Sparse Modeling-based Sequential Ensemble Learning for Effective Outlier Detection in High-dimensional Numeric Data

Guansong Pang[†] and **Longbing Cao**[†] and **Ling Chen**[†] and **Defu Lian**[*] and **Huan Liu**[‡]
[†]School of Software, University of Technology Sydney, Australia
[*]School of Computer Science and Engineering, University of Electronic Science and Technology of China, China
[‡]Computer Science and Engineering, Arizona State University, USA
pangguansong@gmail.com, {longbing.cao;ling.chen}@uts.edu.au, dove.ustc@gmail.com, huan.liu@asu.edu

Abstract

The large proportion of irrelevant or noisy features in real-life high-dimensional data presents a significant challenge to subspace/feature selection-based high-dimensional outlier detection (a.k.a. outlier scoring) methods. These methods often perform the two dependent tasks: relevant feature subset search and outlier scoring independently, consequently retaining features/subspaces irrelevant to the scoring method and downgrading the detection performance. This paper introduces a novel sequential ensemble-based framework SEMSE and its instance CINFO to address this issue. SEMSE learns the sequential ensembles to mutually refine feature selection and outlier scoring by iterative sparse modeling with outlier scores as the pseudo target feature. CINFO instantiates SEMSE by using *three successive recurrent components* to build such sequential ensembles. Given outlier scores output by an existing *outlier scoring* method on a feature subset, CINFO first defines a *Cantelli*'s inequality-based *outlier thresholding* function to select outlier candidates with a false positive upper bound. It then performs lasso-based *sparse regression* by treating the outlier scores as the target feature and the original features as predictors on *the outlier candidate set* to obtain a feature subset that is tailored for the outlier scoring method. Our experiments show that two different outlier scoring methods enabled by CINFO (i) perform significantly better on 11 real-life high-dimensional data sets, and (ii) have much better resilience to noisy features, compared to their bare versions and three state-of-the-art competitors. The source code of CINFO is available at https://sites.google.com/site/gspangsite/sourcecode.

Introduction

High-dimensional data is ubiquitous in broad real-life applications, e.g., thousands of molecular or gene expression features in bioinformatics, and millions of trading behaviors in stock market surveillance. Identifying outliers that deviate significantly from the majority of data objects can provide important insights into these applications. For example, the detection of outlying gene expressions can facilitate early treatment of diseases (Aggarwal 2017); and the recognition of abnormal trading can signal abusive and manipulative trading practices (Cao, Ou, and Yu 2012) .

However, identifying outliers in high-dimensional numeric data is a challenging task. Specifically, high-dimensional data poses the two major challenges below. (i) It often contains a large percentage of irrelevant features. The irrelevant features mask outliers as normal objects and thus they are 'noise' to outlier detection (a.k.a. outlier scoring). Irrelevant features also form a major cause of the 'curse of dimensionality' (Zimek, Schubert, and Kriegel 2012). (ii) The number of candidate feature subsets increases exponentially as the dimensionality increases, leading to great difficulty in a complete search of the feature space.

To detect outliers in the above high-dimensional data, subspace/feature selection-based methods (Lazarevic and Kumar 2005; Keller, Muller, and Bohm 2012; Noto, Brodley, and Slonim 2012; Paulheim and Meusel 2015) are the major solutions. They search for relevant feature subset(s) to apply off-the-shell outlier detection methods on these relevant feature subset(s) to alleviate the negative effect brought by irrelevant features. However, these methods often separate the subspace search from the subsequent outlier scoring methods. Consequently, they may retain feature subsets that are irrelevant to the scoring methods and the resultant detection performance of the outlier scoring methods is largely biased. Due to the unsupervised nature of outlier detection and the huge search space, it is challenging to involve the outlier scoring methods when searching the feature subset(s).

In this paper, we introduce a novel SparsE Modeling-based Sequential Ensemble learning (*SEMSE* for short) framework for outlier detection in high-dimensional numeric data. Specifically, SEMSE first uses a given *outlier scoring* method to compute the outlier scores of data objects, and defines an *outlier thresholding* function to identify a set of outlier candidates. SEMSE then performs *sparse regression* on the outlier candidate set by treating the outlier scores as a target feature and the original features as predictors to select the most relevant features w.r.t. the outlier scores. This process is referred to as *fragmentary* sparse modeling to highlight that the sparse regression is built on a small data subset (i.e., *the outlier candidate set*) rather than the full data set. SEMSE finally applies the same given outlier detector to the data with the selected features to produce a refined outlier scoring. The above three steps are iteratively performed to produce a set of outlier scores until the loss function of the sparse regression does not decrease.

Essentially, this learning procedure integrates the two correlated tasks: feature selection and outlier detection, and iteratively and mutually refines them, resulting in a set of *dependent* outlier detection (or feature selection) models which are commonly known as *sequential ensemble* (Freund and Schapire 1995). This enables SEMSE to produce feature subsets that are tailored for the outlier scoring method. Single sequential ensemble may perform unstably in data sets with many noisy features. We therefore have a boostrap aggregating (i.e., *bagging*) (Breiman 1996) of the sequential ensembles (i.e., an ensemble of sequential ensembles) to further enhance its capability and stability.

We further implement SEMSE by defining a *Cantelli*'s INequality-based Fragmentary lassO, termed CINFO, to build the sequential ensembles. Specifically, CINFO first defines a *Cantelli*'s inequality (Dubhashi and Panconesi 2009) based outlier thresholding function to select the outlier candidates, and further applies lasso-based fragmentary sparse regression on the outlier candidate set to obtain the relevant feature subset. Two diverse subsampling-based outlier scoring methods, namely LeSiNN (Pang, Ting, and Albrecht 2015) and iForest (Liu, Ting, and Zhou 2012) that respectively work on the full space and random subspaces of the input data, are respectively used to obtain the outlier scores to demonstrate the flexibility of SEMSE.

