Learning Representations of Ultrahigh-dimensional Data for Random Distance-based Outlier Detection

Guansong Pang
University of Technology Sydney
Sydney, Australia
pangguansong@gmail.com

Longbing Cao
University of Technology Sydney
Sydney, Australia
longbing.cao@uts.edu.au

Ling Chen
University of Technology Sydney
Sydney, Australia
ling.chen@uts.edu.au

Huan Liu
Arizona State University
Tempe, United States
huan.liu@asu.edu

ABSTRACT

Learning expressive low-dimensional representations of ultrahigh-dimensional data, e.g., data with thousands/millions of features, has been a major way to enable learning methods to address the curse of dimensionality. However, existing *unsupervised* representation learning methods mainly focus on preserving the data regularity information and learning the representations independently of subsequent outlier detection methods, which can result in suboptimal and unstable performance of detecting irregularities (i.e., outliers).

This paper introduces a ranking model-based framework, called RAMODO, to address this issue. RAMODO unifies representation learning and outlier detection to learn low-dimensional representations that are *tailored* for a state-of-the-art outlier detection approach - the random distance-based approach. This customized learning yields more optimal and stable representations for the targeted outlier detectors. Additionally, RAMODO can leverage little labeled data as *prior knowledge* to learn more expressive and application-relevant representations. We instantiate RAMODO to an efficient method called REPEN to demonstrate the performance of RAMODO.

Extensive empirical results on eight real-world ultrahigh dimensional data sets show that REPEN (i) enables a random distance-based detector to obtain significantly better AUC performance and two orders of magnitude speedup; (ii) performs substantially better and more stably than four state-of-the-art representation learning methods; and (iii) leverages less than 1% labeled data to achieve up to 32% AUC improvement.

KEYWORDS

Outlier Detection, Representation Learning, Ultrahigh-dimensional Data, Dimension Reduction

KDD 2018, August 19–23, 2018, London, United Kingdom

ACM ISBN 978-1-4503-5552-0/18/08…$15.00
https://doi.org/10.1145/3219819.3220042

ACM Reference Format:
Guansong Pang, Longbing Cao, Ling Chen, and Huan Liu. 2018. Learning Representations of Ultrahigh-dimensional Data for Random Distance-based Outlier Detection. In *KDD 2018: 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, August 19–23, 2018, London, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3219819.3220042

1 INTRODUCTION

Outlier detection, which is referred to as the process of identifying data objects that deviate significantly from the majority of data objects, can offer important insights into broad applications in a range of domains such as medical diagnosis, fraud detection, and information security. Many of these applications often have ultrahigh dimensionality, e.g., thousands of features in detecting abnormal bioactivities [9], hundreds of thousands of features in corporate fraud detection [15], and millions of features in detecting malicious URLs [20]. Such ultrahigh-dimensional data presents significant challenges to existing outlier detection methods due to the curse of dimensionality [38].

One straightforward yet challenging solution is to map such high-dimensional data sets into low-dimensional representations that preserve the relevant information for subsequent learning tasks. Many unsupervised representation learning techniques have been introduced to address this issue, such as spectral-based methods, neural network-based methods, and manifold learning [5, 33]. However, most of these methods focus on preserving the data regularity information (e.g., data reconstruction/proximity information) for learning tasks like clustering and data compression. They may therefore fail to retain the important information for uncovering the irregularities (i.e., outliers). A few studies (e.g., [28]) have attempted to learn representations for identifying outliers in very recent years. However, they learn the representations independently of subsequent outlier detection methods. As a result, the optimal representations they produce may be suboptimal to a given specific outlier detection method, which leads to ineffective and unstable performance of the outlier detector.

Also, since these techniques focus on unsupervised learning, it is difficult for them to incorporate application-specific knowledge (e.g., a few labeled outliers) into the representation learning. When such prior knowledge is available as in many real-world outlier

detection applications, the valuable information cannot be used by these representation learning techniques.

In an attempt to address the above issues, this paper introduces a RAnking MOdel-based representation learning framework to learn representations of ultrahigh-dimensional data for Distance-based Outlier detection methods (RAMODO). RAMODO incorporates random distance-based outlier detection methods into the objective function of its representation learning to learn customized representations for such outlier scoring methods. *Random distance-based outlier detection* defines the outlierness of a data object based on its distance to the data objects in a random subsample. This approach has shown state-of-the-art accuracy and scalability [26, 30, 32], but it still suffers from the curse of dimensionality. RAMODO therefore chooses random distance-based methods as its representation learning target to introduce state-of-the-art distance-based methods for ultrahigh-dimensional outlier detection. Moreover, RAMODO can be easily extended to leverage the application-specific knowledge to learn more expressive and application-relevant representations.

RAMODO is implemented by a method called REPEN that learns customized REPresentations for a random nEarest Neighbor distance-based method, Sp [30]. REPEN defines its objective function using Sp-based outlier scores to guide the representation learning, resulting in low-dimensional representations that are tailored for Sp. While Sp is chosen for its state-of-the-art effectiveness and efficiency [30], RAMODO can also be customized for other random distance-based methods.

Accordingly, this paper makes three major contributions:

(1) We introduce the RAMODO framework to learn customized low-dimensional representations of ultrahigh-dimensional data for random distance-based outlier detectors. Unlike existing methods that preserve the regularity information and separate representation learning from subsequent outlier detectors, RAMODO unifies representation learning and outlier detection to learn a small set of features that are tailored for the random distance-based detectors. As a result, RAMODO can learn better representations for the detectors with more effective and stable performance.
(2) The RAMODO framework is instantiated into a method called REPEN to learn customized representations for one state-of-the-art random distance-based method, Sp. Sp has provable error bounds and is highly scalable, which enables REPEN to learn the representations with an upper error bound and scale up to large ultrahigh-dimensional data.
(3) We further introduce a method for REPEN to incorporate a small set of labeled outliers as application-specific knowledge, which helps the REPEN-enabled Sp identify application-relevant outliers, rather than data noises or uninteresting data objects due to the lack of such prior knowledge. This capability results in practical solutions in many real-world applications where a few labeled outliers are available.

Extensive empirical results on eight real-world data sets with thousands to millions of features and two sets of synthetic data show that REPEN (i) enables the original distance-based outlier detector to obtain significantly better AUC performance and two orders of magnitude speedup; (ii) performs substantially better and more stably than four state-of-the-art representation learning competitors; (iii) achieves up to 32% AUC improvement by leveraging less than 1% labeled outliers as prior knowledge; (iv) performs stably w.r.t. a wide range of representation dimensions; and (v) obtains linear time complexity w.r.t. both data size and dimensionality.

2 RELATED WORK

2.1 Distance-based Outlier Detection

Distance-based outlier detection is arguably one of the most widely-used detection approaches [6]. Some very popular distance-based methods include K-th nearest neighbor distance- and average K nearest neighbors distance-based methods [4]. This type of methods has time complexity quadratic w.r.t. data size. The time complexity may be reduced to be nearly linear by using indexing [4] or distributed computing techniques [8]. Recent studies [26, 30, 32] show that random distance-based methods or distance-based ensemble methods can achieve not only a similar time complexity reduction but also low false positive errors, resulting in scalable state-of-the-art distance-based detectors. However, these techniques still do not address the curse of dimensionality issue. Subspace-based approaches [1, 16] define outlierness using a set of relevant feature subspaces to avoid the curse of dimensionality. Outlying feature selection, which retains a feature subset that is relevant to outlier detection, emerges as an alternative solution to subspace-based methods [3, 23–25]. However, both approaches are mainly focused on data sets with tens/hundreds of features due to their prohibitive subspace search in the ultrahigh-dimensional space.

