage position in comparison with LOF because LOF employs a relative measure and LeSiNN does not. LeSiNN employing a relative measure is expected to perform better than the results presented here.

VI. Conclusion and Future Work

The concept of LeSiNN provides a new perspective in understanding nearest neighbour methods. The use of LeSiNN is certainly simpler and more direct than the ones based on kNN or density. As a result, it is one of the most efficient anomaly detectors in terms of time and space complexities which can easily scale up to large data size and high dimensionality. As far as we know, this is the first ensemble method which works well with models trained from one instance!

LeSiNN is one of few anomaly detection algorithms which can detect anomalies in both categorical and numeric data.

We show that LeSiNN performs comparably to or significantly better than all three state-of-the-art categorical data oriented anomaly detectors and LOF in terms of AUC; and LeSiNN runs at least two orders of magnitude faster than these four methods. Compare with more efficient iForest and Sp, LeSiNN outperforms both of them significantly in most data sets in terms of AUC.

This paper empirically shows the effectiveness and efficiency of LeSiNN only. We are interested in investigating the theoretical support for LeSiNN in our future work.

Acknowledgment

This work is supported in part by Monash University Postgraduate Publication Award, awarded to Guansong Pang; and it is partially supported by a grant from the U.S. Air Force Research Laboratory, under agreement # FA2386-13-1-4043, awarded to Kai Ming Ting.

References

[1] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," *ACM Sigmod Record*, vol. 29, no. 2, pp. 93–104, 2000.

[2] A. Zimek, M. Gaudet, R. J. Campello, and J. Sander, "Subsampling for efficient and effective unsupervised outlier detection ensembles," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2013, pp. 428–436.

[3] F. Angiulli and C. Pizzuti, "Fast outlier detection in high dimensional spaces," in *Principles of Data Mining and Knowledge Discovery*. Springer, 2002, pp. 15–27.

[4] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," *ACM SIGMOD Record*, vol. 29, no. 2, pp. 427–438, 2000.

[5] M. Wu and C. Jermaine, "Outlier detection by sampling with accuracy guarantees," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2006, pp. 767–772.

[6] M. Sugiyama and K. Borgwardt, "Rapid distance-based outlier detection via sampling," *Advances in Neural Information Processing Systems*, pp. 467–475, 2013.

[7] K. Fukunaga, *Introduction to statistical pattern recognition*, 2nd ed. Academic press, 1990.

[8] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger, "The R*-tree: An efficient and robust access method for points and rectangles," in *Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data*. New York, NY, USA: ACM, 1990, pp. 322–331.

[9] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation-based anomaly detection," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 6, no. 1, pp. 3:1–3:39, 2012.

[10] Z. He, X. Xu, Z. J. Huang, and S. Deng, "FP-outlier: Frequent pattern based outlier detection," *Computer Science and Information Systems/ComSIS*, vol. 2, no. 1, pp. 103–118, 2005.

[11] L. Akoglu, H. Tong, J. Vreeken, and C. Faloutsos, "Fast and reliable anomaly detection in categorical data," in *Proceedings of the 21st ACM international conference on Information and knowledge management*. ACM, 2012, pp. 415–424.

[12] A. Ghoting, M. E. Otey, and S. Parthasarathy, "LOADED: Link-based outlier and anomaly detection in evolving data sets," in *IEEE International Conference on Data Mining*. IEEE, 2004, pp. 387–390.

[13] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: An update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[14] E. Achtert, H. Kriegel, E. Schubert, and A. Zimek, "Interactive data mining with 3d-parallel-coordinate-trees," in *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2013, New York, NY, USA, June 22-27, 2013*, 2013, pp. 1009–1012.

[15] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, 1975.

[16] K. Bache and M. Lichman, "UCI machine learning repository," *URL http://archive. ics. uci. edu/ml*, 2013.

[17] D. M. Rocke and D. L. Woodruff, "Identification of outliers in multivariate data," *Journal of the American Statistical Association*, vol. 91, no. 435, pp. 1047–1061, 1996.

[18] D. J. Hand and R. J. Till, "A simple generalisation of the area under the ROC curve for multiple class classification problems," *Machine Learning*, vol. 45, no. 2, pp. 171–186, 2001.

[19] C. C. Aggarwal, *Outlier analysis*. Springer, 2013.