Accordingly, this paper makes two major contributions:

1. We introduce a novel sequential ensemble learning framework SEMSE for identifying outliers in high-dimensional numeric data. SEMSE defines a recurrent fragmentary sparse modeling process to build the sequential ensembles, in which feature selection and outlier scoring are iteratively and mutually refined. It results in more reliable outlier scores on data with many noisy features, compared to existing subspace/feature selection-based solutions.
2. SEMSE is further instantiated to CINFO, a method that introduces a *Cantelli*'s inequality-based fragmentary lasso to learn the sequential ensembles. The *Cantelli*'s inequality provides a false positive upper bound for outlier thresholding with no specific probability distribution assumption on the outlier scores, which well guarantees the refinement of feature selection and outlier scoring in the later stage of sequential ensembles.

A series of empirical results shows that (i) the CINFO-enabled LeSiNN and iForest perform significantly better than three state-of-the-art competitors and the bare versions of LeSiNN and iForest on 11 real-world high-dimensional data sets; (ii) CINFO has much better resilience to noisy features than its competitors; and (iii) CINFO has linear time complexity w.r.t. data size and data dimensionality.

Related Work

High-dimensional Outlier Detection Methods

Subspace-based methods (Aggarwal and Yu 2005; Keller, Muller, and Bohm 2012; Dang et al. 2014) are popular solutions for high-dimensional outlier detection. They search for a set of feature subspaces and use them to avoid the curse of dimensionality, but the subspace search is often costly as it requires extensive search in identifying the feature subspaces in high-dimensional data. Random subspace generation is a widely used solution to address this efficiency issue (Lazarevic and Kumar 2005; Nguyen, Ang, and Gopalkrishnan 2010), but it may include many noisy features into subspaces while omit relevant features in high-dimensional data with dominant noisy features.

Alternatively, feature selection-based methods aim to identify a single optimal feature subset that reveals the exceptional behaviors of all outliers. Although feature selection has shown effective in enabling clustering and classification for decades (Li et al. 2016), there exists limited work on outlier detection because it is challenging to (i) define feature relevance to outlier detection given its unsupervised nature and (ii) find a single feature subset enabling detection of all outliers. The method we denote it as RegFS in (Noto, Brodley, and Slonim 2012; Paulheim and Meusel 2015) defines the relevance of features by their correlation to the other features. The assumption is that outliers correspond to the violation of the dependency among normal objects and independent features are not useful in capturing such dependency/violation (Aggarwal 2017). This assumption may be invalid since some features can be strongly relevant to outlier detection but not correlated to other features.

One shared problem for the subspace/feature selection-based methods is that they search feature subset(s) independently from the subsequent outlier detection methods, and they may consequently result in feature subset(s) that are suboptimal to the outlier detectors. Other related work (Angiulli and Pizzuti 2005; Ghoting, Parthasarathy, and Otey 2006; Kriegel and Zimek 2008; Li, Shao, and Fu 2015) dedicates to more outlier-sensitive outlierness measures or data representation. Since they work on the full space, their performance may be still largely biased by noisy features.

There have been some subspace/feature selection methods (Angiulli, Fassetti, and Palopoli 2009; Akoglu et al. 2012; Pang et al. 2016; 2017a; Pang, Cao, and Chen 2016; Pang et al. 2017b) for *categorical* data. We have tried popular unsupervised discretization methods like equal-width and equal-frequency to adopt these methods to numeric data, but they performed poorly. Discretization methods need to be specially designed for outlier detection and need further careful development. We therefore focus on comparing CINFO with numeric data-based methods in our experiments.

Outlier Ensemble Learning

Unlike the well-established ensemble methods for clustering and classification, outlier ensemble learning attracts wide attention only in recent years (Aggarwal 2013; Zimek, Campello, and Sander 2013). Most existing outlier ensembles (Lazarevic and Kumar 2005; Liu, Ting, and Zhou 2012; Sugiyama and Borgwardt 2013; Pang, Ting, and Albrecht 2015) are in the parallel ensemble learning paradigm that constructs a set of independent base models. In contrast, sequential ensembles construct dependent base models by using the results of the current base model to refine the next one. It is very difficult to construct sequential ensembles for outlier detection as class labels are often assumed to be unavailable. As far as we know, the method called CARE in

(Rayana, Zhong, and Akoglu 2016) is the only work of this kind, which intends to reduce the masking and swamping effects (Hadi and Simonoff 1993) by iteratively removing potential outliers to refine the base models. This does not help in addressing the aforementioned issues in the high-dimensional space. SEMSE is fundamentally different from CARE, as we explore to iteratively eliminate noisy features to mutually refine feature selection and outlier scoring.

SEMSE for Mutual Refinement of Feature Selection and Outlier Scoring

The SEMSE framework builds a set of sequential ensembles to mutually refine outlier scoring and feature selection. As shown in Figure 1, SEMSE works as follows. At the t-th iteration, given a set of N data objects $\mathcal{X}$=$\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$ described by a set of D features (i.e., $\mathbf{x}_i$=$\{x_{i1}, x_{i2}, \cdots, x_{iD}\}$) and their outlier score vector $\mathbf{y}^{t-1} \in \mathbb{R}^N$ obtained in the previous iteration, SEMSE first defines an outlier thresholding function η^t to yield a set of L^t outliers $\mathcal{R}^t \in \mathbb{R}^{L^t \times (D+1)}$. $\mathcal{R}^t$ contains $D+1$ dimensions as it concatenates the original D dimensions and $\mathbf{y}^{t-1}$. SEMSE further treats $\mathbf{y}^{t-1}$ as the target feature and the other D features as predictors, and applies a sparse regression model ψ^t on $\mathcal{R}^t$ to produce a new data set $\mathcal{S}^t$ with a set of M^t optimal features w.r.t. $\mathbf{y}^{t-1}$, i.e., $\mathcal{S}^t \in \mathbb{R}^{N \times M^t}$, together with an empirical error mse^t. SEMSE then uses an outlier scoring function ϕ^t on $\mathcal{S}^t$ to re-compute an outlier score vector $\mathbf{y}^t$. SEMSE repeats these recurrent steps to yield a set of outlier score vectors until $mse^{t+1} > mse^t$. These recurrent steps compose a sequential ensemble model. SEMSE finally performs bagging to aggregate a set of sequential ensemble models to obtain the final outlier scores.