2.2 Unsupervised Representation Learning for Outlier Detection

Numerous unsupervised representation learning methods have been proposed to learn low-dimensional representations of high-dimensional data [5, 33]. They include: spectral-based methods, like principal component analysis (PCA) and its variants [7, 14]; neural network-based methods, like autoencoder and its variants [13, 21]; manifold learning, like locally linear embedding (LLE) and Hessian LLE [10]; random projection, like sparse random projection [19], to name a few. However, these methods can be biased by the presence of outliers, since they treat the inliers and outliers equally. Also, these methods are mainly designed to preserve the information of data regularities for unsupervised learning tasks like clustering and data compression. Their resultant representations may therefore ignore important information for uncovering the irregularities.

In recent years, robust PCA and robust autoencoder methods have been introduced to learn robust representations to reduce the bias caused by the outliers [7, 34]. However, these methods mainly address the bias issue. The very recent coherent pursuit in [28] and the combination of robust PCA and autoencoders [36] help address the above both issues. However, these two methods involve the costly eigen analysis or alternative optimization, making it unscalable to large ultrahigh-dimensional data. In addition, the method in [36] needs to tune its parameters in a semi-supervised way. Lastly, all the above methods may produce suboptimal representations for a specific outlier detection method, as they ignore the subsequent outlier detection when learning the data representations.

2.3 Using Labeled Outliers as Prior Knowledge

One problem with unsupervised outlier detection methods is that many of the outliers they identify are data noises or uninteresting data objects due to the lack of prior knowledge of irregularities [2]. Building the outlier detectors with application-specific knowledge (e.g., labeled outliers) may help identify application-relevant outliers. Related studies have attempted to incorporate some labeled outliers into graph outlier detection by belief propagation [22, 31]. Converting ultrahigh-dimensional data into K-nearest-neighbor graph may facilitate the adoption of this technique, but such conversion faces big challenges due to the difficulty of obtaining accurate distances in the high-dimensional space. Active learning and/or semi-supervised learning have been explored in [12, 27] to iteratively interact with domain experts to label some outliers and improve the detection performance with the labeled outliers, but these methods may require extensive feedback from the experts. Different from the above contexts, our model focuses on representation learning and it incorporates a few labeled outliers by a minor change to its inputs, and is thus very flexible in practice.

3 THE PROPOSED FRAMEWORK: RAMODO

3.1 Problem Statement

We aim to learn a low-dimensional space out of the ultrahigh-dimensional data, such that it becomes easier and/or more efficient for random distance-based detectors to identify outliers in the new space. Specifically, given a set of N data objects $\mathcal{X}=\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$ with $\mathbf{x}_i \in \mathbb{R}^D$ (i.e., $\mathbf{x}_i=\{x_{i1}, x_{i2}, \cdots, x_{iD}\}$), and a random distance-based outlier scoring function $\phi : \mathcal{X} \mapsto \mathbb{R}$ that uses distances in a random subsample to define outlierness, our goal is to learn a representation function $f : \mathcal{X} \mapsto \mathbb{R}^M$ ($M \ll D$) in a way that we have $\phi(f(\mathbf{x}_i)) > \phi(f(\mathbf{x}_j))$ if $\mathbf{x}_i$ is an outlier and $\mathbf{x}_j$ is an inlier.

3.2 Ranking Model-based Representation Learning Framework

We introduce the RAMODO framework by using a generalized pairwise ranking model to learn data representations that are tailored for random distance-based detectors. As shown in Figure 1, RAMODO consists of four essential components. RAMODO first performs an *outlier thresholding* to partition the data into inlier and outlier candidate sets. It then generates a meta triplet sample $T = \left(< \mathbf{x}_i, \cdots, \mathbf{x}_{i+n-1} >, \mathbf{x}^+, \mathbf{x}^-\right)$ by randomly picking up n data objects from the inlier candidate set $\mathcal{I}$ as the query set $(\mathbf{x}_i, \cdots, \mathbf{x}_{i+n-1})$, one object from $\mathcal{I}$ as a positive example ($\mathbf{x}^+$), and one object from the outlier candidate set $\mathcal{O}$ as a negative example ($\mathbf{x}^-$). This *meta triplet sampling* serves as an input layer to a network architecture. RAMODO further learns the *data representations* by a function f that can be composed of one or multiple fully-connected hidden layer(s). RAMODO finally performs the optimization guided by a *outlier score-based ranking loss*, $L\Big(\phi\big(f(\mathbf{x}^+)| < f(\mathbf{x}_i), \cdots, f(\mathbf{x}_{i+n-1}) > \big), \phi\big(f(\mathbf{x}^-)| < f(\mathbf{x}_i), \cdots, f(\mathbf{x}_{i+n-1}) > \big)\Big)$, in which $\phi(\cdot|\cdot)$ is a random distance-based scoring function and $L(\cdot,\cdot)$ is a loss function.

Note that RAMODO is an unsupervised framework, but it is flexible to incorporate *application-specific knowledge* (e.g., some labeled inliers or outliers) into the inlier or outlier candidate set when there exists such labeled data.

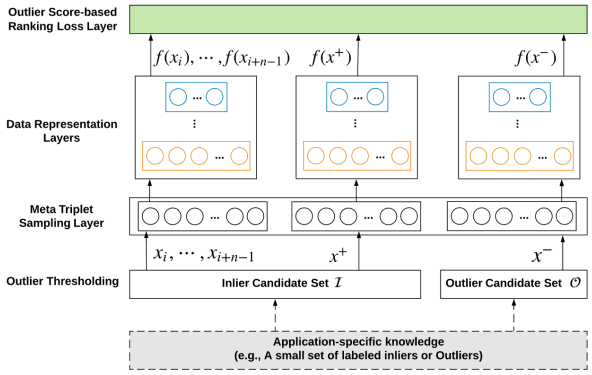

Figure 1: The Proposed RAMODO Framework. RAMODO learns a representation function $f(\cdot)$ to map D-dimensional input objects into a M-dimensional space, with $M \ll D$.

Why the representations learned by RAMODO are tailored for random distance-based outlier scoring? RAMODO optimizes the representations by encouraging $\phi\big(f(\mathbf{x}^-)| < f(\mathbf{x}_i), \cdots, f(\mathbf{x}_{i+n-1}) > \big)$ to be larger than $\phi\big(f(\mathbf{x}^+)| < f(\mathbf{x}_i), \cdots, f(\mathbf{x}_{i+n-1}) > \big)$, in which ϕ is equivalent to the random distance-based outlier scoring using $< f(\mathbf{x}_i), \cdots, f(\mathbf{x}_{i+n-1}) >$ as a random subsample. The representations learned by RAMODO are therefore tailored for the random distance-based detectors. For example, ϕ can be instantiated to be the popular KNN-based detectors using the K-th (or average) nearest neighbor distances in the query set as outlier scores.

How to guarantee the quality of triplet sampling? Having a reliable outlier candidate set is the key to the quality of the triplet sampling and the representation learning. However, in an unsupervised case, we do not know whether a given $\mathbf{x}$ is an outlier or not. We show in the RAMODO's instance below that two approaches can be used to produce an outlier candidate set of good quality. The first approach is to use state-of-the-art outlier detection methods and *Chebyshev*'s inequality to include the most likely outliers into the candidate set. Another approach is to incorporate a few labeled outliers into the outlier candidate set when such labeled data is available.

What are the intuitions behind RAMODO? Since the representation function f is guided by the scoring function ϕ, f only attains the information that is the most important for ϕ to distinguish outliers from inliers. By working on such a highly relevant space, ϕ is expected to obtain an accuracy that is comparable to, or better than that working on the original space even when $M \ll D$. In terms of efficiency, distance-based outlier scoring normally involves nearest neighbor searching, which can be very computationally costly in high-dimensional space since indexing methods like k-d tree fails to work. By working on a significantly lower-dimensional (e.g., $M \leq 30$) space, the distance computation and nearest neighbor searching in ϕ can be substantially accelerated when indexing methods are used.