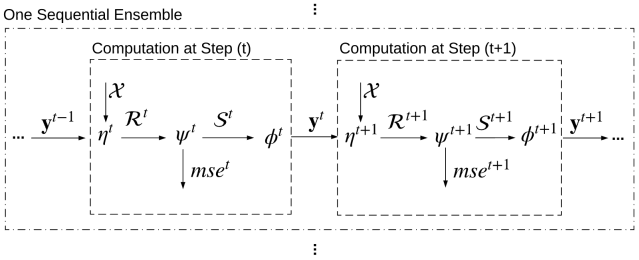

Figure 1: Our SEMSE Framework. $\mathbf{y}$ contains outlier scores of all data objects. η, ψ and ϕ are functions for outlier thresholding, fragmentary sparse modeling, and outlier scoring, respectively.

Essentially, SEMSE uses the pseudo target feature $\mathbf{y}^{t-1}$ to generate $\mathcal{S}^t$ with a feature subset that is mostly correlated to the outlier scores produced by the scoring function ϕ^{t-1}. Since ϕ^t works on $\mathcal{S}^t$ with the selected features that are tailored for it, SEMSE likely obtains an enhanced score vector $\mathbf{y}_t$, and it can in turn yield a better feature subset in $\mathcal{S}^{t+1}$ in next iteration. This cycle enables SEMSE to obtain more reliable outlier scores compared to that computed on the original feature space. The sequential ensembles help SEMSE reduce the learning bias while the final bagging stage helps reduce the learning variance (Aggarwal 2017).

SEMSE has good generalizability since it can be instantiated to a specific sequential ensemble method by specifying its three components η, ψ and ϕ. We introduce an instance of SEMSE in next section and then verify its performance by theoretical and empirical analyses.

A SEMSE Instance: CINFO

CINFO instantiates SEMSE by a *Cantelli*'s inequality-based outlier thresholding function η, a lasso-based fragmentary sparse regression function ψ, and a subsampling-based outlier scoring function ϕ. After building a sequential ensemble with these three functions, bagging is performed to obtain a set of such sequential ensembles and combine their outlier scores to well identify high-dimensional outliers.

Building a Sequential Ensemble

Outlier Thresholding η with *Cantelli*'s Inequality. The outlier thresholding function η is to identify a set of most likely outliers. We define a *Cantelli*'s inequality-based η as follows, which provides an upper bound for false positives.

Definition 1 (Outlier Thresholding)**.** *Given an outlier score vector $\mathbf{y} \in \mathbb{R}^N$, in which large scores indicate high outlierness, and let μ and δ^2 be its expected value and variance, then the outlier candidate set $\mathcal{R}$ is defined as follows:*

$$\mathcal{R} = \{(\mathbf{x}_i, y_i) | \eta(y_i, a) \geq 0\},\ \forall \mathbf{x}_i \in \mathcal{X}, y_i \in \mathbf{y}, \quad (1)$$

where $\eta(y_i, a) = y_i - \mu - a\delta$ and a is user-defined.

This outlier thresholding is equivalent to selecting the outlier candidates with a false positive upper bound of $\frac{1}{1+a^2}$ based on *Cantelli*'s inequality (see our theoretical support in next section).

Fragmentary Sparse Modeling ψ with Lasso. CINFO performs fragmentary sparse modeling on the data subset $\mathcal{R} \in \mathbb{R}^{L \times (D+1)}$. $\mathcal{R}$ is the newly created data set with reduced objects at the outlier thresholding stage, in which L represents the number of data objects identified as outliers by the η function and thus $L \ll N$. Specifically, CINFO conducts an univariate sparse regression learning as follows:

$$\psi(\mathcal{R}, \lambda) = \arg\min_{\boldsymbol{\omega}} \left(\frac{1}{2L} \sum_{i=1}^{L} (y_i - \mathbf{x}_i^\intercal \boldsymbol{\omega})^2 + \lambda ||\boldsymbol{\omega}||_1 \right), \quad (2)$$

where $\boldsymbol{\omega}$ is the coefficient vector and λ is a regularization parameter. When λ is large, solving Eqn. (2) obtains a shrinking solution to the least squares model, resulting in a number of zero-coefficient features that are not correlated to the outlier score $\mathbf{y}$. We then obtain another newly created data set $\mathcal{S} \in \mathbb{R}^{N \times M}$ with reduced features (i.e., $M < D$):

$$\mathcal{S} = \{\mathbf{x}_{\cdot i} | \omega_i \neq 0,\ 1 \leq i \leq D\}, \quad (3)$$

The parameter λ is critical to the performance of lasso. Inappropriate λ will lead to overfitting or underfitting. To address this issue, we use 10-fold cross validation on $\mathcal{R}$ to choose the best λ that minimizes the mean square error mse.

CINFO performs fragmentary sparse modeling for two major reasons. (i) Restricting the sparse modeling only on

the outlier candidate set $\mathcal{R}$ enables CINFO to select features that are mostly relevant to outlier identification. Since outliers are normally a minority of the data, sparse modeling on the full data set can be dominated by normal objects and fail to obtain outlier-sensitive features. (ii) It helps tune the parameter λ much more efficiently. Since $L \ll N$, performing the cross validation on $\mathcal{R}$ is substantially much faster than on the full data set $\mathcal{X}$.

Subsampling-based Outlier Scoring ϕ. To demonstrate the flexibility of SEMSE, we use the two very different subsampling-based outlier scoring methods LeSiNN and iForest to specify ϕ, respectively.

LeSiNN is a subsampling-based ensemble of the nearest-neighbor outlier detector using the *full dimensionality* of the input data $\mathcal{S}$. Given a data object $\mathbf{x}_i \in \mathcal{S}$, its outlier score is computed as the average of the nearest neighbor distances in l subsamples:

$$y_i = \phi(\mathbf{x}_i) = \frac{1}{l}\sum_{j=1}^{l} nn\_dist(\mathbf{x}_i|\mathcal{M}_j), \tag{4}$$

where $\mathcal{M}_j \subset \mathcal{S}$ is a random data subsample and nn_dist returns the nearest neighbor distance of $\mathbf{x}_i$ in $\mathcal{M}_j$.

iForest posits that outliers are susceptible to isolation and builds isolation trees on random *subspaces* in $\mathcal{S}$ to identify outliers. Each tree is grown by using a random subsample until every data object is isolated, where the feature and cut-point at each tree node are randomly selected. The inverse of the path length traversed from the root to a leaf node by $\mathbf{x}_i$ is used as its outlier score:

$$y_i = \phi(\mathbf{x}_i) = (2^{-\frac{E(h(\mathbf{x}_i))}{c(|\mathcal{M}|)}})^{-1}, \tag{5}$$

where $h(\mathbf{x}_i)$ denotes the path length of $\mathbf{x}_i$ in a subsample $\mathcal{M}$, $E(h(\mathbf{x})) = \frac{1}{l}\sum_{j=1}^{l} h_j(\mathbf{x}_i|\mathcal{M}_j)$ is the average path length of $\mathbf{x}_i$ from a set of l isolation trees, and $c(|\mathcal{M}|)$ is the expected path length given the subsample size $|\mathcal{M}|$.