4 A RAMODO INSTANCE: REPEN

RAMODO is instantiated to the method REPEN that learns data representations for the random nearest neighbor distance-based detector, Sp [30]. While Sp is a highly scalable outlier detector with significant accuracy improvement over several other popular distance-based methods and state-of-the-art high-dimensional methods [26, 30, 32], it still suffers from the curse of dimensionality. REPEN is customized for Sp to substantially improve its performance on ultrahigh-dimensional data.

4.1 Outlier Thresholding Using State-of-the-art Detectors and Cantelli's Inequality

We first use Sp to obtain an initial fairly good outlier ranking. Note that the original Sp [30] may perform unstably since it only uses one single subsample to define the outlierness. Following [26, 32], we use bootstrap aggregating to produce a bagging ensemble of Sp to obtain a more stable and reliable initial outlier ranking. While REPEN only uses the Sp-based ensemble for efficiency consideration, we may combine Sp with other methods to build heterogeneous ensembles [29] to further improve this initial outlier ranking.

Definition 4.1 (Sp-based Outlier Scoring). Given a data object $\mathbf{x}_i$, Sp defines its outlierness as follows:

$$r_i = \frac{1}{m}\sum_{j=1}^{m} nn\_dist(\mathbf{x}_i|\mathcal{S}_j), \tag{1}$$

where $\mathcal{S}_j \subset \mathcal{X}$ is a random data subsample, m is the ensemble size, and $nn\_dist(\cdot|\cdot)$ returns the nearest neighbor distance of $\mathbf{x}_i$ in $\mathcal{S}_j$.

We then use the *Cantelli*'s inequality [11], a one-sided *Chebyshev*'s inequality, to define pseudo outliers.

Definition 4.2 (Cantelli's Inequality-based Outlier Thresholding). Given an outlier score vector $\mathbf{r} \in \mathbb{R}^N$, in which large values indicate high outlierness, and let μ and δ^2 be its expected value and variance, then *the outlier candidate set* O is defined as:

$$O = \{\mathbf{x}_i | r_i \geq \mu + \alpha\delta\},\ \forall \mathbf{x}_i \in \mathcal{X}, r_i \in \mathbf{r}, \tag{2}$$

where $\alpha \geq 0$ is user-defined based on a desired false positive bound.

We show in Section 5.2 that Eqn. (2) is equivalent to selecting an outlier candidate set with a false positive upper bound of $\frac{1}{1+\alpha^2}$. After determining O, we obtain the *inlier candidate set* by $\mathcal{I} = \mathcal{X}\backslash O$.

4.2 Triplet Sampling Based on Outlier Scores

An outlier score-based importance sampling is then used to generate meta triplet samples. We first sample n query objects from the inlier candidate set $\mathcal{I}$ according to their outlier scores. The probability of a data object $\mathbf{x}_i \in \mathcal{I}$ being sampled as the query object is inversely proportional to its outlier score and is defined as follows:

$$p(\mathbf{x}_i) = \frac{\mathbb{Z} - r_i}{\sum_{t=1}^{|\mathcal{I}|}[\mathbb{Z} - r_t]}, \tag{3}$$

where $\mathbb{Z} = \sum_{t=1}^{|\mathcal{I}|} r_t$. The importance sampling offers high probability of choosing a set of representative inliers as query objects.

We then sample a positive example $\mathbf{x}^+$ from $\mathcal{I}$ using uniform sampling. Instead of importance sampling, we use uniform sampling to diversify the positive examples in different triplets.

We further sample a negative example $\mathbf{x}^-$ from the outlier candidate set O. Importance sampling is used here to obtain high probability of choosing the most likely outliers as negative examples. Given a data object $x_j \in O$, its probability of being chosen as a negative example is defined as:

$$p(\mathbf{x}_j) = \frac{r_j}{\sum_{t=1}^{|O|} r_t}. \tag{4}$$

4.3 A Shallow Data Representation

One single hidden layer is defined below to learn a shallow data representation. The shallow representation is used for two main reasons. (i) In many ultrahigh-dimensional data, we often have $N \ll D$. As a result, we may not have sufficient data to train a deep representation. (ii) Deep representation learning requires extensive computation for data sets with millions of features.

Definition 4.3 (Single-layer Fully-connected Representations). Given an input $\mathbf{x}$, it is mapped to a new space of M dimensions by:

$$f_\Theta(\mathbf{x}) = \{\psi(\mathbf{w}_1^\intercal \mathbf{x}), \psi(\mathbf{w}_2^\intercal \mathbf{x}), \cdots, \psi(\mathbf{w}_M^\intercal \mathbf{x})\}, \tag{5}$$

where $\psi(\cdot)$ is an activation function, $\mathbf{w}_i \in \mathbb{R}^D$ is a weight vector, and $\Theta = \{\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_M\}$ is the parameter set to be learned.

The ReLu function $\psi(z) = max(0, z)$ is used because of its efficient computation and gradient propagation.

4.4 Ranking Loss Using Random Nearest Neighbor Distance-based Outlier Scores

A random nearest neighbor distance-based function is used in $\phi(\cdot|\cdot)$ for the loss function $L(\cdot,\cdot)$ to learn customized data representations for Sp. Let $Q =< f_\Theta(\mathbf{x}_i), \cdots, f_\Theta(\mathbf{x}_{i+n-1}) >$ be the query set, then given an object $\mathbf{x}$, REPEN defines the outlierness of $f_\Theta(\mathbf{x})$ using its nearest neighbor distance in Q:

$$\phi\big(f_\Theta(\mathbf{x})|Q\big) = nn\_dist\big(f_\Theta(\mathbf{x})|Q\big). \tag{6}$$

Hence, given a triplet $T = \big(Q, f_\Theta(\mathbf{x}^+), f_\Theta(\mathbf{x}^-)\big)$, our goal is to learn a representation function $f(\cdot)$ that results in

$$nn\_dist\big(f_\Theta(\mathbf{x}^+)|Q\big) < nn\_dist\big(f_\Theta(\mathbf{x}^-)|Q\big), \tag{7}$$

i.e., the pseudo outlier $\mathbf{x}^-$ has a larger nearest neighbor distance in Q than the pseudo inlier $\mathbf{x}^+$. We then define the following hinge loss function for the triplet T to achieve this goal:

$$J(\Theta; T) = L\Big(\phi\big(f_\Theta(\mathbf{x}^+)|Q\big), \phi\big(f_\Theta(\mathbf{x}^-)|Q\big)\Big) = \max\{0, c + nn\_dist\big(f_\Theta(\mathbf{x}^+)|Q\big) - nn\_dist\big(f_\Theta(\mathbf{x}^-)|Q\big)\}, \tag{8}$$

where c is a margin parameter that controls the difference between the two distances $nn\_dist\big(f_\Theta(\mathbf{x}^+)|Q\big)$ and $nn\_dist\big(f_\Theta(\mathbf{x}^-)|Q\big)$. The hinge loss is a convex function, which penalizes the violation of the ranking order in Eqn. (7) and encourages a separation between positive examples (inliers) and negative examples (outliers).

Given a set of triplets $\mathcal{T}$, our objective function becomes

$$\arg\min_{\Theta} \frac{1}{|\mathcal{T}|}\sum_{i=1}^{|\mathcal{T}|} J(\Theta; T_i). \tag{9}$$

Since the positive and negative examples come from the bottom-ranked and top-ranked objects of the ranking $\mathbf{r}$ respectively, given

an infinite number of T, REPEN optimizes the precision at top $|O|$ of the detector Sp by minimizing the loss function J. Additionally, REPEN does not learn the representation dimension M automatically, since there does not exist reliable supervision information to effectively guide the learning.

4.5 The Algorithm and Its Time Complexity

Algorithm 1 presents the procedure of REPEN. Step 1 uses an ensemble of Sp to produce an outlier ranking. Following [26, 30], a small subsampling size of 8 and an ensemble size of 50 are used to yield an initial reliable ranking $\mathbf{r}$. Step 2 yields the inlier and outlier candidate sets using the *Cantelli*'s inequality with $\alpha = 1.732$, which is equivalent to an outlier thresholding with a 25% false-positive upper bound. After a random weight initialization in Step 3, REPEN iteratively learns the parameters Θ using mini-batch gradient descents in Steps 4-13.

Specifically, Steps 6-9 generate one mini-batch of training triplets of size b. We found that REPEN using $n = 1$ performs stably, and more importantly, runs substantially faster than that using a larger n. According to the analysis in Section 5.1, using small n can also result in a small error bound of the representation learning. $n = 1$ is thus used. The parameter settings of n_epochs , $n\_batches$ and b are provided in Section 6.3.1. Additionally, the sampling in Steps 6-9 is with replacement. Step 10 then computes the average loss given one batch of triplet samples, in which square Euclidean distance is used in $nn\_dist(\cdot|\cdot)$ and $c = 1000$ is used in order to encourage large margins between inliers and outliers in the representation space. After that, Step 11 computes the partial derivatives w.r.t. the weight parameters and performs a gradient descent step using the ADADELTA optimization [35]. Lastly, Step 14 uses the function f to map all objects into the representation space.

Algorithm 1 *REPEN*

Input: $\mathcal{X} \in \mathbb{R}^D$ - ultrahigh-dimensional data objects
Output: $\mathcal{X}' \in \mathbb{R}^M$ - low-dimensional data representations
1: $\mathbf{r} \leftarrow \frac{1}{m}\sum_{i=1}^{m} nn\_dist(\mathcal{X}|\mathcal{S}_i)$
2: $O \leftarrow \{\mathbf{x}_i | r_i \geq \mu + a\delta\},\ \forall \mathbf{x}_i \in \mathcal{X}$, and $\mathcal{I} \leftarrow \mathcal{X} \setminus O$
3: Randomly initialize Θ
4: **for** $i = 1$ to n_epochs **do**
5: **for** $j = 1$ to $n\_batches$ **do**
6: $\mathcal{B}_q \leftarrow$ Sample b sets of n-sized query set from $\mathcal{I}$ using Eqn. (3)
7: $\mathcal{B}_p \leftarrow$ Sample b objects from $\mathcal{I}$ using uniform sampling
8: $\mathcal{B}_n \leftarrow$ Sample b objects from O using Eqn. (4)
9: Generate the triplet set $\mathcal{T} = \{T_1, \cdots, T_b\}$, with each triplet $T_s = (< \mathbf{x}_t, \cdots, \mathbf{x}_{t+n-1} >, \mathbf{x}^+, \mathbf{x}^-)$ from the s-th element of $\mathcal{B}_q$, $\mathcal{B}_p$, and $\mathcal{B}_e$ respectively
10: $J(\Theta) \leftarrow \frac{1}{b}\sum_{s=1}^{b} J(\Theta; T_s)$
11: Perform a gradient descent step w.r.t. the parameters in Θ
12: **end for**
13: **end for**
14: $\mathcal{X}' \leftarrow f_\Theta(\mathcal{X})$
15: **return** $\mathcal{X}'$

Step 1 requires $O(mND|\mathcal{S}|)$ to obtain the nearest neighbor distance-based outlier scores for all objects. Step 2 requires $O(N)$ to scan over the outlier ranking list to produce the inlier and outlier candidate sets. Since n_epochs, $n\_batches$ and b are small constant, the time complexity of the optimization in Steps 4-13 is mainly determined by the computation of the loss function in Step 9 and the gradient descents in Step 10. Since REPEN has only one fully connected hidden layer, both the loss function and gradient descents have a time complexity linear w.r.t. D and M. After the optimization, REPEN takes $O(NDM)$ to map the original data into the representation space. The overall time complexity of REPEN is expected to be linear w.r.t. data size and dimensionality size.

4.6 Leveraging A Few Labeled Outliers to Improve Triplet Sampling

Given a small set of l labeled outliers, a minor change to the sampling of negative examples in REPEN (i.e., Step 8 in Algorithm 1) can well leverage them to improve the quality of triplets. Specifically, we first sample $\frac{b}{2}$ objects from the outlier candidate set O using the importance sampling described in Eqn. (4); and then the other half of the negative examples are sampled from the set of l labeled outliers using uniform sampling with replacement. The negative examples from O may cover types of outliers that do not present in the labeled outliers, while sampling from the labeled outlier pool guarantees the presence of outliers in the negative examples. Hence, these two sources of negative examples can well complement each other for more effective subsequent representation learning.

Note that the above method assumes the availability of only a few labeled outliers since it is typically very difficult or costly to collect labeled outliers. We may sample all negative examples from the labeled outliers in the cases where the number of labeled outliers is sufficiently large to cover different types of outliers.

5 THEORETICAL FOUNDATION OF REPEN

5.1 Upper Error Bound for the Representation Learning

We show below that REPEN can obtain an upper error bound for its representation learning based on the theorems in [30]. An equivalent form of K-th nearest neighbor distance-based outliers is the $DB(\beta, \delta)$-outliers [17] that defines an object $\mathbf{x}$ as an outlier if $|\{\mathbf{x}' \in \mathcal{X} | dist(\mathbf{x}, \mathbf{x}') > \delta\}| \geq \beta N$, where $dist$ is a distance measure. We denote the $DB(\beta, \delta)$-outliers by $O(\beta; \delta)$, and let $\mathcal{I}(\beta; \delta) = \mathcal{X} \setminus O(\beta; \delta)$ be the rest of instances. $\mathcal{I}_s(\beta; \delta)$ is a subset of $\mathcal{I}(\beta; \delta)$ such that $\min_{\mathbf{x}' \in \mathcal{I}_s(\beta;\delta)} dist(\mathbf{x}, \mathbf{x}') > \delta$ for every outlier $\mathbf{x} \in O(\beta; \delta)$. Let a δ-radius partition $\mathcal{P}_\delta$ of $\mathcal{I}_s(\beta; \delta)$ be a set of h non-empty disjoint clusters, i.e., $\mathcal{I}_s(\beta; \delta) = \cup_{i=1}^{h} C_i$, such that each cluster satisfies $\max_{\mathbf{x}, \mathbf{x}' \in C} dist(\mathbf{x}, \mathbf{x}') < \delta$. It shows in [30] that

$$p\left(nn\_dist(\mathbf{x}|\mathcal{S}) \leq nn\_dist(\mathbf{x}'|\mathcal{S})\right) < 1 - \gamma^n \max_{\mathcal{P}_\delta} \eta_\delta(n),\ \forall \mathbf{x} \in O(\beta; \delta), \forall \mathbf{x}' \in \mathcal{I}(\beta; \delta), \quad (10)$$

$\gamma = \frac{|\mathcal{I}_s(\beta;\delta)|}{N}$ and $\eta_\delta(n) = \sum_{\forall i; n_i \geq 0} g(n_1, \cdots, n_h; n, p_1, \cdots, p_h)$, in which $n = |\mathcal{S}|$ and g is a probability mass function of the multinomial distribution with $\sum_{i=1}^{h} n_i = n$ and $\sum_{i=1}^{h} p_i = 1$. By replacing $\mathbf{x}$ with $f_\Theta(\mathbf{x})$ and $\mathcal{S}$ with $\mathcal{Q}$, we obtain the following error bound:

$$p\Big(nn\_dist\big(f_\Theta(\mathbf{x}^-)|\mathcal{Q}\big) \leq nn\_dist\big(f_\Theta(\mathbf{x}^+)|\mathcal{Q}\big)\Big) < 1 - \gamma^n \max_{\mathcal{P}_\delta} \eta_\delta(n). \quad (11)$$

Different from [30] that uses $nn\_dist(\cdot|\cdot)$ to identify outliers directly, REPEN uses the difference between $nn\_dist\big(f_\Theta(\mathbf{x}^+)|Q\big)$ and $nn\_dist\big(f_\Theta(\mathbf{x}^-)|Q\big)$ to perform the gradient descent steps in its representation learning. Hence, Eqn. (11) can be seen as an upper error bound for the representation learning.