The use of subsampling results in the linear time complexities in LeSiNN and iForest, which is critical for the efficiency of CINFO. LeSiNN and iForest are the state-the-of-art detectors and they are expected to yield fairly good outlier scores to ensure that there are at least some outliers in $\mathcal{R}$ output by η.

Combination of Outlier Scores. CINFO performs the above three recurrent components η, ψ and ϕ until the mean squared error mse produced by ψ does not further decrease. Assume the sequential ensemble learning terminates after T iterations, i.e., $t = \{1, 2, \cdots, T\}$, we will obtain a set of T outlier score vectors and their associated mse. We employ the commonly-used weighted summation (Freund and Schapire 1995) to combine the outlier score vectors with mse as weights, and define an outlier score for each data object in the sequential ensemble as follows:

$$seq\_score(\mathbf{x}_i) = \frac{1}{T}\sum_{t=1}^{T} w^t \tau(y_i^t), \tag{6}$$

where w^t is a normalized weight by $w^t = \frac{\mathbb{Z}_{mse} - mse^t}{\sum_{t=1}^{T}[\mathbb{Z} - mse^t]}$ with $\mathbb{Z} = \sum_{t=1}^{T} mse^t$, and $\tau(y_i^t) = \frac{y_i^t}{||\mathbf{y}^t||_1}$ is a vector normalization function that normalizes the vector $\mathbf{y}$ into an unit norm to address the heterogeneity of the outlier scores from heterogeneous feature subsets.

Note that the initial outlier score vector $\mathbf{y}_0$ is not integrated into the above weighted combination. This is because $\mathbf{y}_0$ is obtained from the original full feature space with noisy features and is thus not as reliable as the later score vectors.

Aggregating a Set of Sequential Ensembles

Using single sequential ensemble may produce high detection errors, when the initial outlier score vector $\mathbf{y}_0$ happens to mislead the subsequent outlier scoring in the sequential ensemble. We therefore further aggregate a set of sequential ensembles to address this issue by bagging. Bagging is a representative approach for building a set of base models independently, which can largely reduce the generalization error (Breiman 1996). Specifically, the final outlier score of a given object is the average over its outlier scores obtained from a set of independent sequential ensembles:

$$score(\mathbf{x}_i) = \frac{1}{m}\sum_{j=1}^{m} seq\_score_j(\mathbf{x}_i), \tag{7}$$

where m is the number of sequential ensembles we built.

The Algorithm and Its Time Complexity

Algorithm 1 presents the procedure of CINFO. Given a data set $\mathcal{X}$, Step 2 obtains the initial outlier scores. Steps 4-10 use the three recurrent functions η, ψ and ϕ to build a sequential ensemble for iterative refinement of the selected feature subset in $\mathcal{S}$ and outlier scores $\mathbf{y}$. The lasso problem in Step 7 is implemented by alternating direction method of multipliers (ADMM), and $\boldsymbol{\omega}$, mse and λ are obtained by 10-fold cross validation on $\mathcal{R}$. The outer loop in Steps 1-12 builds a set of independent sequential ensembles by bagging, followed by the average combination of the outlier scores from these sequential ensembles in Step 13. CINFO then returns an outlier ranking based on the outlier scores.

The sequential ensemble learning in Steps 4-10 often terminates in a few iterations (e.g., within 10). The bagging in the outer loop typically converges quickly, say after about 10-30 iterations. Therefore, the time complexity of CINFO is determined by the complexity of the three functions η, ψ and ϕ. Obviously, η has a linear time complexity w.r.t. data size and the number of features. The LeSiNN/iForest-based ϕ function has the similar linear time complexity (Liu, Ting, and Zhou 2012; Pang, Ting, and Albrecht 2015). Moreover, a linear convergence rate is expected for ADMM-based lasso implementation according to (Hong and Luo 2017). We therefore expect that the overall time complexity of CINFO is linear w.r.t. data size and dimensoinality size.

Theoretical Foundation

The following three subsections present some theoretical support for the specifications of the three functions η, ψ and ϕ in CINFO, respectively.

Algorithm 1 *CINFO*

Input: $\mathcal{X}$ - data objects, a - outlier thresholding parameter, m - bagging size
Output: R - an outlier ranking of objects

for $j = 1$ to m **do**
  $\mathbf{y}^0 \leftarrow \phi(\mathcal{X})$
  $mse^0 = 1, t = 0$
  repeat
    $t \leftarrow t + 1$
    $\mathcal{R}^t \leftarrow \eta(\mathbf{y}^{t-1}, a)$
    $\boldsymbol{\omega}^t,\ mse^t \leftarrow \psi(\mathcal{R}^t, \lambda^t)$
    $\mathcal{S}^t \leftarrow \{\mathbf{x}_{.i} | \omega_i^t \neq 0,\ 1 \leq i \leq D\}$
    $\mathbf{y}^t \leftarrow \phi(\mathcal{S}^t)$
  until $mse^t > mse^{t-1}$
  $seq\_score_j(\mathcal{X}) = \frac{1}{T}\sum_{t=1}^{T}(\mathbf{w}^t)^\intercal \tau(\mathbf{y}^t)$
end for
$score(\mathcal{X}) = \frac{1}{m}\sum_{j=1}^{m} seq\_score_j(\mathcal{X})$
$R \leftarrow$ Sort $\mathcal{X}$ w.r.t. $score$
return R