We often have a sufficiently large γ in Eqn. (11). This is because $\mathcal{I}_s(\beta;\delta)$ represents a set of inliers that demonstrate very different characteristics from outliers, and the proportion of such inliers can be very large in practice. Hence, the error bound is often a small value when using a small query set Q (i.e., a small n). In such cases, the expressiveness of REPEN's representations is well guaranteed when the pseudo outliers $\mathbf{x}^-$ and inliers $\mathbf{x}^+$ are genuine.

5.2 Reliable Triplet Sampling

Since it has a high probability of hitting genuine inliers given their large proportion, having genuine outliers in the outlier candidate set is the key to the representation learning. Thus, the above error bound has a good guarantee when a few labeled outliers are available. In the case of only having the score vector $\mathbf{r}$, our problem is to determine an outlierness threshold for $\mathbf{r}$, such that we obtain an outlier candidate set with small false positives. We show below that the *Cantelli*'s inequality can help achieve this goal. Given μ and δ^2 be the mean and variance of $\mathbf{r}$, we have $p(r_i \geq \mu + \epsilon) \leq \frac{\delta^2}{\delta^2+\epsilon^2}$ with $\epsilon \geq 0$ per the *Cantelli*'s inequality. By replacing $\epsilon = \alpha\delta$, we obtain

$$p(r_i \geq \mu + \alpha\delta) \leq \frac{1}{1+\alpha^2}, \tag{12}$$

which states that most $\mathbf{r}$ values distribute very closely to the expected value, with the probability of up to $\frac{1}{1+\alpha^2}$ that a few exceptions occur. In other words, when $\mu + \alpha\delta$ is used as the threshold of labeling outliers, we have a false positive upper bound of $\frac{1}{1+\alpha^2}$. Note that the *Cantelli*'s inequality holds for any distribution that has statistical mean and variance, and thus makes no assumption on specific probability distributions. This property enables us to obtain a good outlier candidate set even when $\mathbf{r}$ follows different distributions in different data sets.

6 EXPERIMENTS

6.1 Data Sets

As shown in Table 1, eight real-world ultrahigh-dimensional data sets are used, which cover diverse domains, e.g., malicious URL detection, cancer detection, molecular bioactivity detection, and Internet advertisement detection. OvarianTumour (*OT*) is taken from the RSCTC Discovery Challenge for medical diagnosis and treatment[1]. *Webspam* is taken from the Pascal Large Scale Learning Challenge[2]. *URL* consists of 120-day collection of malicious and benign URLs [20]. Since *URL* contains evolving URLs data, the first-week subset of this collection is used. *R8*[3] and *News20*[4] are text classification data sets. *AD* is a data set for detecting Internet advertisements, LungCancer (*LC*) is for cancer detection, and *p53* is about abnormal protein activity detection. These three data sets are available at UCI Machine Learning Repository[5]. The data sets *OT*, *URL*, *AD*, *LC*, *p53* and *Webspam* contain real outliers, but *URL* and *Webspam* were originally prepared for balanced classification tasks. Following the literature [6, 16, 37], *URL* and *Webspam* are converted to outlier detection data sets with 2% outliers by downsampling the small class or the positive class. This downsampling approach is also applied to *News20*. For *R8* that is very imbalanced, we follow [16, 18, 29, 32] to treat the rare classes as outliers and the largest class as inliers. After the above data preparation, all data sets contain (semantically) real outliers.

[1] http://tunedit.org/challenge/RSCTC-2010-A
[2] ftp://largescale.ml.tu-berlin.de/largescale/
[3] http://csmining.org/tl_files/Project_Datasets/r8_r52/r8-train-all-terms.txt
[4] https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary/news20.binary.bz2
[5] https://archive.ics.uci.edu/ml/index.php

6.2 Performance Evaluation Methods

We evaluate the effectiveness of the representations by the performance of the subsequent outlier detector Sp. Sp is used with the recommended settings as in [26]. Given a set of data objects in the original space or the representation space, Sp returns a full ranking list of data objects based on their outlier scores. After that, many performance measures, such as the area under Receiver Operating Characteristic curve (AUC) and average precision, are available for evaluating the quality of the ranking. A detailed analysis of the advantages and drawbacks of these measures for unsupervised outlier detection can be found in [6]. Following the literature [6, 16, 30, 32, 37], the popular measure AUC is used. AUC inherently considers the class-imbalance nature of outlier detection, making it comparable across data sets with different outlier proportions [6]. An AUC value close to 0.5 indicates a random ranking of the objects. Higher AUC indicates better detection performance.

The paired *Wilcoxon* signed rank test is used to examine the significance of the performance of REPEN against its competitors. For algorithms or data sets that involve sampling, their AUCs are the averaged results over 10 independent runs.

The runtime presented in our experiments is calculated at a node in a 2.8GHz Titan cluster with 256GB memory.

6.3 Effectiveness in Real-world Data with Thousands to Millions of Features

6.3.1 Experiment Settings. We compare the AUC performance and detection runtime of Sp in the low-dimensional representation space learned by REPEN and in the original high-dimensional space to evaluate the effectiveness of REPEN. Sp is applied with the same settings, subsampling size set to 8 and ensemble size set to 50, in both the representation space and original space. For the nearest neighbor searching, k-d tree indexing is used.

REPEN with the default representation dimension $M = 20$ is used. In the optimization, $n\_epochs = 30$, $b = 256$ and 5,000 samples per epoch (equivalent to $\frac{5,000}{b}$ batches per epoch) are used by default. Note that Sp is very cost-effective, which only requires the presence of a few representative inliers in a subsample to effectively distinguish outliers from inliers [26, 30, 32]. Therefore, although the number of possible triplets is huge for large data sets, our experiments showed that these settings enable the optimization to achieve the convergence and impressive detection accuracy for all the data sets except *p53*, in which outliers are deeply mixed with inliers and thus we increase the number of samples per epoch to its data size to sufficiently train the representation model. Both Sp and REPEN are implemented in Python [6].

[6] The code of REPEN is available at https://sites.google.com/site/gspangsite/sourcecode.

6.3.2 Findings - REPEN Enables the Detector Sp to Obtain Significantly Better AUC Performance and Two Orders of Magnitude Speedup. Table 1 shows the AUC performance and *detection runtime*[7] of Sp in the original data space and the 20-D space output by REPEN. Although REPEN retains only 20 dimensions, which are at most 1.3% of the dimensionality size in the original data, the REPEN-enabled Sp is either substantially better than, or roughly the same as, that working on the original space. Our significance test based on the AUC performance on the eight data sets obtains a p-value of 0.0156, indicating that the REPEN-enabled Sp significantly outperforms the original Sp at the 95% confidence level.

Table 1: AUC and Detection Runtime of the Original Sp (ORG) and the REPEN-enabled Sp (REPEN). D is the dimensionality size and N is the data size. *IMP* and *SU* indicate the AUC improvement and speedup of REPEN over ORG, respectively.