Upper Bound for Outlier Thresholding

Corollary 1 (False Positive Bound)**.** *Assume the scores in* $\mathbf{y}$ *have the expected value* μ *and variance* δ^2*. Let* $y_i \in \mathbf{y}$*, the outlier thresholding function* $\eta(y_i, a) = y_i - \mu - a\delta$ *then has a false positive upper bound of* $\frac{1}{1+a^2}$*.*

Proof. We have $\text{Prob}(y_i \geq \mu + \alpha) \leq \frac{\delta^2}{\delta^2+\alpha^2}$ per *Cantelli*'s inequality. By replacing $\alpha = a\delta$, we obtain

$$\text{Prob}(y_i \geq \mu + a\delta) \leq \frac{1}{1+a^2}. \tag{8}$$

It states that the values in $\mathbf{y}$ have a maximum probability of $\frac{1}{1+a^2}$ to be greater than $\mu + a\delta$. Since large y_i indicates high outlierness, this inequality implies that the probability that we could wrongly identify normal objects as outliers is up to $\frac{1}{1+a^2}$ when we define the threshold as $\mu + a\delta$. □

Cantelli's inequality is an one-sided *Chebyshev*'s inequality. Similar to *Chebyshev*'s inequality, it makes no assumption on specific probability distributions. It holds for a wide class of probability distributions that have statistical mean and variance. This property enables η to be data-dependent and perform well for $\mathbf{y}$ following different distributions.

Optimal Feature Subsets w.r.t. Outlier Scoring ϕ

Since the sparse modeling in Eqn. (2) is a convex problem (Hastie, Tibshirani, and Wainwright 2015), the feature subset in $\mathcal{S}$ is expected to be globally optimal w.r.t. the target $\mathbf{y}$ on the outlier candidate set $\mathcal{R}$. In other words, the selected features are customized to the outlier scoring function ϕ that produces the score vector $\mathbf{y}$. This enables ϕ to work on a more reliable feature set when re-computing the outlier scores by using $\mathcal{S}$, resulting in refined outlier scores compared to that in the previous iteration.

In the best case, the outlier scoring or feature selection is iteratively refined. In another extreme, when the outlier scores are poor, e.g., no true outliers are in the outlier candidates, it can mislead the feature selection and does not help improve the successive outlier scoring. The next section analyzes the use of subsampling to obtain quality outlier scores.

Obtaining Good Outlier Scores by Subsampling

In addition to substantial speedup, using subsampling can well guarantee the outlier scoring quality, which is supported by theoretical results from the perspectives of, e.g., density estimation (Zimek et al. 2013), data distribution (Sugiyama and Borgwardt 2013) and variance reduction (Aggarwal 2017). We provide the following analysis to further complement these existing theoretical results.

Following (Zimek et al. 2013), for two data objects $\mathbf{x}_1$ and $\mathbf{x}_2$, their expected kNN distance k_dist in $\mathcal{X}$ can be respectively approximated by $E(k\_dist(\mathbf{x}_1|\mathcal{X})) = r\left(\frac{k}{N_1}\right)^{\frac{1}{D}}$ and $E(k\_dist(\mathbf{x}_2|\mathcal{X})) = r\left(\frac{k}{N_2}\right)^{\frac{1}{D}}$, where N_1 and N_2 are the numbers of objects uniformly distributed in the r-radiu sphere of $\mathbf{x}_1$ and $\mathbf{x}_2$, respectively; and their expected k_dist in a random subsample $\mathcal{R}$ of size L can then be given by $E(k\_dist(\mathbf{x}_1|\mathcal{R})) = r\left(\frac{k}{N_1\frac{L}{N}}\right)^{\frac{1}{D}}$ and $E(k\_dist(\mathbf{x}_2|\mathcal{R})) = r\left(\frac{k}{N_2\frac{L}{N}}\right)^{\frac{1}{D}}$. After some transformation, we can obtain:

$$\frac{E(k\_dist(\mathbf{x}_1|\mathcal{R})) - E(k\_dist(\mathbf{x}_2|\mathcal{R}))}{E(k\_dist(\mathbf{x}_1|\mathcal{X})) - E(k\_dist(\mathbf{x}_2|\mathcal{X}))} = (\frac{N}{L})^{\frac{1}{D}}. \tag{9}$$

Eqn. (9) implies that the contrast between the kNN-based densities in the subsamples and that in the full data set are enlarged and are inversely proportional to the subsampling size. This indicates that subsampling helps enhance the contrast between kNN/density-based outlier scores. Moreover, it also guarantees a ranking-stable result, i.e., $E(k\_dist(\mathbf{x}_1|\mathcal{R})) > E(k\_dist(\mathbf{x}_2|\mathcal{R}))$ if $E(k\_dist(\mathbf{x}_1|\mathcal{X})) > E(k\_dist(\mathbf{x}_2|\mathcal{X}))$. These two properties enable the subsampling-based scoring to yield better outlier scores (Kriegel et al. 2011).

Numerous existing outlier scoring methods including LeSiNN assume that outliers are data objects in low-density regions. Therefore, the above results are widely applicable, and subsampling is recommended for the specification of ϕ in CINFO when using this type of methods.

Experiments and Evaluation

Data Sets

As shown in Table 1, 11 real-world data sets are used[1], which cover diverse domains, e.g., intrusion detection, molecular bioactivity detection, Internet advertising and image object recognition. Some data sets like *AD*, *AID362*, *Probe*, *U2R* and *Thrombin* contain semantically real outliers. For the other data sets, we follow the literature (Lazarevic and Kumar 2005; Pang et al. 2016; Rayana, Zhong, and Akoglu 2016) to treat rare classes as outliers and the largest class as the normal class. Categorical features are converted into numeric ones by 1-of-ℓ encoding (Campos et al. 2016).

[1]They are available at http://archive.ics.uci.edu/ml, http://www.kdd.org/kdd-cup, http://vision.cs.uiuc.edu/attributes/ and https://people.cs.umass.edu/∼marlin/data.shtml.

Performance Evaluation Methods

All outlier detectors finally yield an object ranking w.r.t. its outlier score, i.e., the top-ranked objects are the most likely outliers. We evaluate the quality of the ranking by the area under ROC curve (AUC) (Hand and Till 2001). AUC inherently considers the class-imbalance nature of outlier detection tasks, making it comparable across different data sets (Campos et al. 2016). Higher AUC indicates better detection performance. The *Wilcoxon* signed rank test is used to examine the significance of the AUC performance of CINFO against its competitors. Since LeSiNN and iForest are randomized algorithms, all their AUCs are the averaged results over 10 independent runs.

Experiment Environment

CINFO and its competitors are implemented in MATLAB. All the experiments are executed at a node in a 3.4GHz Phoenix cluster with 32GB memory. In all our experiments, CINFO uses $a = 1.732$ (i.e., the upper bound for false positives in η is 25%) and $m = 30$ [2]; and the number of subsamples l and subsampling size $|\mathcal{M}|$ for LeSiNN and iForest are set as the recommended settings of their authors.

Effectiveness in Real-world Data

Experimental Settings. We compare the CINFO-enabled LeSiNN and iForest with their bare versions to evaluate whether CINFO can eliminate irrelevant features and retain (or improve) the performance of these two detectors.

Findings - CINFO Significantly Improving Different Types of Outlier Detectors. Table 1 demonstrates the feature reduction and AUC performance of CINFO-based LeSiNN and iForest, compared to LeSiNN and iForest performing in the original feature space. CINFO-enabled LeSiNN and iForest work with about 10% (e.g., on *AID362* and *BM*) to over 95% (e.g., on *Isolet*, *SECOM* and *Thrombin*) less features, while their performance is substantially better than, or roughly the same as, their bare versions. On average, CINFO enables LeSiNN and iForest to gain about 4% and 7% improvement, respectively. Our significance test shows that CINFO enables LeSiNN and iForest to achieve significantly better AUC performance at the 95% and 99% confidence levels, respectively.

CINFO uses the sequential ensemble learning to mutually improve its feature selection and outlier scoring, which enables CINFO to safely remove noisy features in these high-dimensional data sets. As a result, CINFO-enabled LeSiNN and iForest work on much cleaner data sets and thus can achieve significant performance improvement.

Comparison to State-of-the-art Competitors

Experimental Settings. CINFO is compared with three state-of-the-art competitors: feature bagging (FB for short) (Lazarevic and Kumar 2005), RegFS (Paulheim and Meusel 2015), and CARE (Rayana, Zhong, and Akoglu 2016) from three different but relevant research lines.

[2]CINFO performs very stably when $m \geq 30$. $m = 30$ is thus used. These test results are omitted due to page limitations.

Table 1: Feature Reduction and AUC Performance of CINFO-enabled LeSiNN and iForest (denoted by LeSiNN* and iForest*). D is the original feature number. D' and D'' are the average numbers of features retained by LeSiNN* and iForest*, respectively. The average iteration for sequential ensembles per data is 2 to 5.

Data Info.		**Feature Reduction**			**AUC Performance**			
Data	N	D	D'	D''	LeSiNN	LeSiNN*	iForest	iForest*
AD	3279	1555	197	245	0.7107	**0.8666**	0.6830	**0.7907**
AID362	4279	117	106	94	0.6704	**0.6710**	0.6461	**0.6658**
aPascal	12695	64	34	46	0.7308	**0.8554**	0.6755	**0.7963**
BM	41188	62	54	52	0.6854	**0.7100**	0.7316	**0.7678**
Caltech16	829	253	59	50	0.9861	**0.9869**	0.9636	**0.9684**
Census	299285	500	399	422	0.6344	**0.6620**	0.6276	**0.6616**
Isolet	730	617	27	28	1.0000	1.0000	0.9996	**1.0000**
Probe	64759	34	27	25	0.9975	**0.9978**	0.9899	**0.9908**
SECOM	1567	590	27	18	0.5316	**0.5867**	0.5448	**0.6506**
U2R	60821	36	28	30	0.9879	**0.9890**	0.9908	**0.9922**
Thrombin	1909	139351	114	58	**0.8997**	0.8916	0.8843	**0.9044**
			Average		0.8031	**0.8379**	0.7943	**0.8353**

- *Subspace-based method - FB*. FB is a framework for enabling outlier detectors to handle high-dimensional data by using feature bagging, i.e., working on a set of random feature subsets of size between $\lfloor \frac{D}{2} \rfloor$ and $(D-1)$. It can also be seen as a random feature selection ensemble.
- *Feature selection-based competitor - RegFS*. RegFS only returns a feature relevance ranking. For a thorough comparison, RegFS selects the top-ranked $\lceil rD \rceil$ features with a wide range of r, i.e., $r = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$. Due to page limitations, we report the results of $r = 0.7$, with which the used detectors obtain the best performance.
- *Sequential outlier ensemble - CARE*. CARE attempts to iteratively refine detection models by removing outlier candidates. It uses feature bagging to introduce diversity and handle high-dimensional data.

Findings - CINFO Significantly Outperforming Three State-of-the-art Competitors. The AUC performance of CINFO, RegFS, FB, and CARE is reported in Table 2. The CINFO-enabled LeSiNN and iForest obtain the best performance on eight data sets, with three very close to the best (having the difference in AUC less than 0.01), and they obtain about 4%-7% improvement over the respective competitors. The improvement is significant at the 95% (w.r.t. RegFS and FS) or 90% (w.r.t. CARE) confidence level.

Unlike FB and RegFS that ignore the outlier scoring methods when they perform feature selection, CINFO couples these two dependent tasks to iteratively refine their performance by sequential ensembles. This enables CINFO to substantially reduce its detection errors and obtain more than 4%-22% AUC improvement over its competitors in tough data sets like *AD*, *aPascal*, *Census*, and *SECOM*, which likely contain a large proportion of noisy features.

CINFO and CARE are two very different sequential ensemble methods. CARE builds sequential ensembles horizontally, which iteratively removes likely outliers for identifying some outliers that are otherwise masked by the removed outliers. In contrast, CINFO works in a vertical manner, which iteratively remove noisy features. Although feature bagging FB is incorporated into CARE, the FB

Table 2: AUC Performance of CINFO, RegFS, FB, and CARE Empowered LeSiNN and iForest. 'NA' indicates the execution cannot be completed in two weeks.