			AUC			Runtime (s)		
Data	D	N	ORG	REPEN	*IMP*	ORG	REPEN	*SU*
AD	1,555	3,279	0.7117	**0.8533**	19.90%	0.16	0.10	2
LC	3,312	145	0.9149	**0.9283**	1.47%	0.59	0.02	38
p53	5,408	16,772	0.6686	**0.6817**	1.95%	230.39	0.91	253
R8	9,467	3,974	0.8602	**0.9080**	5.56%	0.21	0.12	2
OT	54,621	265	0.6976	**0.7627**	9.34%	23.50	0.04	529
News20	1,355,191	10,201	0.5361	**0.5822**	8.60%	5.30	0.29	18
URL	3,231,961	89,063	0.7556	**0.7733**	2.35%	16.40	2.48	7
Webspam	16,609,143	215,001	**0.8781**	0.8713	-0.78%	1879.68	6.08	309

Particularly, REPEN enables Sp to obtain about 1.5% to 20% AUC improvement on most data sets. This is mainly due to the fact that there often exists only a small percentage of features that are relevant to outlier detection, since outliers are the minority objects; the large proportion of irrelevant features in the original space renders Sp less effectively. In contrast, the customized objective function enables REPEN to effectively preserve the most important information for Sp. Therefore, Sp performs outlier detection in a highly relevant space when working with our representation space, resulting in the above substantial improvement.

In terms of runtime, by working on our 20-D representation space, Sp runs up to two-order of magnitude faster than on the original space. This is because the original data has very high dimensionality, data indexing methods fail to work. In contrast, k-d tree indexing excels at the 20-D representations, which enables Sp to achieve a remarkable speedup. Such speedup is striking for the two dense data sets, *p53* and *OT*, and the large sparse data set, *Webspam*. Note that we have used the efficient implementation of distance measures that is specifically designed for high-dimensional sparse data in Sp, so Sp runs very fast in the small sparse data sets, *News20* and *URL*.

6.4 Comparing to State-of-the-art Representation Learning Competitors

6.4.1 Experiment Settings. This section compares REPEN with autoencoder (AE) [13], Hessian locally linear embedding (HLLE) [10], sparse random projection (SRP) [19], and coherent pursuit (CoP) [28]. These methods are chosen because they are state-of-the-art methods of four popular unsupervised representation learning approaches, i.e., neural network-based approaches, manifold learning, random projections, and robust PCA approaches. All methods learn a mapping from the original space to a space of the same dimension ($M = 20$) to examine their effectiveness of learning low-dimensional representations. AE uses the same training settings as REPEN. Since HLLE requires its neighborhood size $K > \frac{M(M+3)}{2}$ in its optimization, $K = 250$ is used. SRP and CoP do not have additional parameters. AE, HLLE and SRP are implemented using the built-in functions in the Scikit-learn Python library. CoP is taken from their authors in MATLAB.

6.4.2 Findings - REPEN Performs Substantially Better and More Stably Than the Competitors. Table 2 demonstrates the AUC performance of Sp using the representations output by REPEN, AE, HLLE, SRP, and CoP. REPEN performs significantly better than AE and SRP at the 95% confidence level, and is comparably better than HLLE and Cop. In general, REPEN enables Sp to achieve substantial improvement over its competitors on the data sets *AD*, *R8*, *OT*, *URL*, and *Webspam*, in which Sp has fairly good performance on the original space. For example, the REPEN-based Sp obtains about 14% to 32% AUC improvement over all its competitors on *AD*, 5% to 42% improvement over all its competitors on *OT*, and 27% to 33% improvement over AE and SRP on *URL*. This is because when Sp obtains good performance on the original data space, the triplet sampling in REPEN yields higher-quality training triplet samples, which lead its optimization to better representations for Sp. The four competitors learn representations independently of the detector, and thus they are unable to achieve this merit. Note that the representation learning in the methods AE and SRP is simpler than the other methods, and they often require a sufficiently large number of representation features (e.g., hundreds to thousands of features) to perform fairly well. Therefore, they perform poorly in most data sets when they only retain 20 representation features.

Table 2: AUC Performance of Sp using REPEN, AE, HLLE, SRP, and CoP. '•' indicates out-of-memory errors while '○' denotes algorithmic constraint violation errors.

Data	REPEN	AE	HLLE	SRP	CoP
AD	**0.8533**	0.6848	0.6462	0.7249	0.7483
LC	**0.9283**	0.9266	○	0.9041	0.9161
p53	**0.6817**	0.5624	0.3870	0.6874	0.6805
R8	0.9080	0.7698	0.8764	0.8494	**0.9326**
OT	**0.7627**	0.5355	0.7273	0.6708	0.7247
News20	0.5822	0.5154	**0.6206**	0.4435	•
URL	**0.7733**	0.6040	•	0.5774	•
Webspam	0.8713	**0.8766**	•	0.6390	•
	p-value	0.0234	0.3125	0.0156	0.3125

Also, REPEN performs much more stably than the other methods across the data sets. REPEN obtains the best performance in five data sets, with the other three ranked in second. Although AE, HLLE and CoP respectively obtain one best performance, they perform poorly in several other data sets, e.g., the performance of AE and HLLE on *AD* and *p53*. One main reason here is that AE, HLLE and CoP are not specifically designed for the outlier detectors, and as a result, the best representations they obtain may be suboptimal for

[7]The detection runtime is the execution time of online detection, which does not include the data loading and offline training time.

Sp. This reinforces the importance of incorporating the subsequent detection methods into the representation learning.

CoP outperforms REPEN on *R8*. This may be because CoP is able to preserve information for detecting some structural outliers in *R8*, while REPEN fails to do that. HLLE outperforms REPEN on *News20*. This may be because a large neighborhood size is required to detect more outliers than REPEN does. We plan to instantiate other RAMODO's instances to handle these two issues in future.

6.5 The Capability of Leveraging Labeled Outliers as Prior Knowledge

6.5.1 Experiment Settings. This section examines the capability of REPEN in incorporating a small set of labeled outliers into its representation learning. Since we downsampled one class in *News20*, *URL*, and *Webspam* to create outliers, the rest of data objects in the downsampled class can be used as the pool of labeled outliers. We randomly pick up l objects from the pool and add them into our training using the method described in Section 4.6. $l \in \{1, 5, 10, 20, 40, 80\}$ is used. The AUC performance of using a set of l labeled outliers is averaged over 10 independent runs.

6.5.2 Findings - REPEN Achieves Up to 32% AUC Improvement by Leveraging 1-80 Labeled Outliers. Figure 2 illustrates the AUC performance of REPEN on *News20* and *URL* using different numbers of labeled outliers. Similar results are observed on *Webspam*. The results of REPEN on *News20* and *URL* in Table 2 are used as the baselines, i.e., the performance of REPEN without using the labeled outliers. The AUC performance increases quickly with the increasing of the number of known outliers. For the complex data set *News20*, in which REPEN originally obtains an AUC of only 0.5822, REPEN can harness the limited number of available outliers to improve the AUC up to 0.7707. For the easier data set *URL*, REPEN improves the AUC performance from 0.7733 to 0.9160 when the labeled outliers are provided. This is mainly because the labeled outlier-based prior knowledge largely improves our training data quality and helps identify more application-relevant outliers.

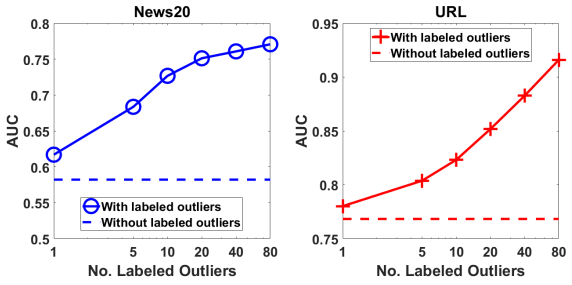

Figure 2: AUC Results of REPEN-based Sp Using Labeled Outliers.

We further investigate the changes of outlier statistics of the two data sets w.r.t. the number of labeled outliers. The results are reported in Table 3. The number of labeled data objects is less than 1% of the unlabeled data according to the $\frac{l}{N}$ factor. Also, the proportion of (labeled and unlabeled) outliers (i.e., $\frac{l+N_o}{N}$) remains to be very small for both data sets. These two data factors indicate that the intrinsic complexity of the problem does not decrease due to the available of a few labeled outliers. The third factor $\frac{l}{N_o}$ represents the ratio of the number of the labeled outliers to the number of unlabeled outliers in the data sets. For *URL*, REPEN can use the labeled outliers that account for only up to 4.50% of unlabeled outliers to achieve 18% AUC improvement. The $\frac{l}{N_o}$ in *News20* is much larger than that in *URL*. In such cases, REPEN leverages the extra labeled outliers to achieve more substantial improvement, resulting in more than 32% AUC improvement. These results show the strong capability of REPEN in making very effective use of the small number of labeled outliers in different cases.

Table 3: Outlier Statistics of News20 and URL w.r.t. the Number of Labeled Outliers. l is the number of labeled outliers, N_o is the number of unlabeled outliers in the data, and N is the data size.

Factor	Data	1	5	10	20	40	80
$\frac{l}{N}$	News20	0.01%	0.05%	0.10%	0.20%	0.39%	0.78%
	URL	0.00%	0.01%	0.01%	0.02%	0.04%	0.09%
$\frac{l+N_o}{N}$	News20	2.01%	2.05%	2.10%	2.20%	2.39%	2.78%
	URL	2.00%	2.00%	2.01%	2.02%	2.04%	2.09%
$\frac{l}{N_o}$	News20	0.49%	2.45%	4.90%	9.80%	19.61%	39.22%
	URL	0.06%	0.28%	0.56%	1.13%	2.25%	4.50%

6.6 Sensitivity Test w.r.t. the Representation Dimension

6.6.1 Experiment Settings. We examine the sensitivity of the performance of REPEN w.r.t. the number of representation features, M, on all the eight data sets. A wide range of M values, $\{1, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$, is used.

6.6.2 Findings - REPEN Performs Stably w.r.t. Different Numbers of Representation Features. Figure 3 presents the AUC performance of REPEN-based Sp using different representation dimensions. REPEN performs very stably when M is set between 10 and 100. Neural network-based methods can represent up to $O(2^M)$ concepts when using M neural nodes [5]. Therefore, REPEN can theoretically learn complex representations (i.e., up to $O(2^{10})$ concepts) when $M = 10$ is used, making it sufficient for most data sets. The expressiveness of REPEN's representations is expected to be increased w.r.t. the representation dimensions. However, the more complex representations we intend to learn, the larger amount of high-quality training data is required. Due to the unsupervised nature of our triplet sampling, our training triplets may not informative enough for REPEN to learn more powerful higher-dimensional (e.g., $M > 10$) representations. Hence, the performance of REPEN flattens after $M = 10$ (this issue may be addressed by incorporating labeled outliers to generate high-quality training triplets, as explained in the discussion below). We recommend $M = 20$ to attain stable accuracy and at the same time make full use of k-d tree indexing for distance computation in the representation space.

It is interesting that REPEN using $M = 1$ can perform as well as that using a large M in the data sets *R8*, *News20* and *URL*. This phenomenon may be due to two main reasons. On one hand, in the simplest case that the inliers and the outliers are well separable by one linear/non-linear decision boundary, learning a one-dimensional representation space (i.e., $M = 1$) is sufficient, since the problem is similar to a one-class or binary classification problem. On the other

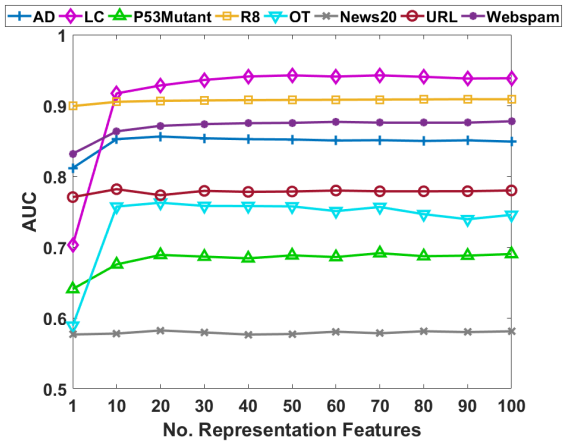

Figure 3: AUC Performance w.r.t. Representation Dimension.

hand, the optimization in REPEN may also encounter the aforementioned training data quality problem. Improving the triplet quality, e.g., by incorporating a few labeled outliers, can upgrade the performance of REPEN using a large M. For example, without using labeled outliers, REPEN obtains similar performance by using $M = 1$ and $M = 20$ on *News20* and *URL*; however, when we use $l = 80$ labeled outliers to improve the triplet quality and learn *1-D* representations, we only obtain AUC results of 0.7201 and 0.8320 on respective *News20* and *URL*, which substantially underperform the AUC results of 0.7707 and 0.9160 based on *20-D* representations, as shown in Figure 2. This again highlights the effectiveness of our method in using the limited number of labeled outliers to improve the triplet quality.

6.7 Scalability Test

6.7.1 Experiment Settings. We generate synthetic data sets by varying the data size in [1000, 125000] of a 10,000-D data set for the scaleup test w.r.t. data size, and likewise, varying the dimensions w.r.t. a fixed data size (i.e., 10,000) for the test w.r.t. dimension. The execution time below includes both training and testing runtime.

6.7.2 Findings - REPEN Achieves Linear Time Complexity w.r.t. Both Data Size and Dimensionality. The scalability test results are illustrated in Figure 4. REPEN has time complexity linear w.r.t. both data size and dimensionality, which justifies our complexity analysis in Section 4.5. REPEN is much faster than HLLE because HLLE requires nearest neighbor searching in the entire data set that has quadratic time complexity w.r.t. data size. SRP is the most efficient method since it only requires the fast operation of random matrix projection. REPEN has a similar network architecture as AE, but it runs slower than AE as it requires more distance computation in its representation learning.

7 CONCLUSIONS

This paper introduces the RAMODO framework and its instance REPEN that learn customized low-dimensional representations of ultrahigh-dimensional data for random distance-based detectors. By unifying the two correlated tasks, representation learning and outlier detection, we gain three main benefits. One benefit is that we

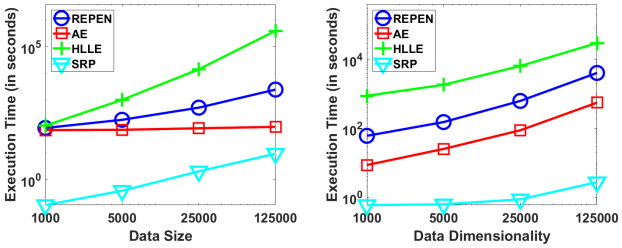

Figure 4: Scalability Test w.r.t. Data Size and Dimensionality. CoP is excluded since it is implemented in a programming language different from the others. Logarithmic scale is used in both axes.

can produce more optimal and stable representations for a specific outlier detector. This is verified by substantially better and more stable detection performance of REPEN, compared to four state-of-the-art representation learning competitors that separate these two correlated tasks. The second benefit is that our method can easily and effectively incorporate application-specific knowledge to learn more application-relevant representations for the given outlier detector, which helps REPEN achieve up to 32% AUC improvement. The third benefit is that we can effectively represent ultrahigh-dimensional data using very low-dimensional representations (i.e., 20-dimensional representations) and our representations perform stably w.r.t. a wide range of representation dimensions.