	LeSiNN				iForest			
Data	CINFO	RegFS	FB	CARE	CINFO	RegFS	FB	CARE
AD	**0.8666**	0.7058	0.7111	0.6934	**0.7907**	0.6832	0.6892	0.6989
AID362	0.6710	0.6371	0.6704	**0.6767**	0.6658	0.6421	0.6659	**0.6752**
aPascal	**0.8554**	0.7464	0.7319	0.7349	**0.7963**	0.7085	0.6642	0.6829
BM	**0.7100**	0.6943	0.6879	0.6818	**0.7678**	0.7328	0.7440	0.7444
Caltech16	0.9869	**0.9874**	0.9869	**0.9874**	0.9684	**0.9728**	0.9670	0.9691
Census	**0.6620**	0.6112	0.6340	0.6198	**0.6616**	0.5638	0.6290	0.6416
Isolet	1.0000	1.0000	1.0000	1.0000	**1.0000**	0.9995	**1.0000**	**1.0000**
Probe	**0.9978**	0.9943	0.9974	0.9970	0.9908	0.9900	0.9908	**0.9941**
SECOM	**0.5867**	0.5343	0.5294	0.5282	**0.6506**	0.5636	0.5533	0.5589
U2R	**0.9890**	0.9645	0.9877	0.9853	**0.9922**	0.9717	0.9904	0.9903
Thrombin	0.8916	NA	0.8995	**0.9023**	**0.9044**	NA	0.9024	0.9034
Average	0.8379	0.7875	0.8033	0.8006	0.8353	0.7828	0.7997	0.8053
p-value	-	0.0078	0.0273	0.0840	-	0.0098	0.0078	0.0840

method itself has limited capability in handling noisy features. CINFO therefore obtains similarly large AUC improvement (i.e., 6%-24%) over CARE on the aforementioned noisy data sets.

Resilience to Noisy Features

Experiment Settings. Following (Zimek, Schubert, and Kriegel 2012), we create a collection of 100-dimensional synthetic data sets with different percentages of relevant features (or noisy features). In this data, normal objects are from a Gaussian distribution and outliers lie at two standard deviations of the distribution in relevant features, and the other features are from uniform distribution and used as noisy features. For each noise level, we generate 10 data sets with the same number of noisy features and average AUC over them to have more reliable results.

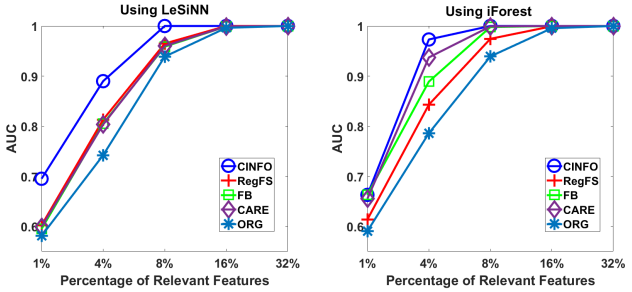

Figure 2: AUC Performance on Data with Different Levels of Noisy Features. 'ORG' denotes the bare LeSiNN/iForest. All methods obtain AUC of one with more than 32% relevant features.

Findings - CINFO Greatly Enhancing the Resilience of the Outlier Detectors w.r.t. Noisy Features, Especially for Very Noise-Sensitive Detectors. The AUC performance on the synthetic data sets are shown in Figure 2. CINFO-enabled LeSiNN and iForest perform consistently better than their four other versions in a wide range of noise levels. The advantage of CINFO is much more obvious in enabling LeSiNN than iForest. This may be due to the fact that LeSiNN works on the full space of the input data while iForest operates on feature subspaces, and as a result, LeSiNN is much more sensitive to the noisy features retained by the feature subset selection methods and is more difficult to be enhanced, compared to iForest. The substantially better performance of the CINFO-enabled LeSiNN over its competitors highlights its superiority in eliminating noisy features and upgrading very noise-sensitive detectors.

Scalability Test

Experiment Settings. We generate data sets by varying the data dimension w.r.t. to a fixed data size (i.e., 1000), as well as varying the data size while fixing the data dimension (i.e., 50), respectively.

Findings - CINFO Obtaining Linear Time Complexity w.r.t. Data Size and Dimensionality. The runtime of the five versions of LeSiNN is shown in Figure 3. In the left panel, all the methods have linear time complexity. CINFO is comparably fast to RegFS and CARE. These three methods are slower than FB and the bare LeSiNN, since they incorporate more sophisticated components to enhance the accuracy of LeSiNN. In the right panel, the CINFO/FB/CARE-enabled and the bare LeSiNN have linear time complexity, and they run considerably faster than RegFS that has a quadratic complexity.

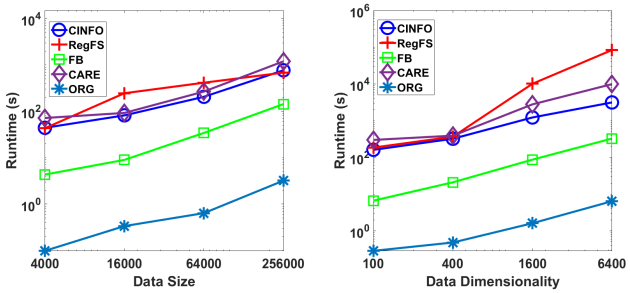

Figure 3: Runtime of CINFO and Its Competitors Using LeSiNN. 'ORG' denotes the bare LeSiNN. Logarithmic scales are used. Similar trends can be expected for using iForest as the outlier detector, since LeSiNN and iForest have similar time complexities.

Conclusions

This paper introduces a sequential ensemble-based high-dimensional outlier detection framework SEMSE and its instance CINFO. They perform an iterative mutual refinement of feature selection and outlier scoring, and can efficiently obtain reliable outlier scores in high-dimensional numeric data with many noisy features. Although CINFO works on considerably smaller feature subsets, it obtains significantly better AUC performance in 11 real-world high-dimensional data sets, substantially better resilience to noisy features, compared to its four competitors. CINFO also has linear time complexity w.r.t. data size and dimensionality. We are further enhancing CINFO by replacing the lasso-based sparse modeling with other sparse constraints.

Acknowledgments This work is partially supported by the ARC Discovery Grant DP140100545.

References

Aggarwal, C., and Yu, S. 2005. An effective and efficient algorithm for high-dimensional outlier detection. *The VLDB Journal* 14(2):211–221.