To effectively learn customized representations for the detector, it is critical to generate high-quality training triplets. We show that (i) the combination of state-of-the-art outlier detectors and *Cantelli*'s inequality can help generate sufficiently reliable triplets to learn expressive representations; and (ii) the availability of a few labeled outliers can be further leveraged to substantially improve the triplet quality and learn better representations.

We are implementing other instances of RAMODO that define sophisticated distance-based outlier scoring to better represent more complex data.

ACKNOWLEDGEMENTS

This work is partially supported by the ARC Discovery Grant DP180100966.

REFERENCES

[1] Charu Aggarwal and S Yu. 2005. An effective and efficient algorithm for high-dimensional outlier detection. *The VLDB Journal* 14, 2 (2005), 211–221.
[2] Charu C Aggarwal. 2017. Supervised outlier detection. In *Outlier Analysis*. Springer, 219–248.
[3] Fatemeh Azmandian, Ayse Yilmazer, Jennifer G Dy, Javed Aslam, David R Kaeli, et al. 2012. GPU-accelerated feature selection for outlier detection using the local kernel density ratio. In *ICDM*. IEEE, 51–60.
[4] Stephen D Bay and Mark Schwabacher. 2003. Mining distance-based outliers in near linear time with randomization and a simple pruning rule. In *SIGKDD*. ACM, 29–38.
[5] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35, 8 (2013), 1798–1828.
[6] Guilherme O Campos, Arthur Zimek, Jörg Sander, Ricardo JGB Campello, Barbora Micenková, Erich Schubert, Ira Assent, and Michael E Houle. 2016. On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery* 30, 4 (2016), 891–927.
[7] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. 2011. Robust principal component analysis? *Journal of the ACM* 58, 3 (2011), 11.
[8] Lei Cao, Yizhou Yan, Caitlin Kuhlman, Qingyang Wang, Elke A Rundensteiner, and Mohamed Eltabakh. 2017. Multi-Tactic Distance-based Outlier Detection. In

ICDE. IEEE, 959–970.

[9] Samuel A Danziger, Jue Zeng, Ying Wang, Rainer K Brachmann, and Richard H Lathrop. 2007. Choosing where to look next in a mutation sequence space: Active Learning of informative p53 cancer rescue mutants. *Bioinformatics* 23, 13 (2007).

[10] David L Donoho and Carrie Grimes. 2003. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences* 100, 10 (2003), 5591–5596.

[11] Devdatt P Dubhashi and Alessandro Panconesi. 2009. *Concentration of measure for the analysis of randomized algorithms*. Cambridge University Press.

[12] Jingrui He and Jaime Carbonell. 2008. Nearest-neighbor-based active learning for rare category detection. In *NIPS*. 633–640.

[13] Geoffrey E Hinton and Ruslan R Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *Science* 313, 5786 (2006), 504–507.

[14] Songlei Jian, Longbing Cao, Guansong Pang, Kai Lu, and Hang Gao. 2017. Embedding-based representation of categorical data by hierarchical value coupling learning. In *IJCAI*. AAAI Press, 1937–1943.

[15] Enric Junqué de Fortuny, Marija Stankova, Julie Moeyersoms, Bart Minnaert, Foster Provost, and David Martens. 2014. Corporate residence fraud detection. In *SIGKDD*. ACM, 1650–1659.

[16] Fabian Keller, Emmanuel Muller, and Klemens Bohm. 2012. HiCS: High contrast subspaces for density-based outlier ranking. In *ICDE*. IEEE, 1037–1048.

[17] Edwin M Knox and Raymond T Ng. 1998. Algorithms for mining distance-based outliers in large datasets. In *VLDB*. 392–403.

[18] Aleksandar Lazarevic and Vipin Kumar. 2005. Feature bagging for outlier detection. In *SIGKDD*. ACM, 157–166.

[19] Ping Li, Trevor J Hastie, and Kenneth W Church. 2006. Very sparse random projections. In *SIGKDD*. ACM, 287–296.

[20] Justin Ma, Lawrence K Saul, Stefan Savage, and Geoffrey M Voelker. 2009. Identifying suspicious URLs: An application of large-scale online learning. In *ICML*. ACM, 681–688.

[21] Alireza Makhzani and Brendan J Frey. 2015. Winner-take-all autoencoders. In *NIPS*. 2791–2799.

[22] Mary McGlohon, Stephen Bay, Markus G Anderle, David M Steier, and Christos Faloutsos. 2009. SNARE: A link analytic system for graph labeling and risk detection. In *SIGKDD*. ACM, 1265–1274.

[23] Guansong Pang, Longbing Cao, Ling Chen, Defu Lian, and Huan Liu. 2018. Sparse Modeling-based Sequential Ensemble Learning for Effective Outlier Detection in High-dimensional Numeric Data. In *AAAI*.

[24] Guansong Pang, Longbing Cao, Ling Chen, and Huan Liu. 2016. Unsupervised feature selection for outlier detection by modelling hierarchical value-feature couplings. In *ICDM*. IEEE, 410–419.

[25] Guansong Pang, Longbing Cao, Ling Chen, and Huan Liu. 2017. Learning Homophily Couplings from Non-IID Data for Joint Feature Selection and Noise-Resilient Outlier Detection. In *IJCAI*. 2585–2591.

[26] Guansong Pang, Kai Ming Ting, and David Albrecht. 2015. LeSiNN: Detecting anomalies by identifying least similar nearest neighbours. In *ICDM Workshop*. IEEE, 623–630.

[27] Dan Pelleg and Andrew W Moore. 2005. Active learning for anomaly and rare-category detection. In *NIPS*. 1073–1080.

[28] Mostafa Rahmani and George Atia. 2017. Coherence pursuit: Fast, simple, and robust subspace recovery. In *ICML*. 2864–2873.

[29] Shebuti Rayana and Leman Akoglu. 2016. Less is more: Building selective anomaly ensembles. *ACM Transactions on Knowledge Discovery from Data* 10, 4 (2016), 42.

[30] Mahito Sugiyama and Karsten Borgwardt. 2013. Rapid distance-based outlier detection via sampling. In *NIPS*. 467–475.

[31] Acar Tamersoy, Kevin Roundy, and Duen Horng Chau. 2014. Guilt by association: Large scale malware detection by mining file-relation graphs. In *SIGKDD*. ACM, 1524–1533.

[32] Kai Ming Ting, Takashi Washio, Jonathan R Wells, and Sunil Aryal. 2017. Defying the gravity of learning curve: A characteristic of nearest neighbour anomaly detectors. *Machine Learning* 106, 1 (2017), 55–91.

[33] Laurens Van Der Maaten, Eric Postma, and Jaap Van den Herik. 2009. Dimensionality reduction: A comparative review. *Journal of Machine Learning Research* 10 (2009), 66–71.

[34] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research* 11 (2010), 3371–3408.

[35] Matthew D Zeiler. 2012. ADADELTA: An adaptive learning rate method. *CoRR* abs/1212.5701 (2012).

[36] Chong Zhou and Randy C Paffenroth. 2017. Anomaly detection with robust deep autoencoders. In *SIGKDD*. ACM, 665–674.

[37] Arthur Zimek, Matthew Gaudet, Ricardo JGB Campello, and Jörg Sander. 2013. Subsampling for efficient and effective unsupervised outlier detection ensembles. In *SIGKDD*. ACM, 428–436.

[38] Arthur Zimek, Erich Schubert, and Hans-Peter Kriegel. 2012. A survey on unsupervised outlier detection in high-dimensional numerical data. *Statistical Analysis and Data Mining* 5, 5 (2012), 363–387.