Aggarwal, C. C. 2013. Outlier ensembles: Position paper. *ACM SIGKDD Explorations Newsletter* 14(2):49–58.

Aggarwal, C. C. 2017. *Outlier analysis*. Springer.

Akoglu, L.; Tong, H.; Vreeken, J.; and Faloutsos, C. 2012. Fast and reliable anomaly detection in categorical data. In *CIKM*, 415–424. ACM.

Angiulli, F., and Pizzuti, C. 2005. Outlier mining in large high-dimensional data sets. *IEEE Transactions on Knowledge and Data Engineering* 17(2):203–215.

Angiulli, F.; Fassetti, F.; and Palopoli, L. 2009. Detecting outlying properties of exceptional objects. *ACM Transactions on Database Systems* 34(1):7.

Breiman, L. 1996. Bagging predictors. *Machine Learning* 24(2):123–140.

Campos, G. O.; Zimek, A.; Sander, J.; Campello, R. J.; Micenková, B.; Schubert, E.; Assent, I.; and Houle, M. E. 2016. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery* 30(4):891–927.

Cao, L.; Ou, Y.; and Yu, P. S. 2012. Coupled behavior analysis with applications. *IEEE Transactions on Knowledge and Data Engineering* 24(8):1378–1392.

Dang, X. H.; Assent, I.; Ng, R. T.; Zimek, A.; and Schubert, E. 2014. Discriminative features for identifying and interpreting outliers. In *ICDE*, 88–99. IEEE.

Dubhashi, D. P., and Panconesi, A. 2009. *Concentration of measure for the analysis of randomized algorithms*. Cambridge University Press.

Freund, Y., and Schapire, R. E. 1995. A desicion-theoretic generalization of on-line learning and an application to boosting. In *COLT*, 23–37. Springer.

Ghoting, A.; Parthasarathy, S.; and Otey, M. E. 2006. Fast mining of distance-based outliers in high-dimensional datasets. In *SDM*. SIAM.

Hadi, A. S., and Simonoff, J. S. 1993. Procedures for the identification of multiple outliers in linear models. *Journal of the American Statistical Association* 88(424):1264–1272.

Hand, D. J., and Till, R. J. 2001. A simple generalisation of the area under the ROC curve for multiple class classification problems. *Machine Learning* 45(2):171–186.

Hastie, T.; Tibshirani, R.; and Wainwright, M. 2015. *Statistical learning with sparsity: The lasso and generalizations*. CRC Press.

Hong, M., and Luo, Z.-Q. 2017. On the linear convergence of the alternating direction method of multipliers. *Mathematical Programming* 162(1-2):165–199.

Keller, F.; Muller, E.; and Bohm, K. 2012. HiCS: High contrast subspaces for density-based outlier ranking. In *ICDE*, 1037–1048. IEEE.

Kriegel, H.-P., and Zimek, A. 2008. Angle-based outlier detection in high-dimensional data. In *SIGKDD*, 444–452.

Kriegel, H.-P.; Kroger, P.; Schubert, E.; and Zimek, A. 2011. Interpreting and unifying outlier scores. In *SDM*, 13–24.

Lazarevic, A., and Kumar, V. 2005. Feature bagging for outlier detection. In *SIGKDD*, 157–166. ACM.

Li, J.; Cheng, K.; Wang, S.; Morstatter, F.; Trevino, R. P.; Tang, J.; and Liu, H. 2016. Feature selection: A data perspective. *CoRR* abs/1601.07996.

Li, S.; Shao, M.; and Fu, Y. 2015. Multi-view low-rank analysis for outlier detection. In *SDM*, 748–756. SIAM.

Liu, F. T.; Ting, K. M.; and Zhou, Z.-H. 2012. Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data* 6(1):3:1–3:39.

Nguyen, H. V.; Ang, H. H.; and Gopalkrishnan, V. 2010. Mining outliers with ensemble of heterogeneous detectors on random subspaces. In *DSFAA*, 368–383. Springer.

Noto, K.; Brodley, C.; and Slonim, D. 2012. FRaC: a feature-modeling approach for semi-supervised and unsupervised anomaly detection. *Data Mining and Knowledge Discovery* 25(1):109–133.

Pang, G.; Cao, L.; Chen, L.; and Liu, H. 2016. Unsupervised feature selection for outlier detection by modelling hierarchical value-feature couplings. In *ICDM*, 410–419. IEEE.

Pang, G.; Cao, L.; Chen, L.; and Liu, H. 2017a. Learning homophily couplings from non-iid data for joint feature selection and noise-resilient outlier detection. In *IJCAI*, 2585–2591.

Pang, G.; Xu, H.; Cao, L.; and Zhao, W. 2017b. Selective value coupling learning for detecting outliers in high-dimensional categorical data. In *CIKM*, 807–816. ACM.

Pang, G.; Cao, L.; and Chen, L. 2016. Outlier detection in complex categorical data by modelling the feature value couplings. In *IJCAI*, 1902–1908. AAAI Press.

Pang, G.; Ting, K. M.; and Albrecht, D. 2015. LeSiNN: Detecting anomalies by identifying least similar nearest neighbours. In *ICDM Workshop*, 623–630. IEEE.

Paulheim, H., and Meusel, R. 2015. A decomposition of the outlier detection problem into a set of supervised learning problems. *Machine Learning* 100(2-3):509–531.

Rayana, S.; Zhong, W.; and Akoglu, L. 2016. Sequential ensemble learning for outlier detection: A bias-variance perspective. In *ICDM*, 1167–1172. IEEE.

Sugiyama, M., and Borgwardt, K. 2013. Rapid distance-based outlier detection via sampling. In *NIPS*, 467–475.

Zimek, A.; Gaudet, M.; Campello, R. J.; and Sander, J. 2013. Subsampling for efficient and effective unsupervised outlier detection ensembles. In *SIGKDD*, 428–436. ACM.

Zimek, A.; Campello, R. J.; and Sander, J. 2013. Ensembles for unsupervised outlier detection: Challenges and research questions. *SIGKDD Explorations Newsletter* 15(1):11–22.

Zimek, A.; Schubert, E.; and Kriegel, H.-P. 2012. A survey on unsupervised outlier detection in high-dimensional numerical data. *Statistical Analysis and Data Mining* 5(5):363–387.