SCIREA Journal of Information Science and Systems Science

http://www.scirea.org/journal/ISSS

May 17, 2021

Volume 5, Issue 2, April 2021

Detection and correction of mislabeled samples based on graph structure

Junyan Li*, Xinxing Wu

University of Kentucky, Lexington, Kentucky, United States

*Corresponding author: lijunyan323@gmail.com (Junyan Li)

Abstract

Machine learning trains and obtains learning models based on a large amount of training samples. Mislabeled training samples will affect the generalization/performance of the final predictive model. Some methods of detecting/correcting mislabeled samples such as graph-based methods, are proposed and used in machine learning to improve predictive models' generalization. However, these methods do not perform well for high-dimensional samples. In this paper, we present three algorithms for detecting/correcting mislabeled samples in high-dimensional feature space. First, we propose an improved high-dimensional detection algorithm: PCA-k-RNG. Next, we introduce a notion of ϵ-scalar relative neighbourhood graph (ϵ-SRNG) and study its relationship with relative neighbourhood graph (RNG) and k-relative neighbourhood graph (k-RNG). Then, we give an alternative high-dimensional detection algorithm: PCA-ϵ-SRNG. After detecting mislabeled training samples, it is necessary to correct these mislabeled samples. Then we further propose a scalar-adapted correction algorithm: Fat location correction/deletion. Finally, we explore and validate our algorithms based on real datasets with high-dimensional features.

Keywords: high dimension, inaccurate supervision learning, mislabeled samples, relative neighbourhood graph (RNG), detection, correction.

Introduction

Machine learning has made great success in every walk of life. Supervised learning, as one kind of framework of machine learning, needs a large number of labeled samples to train and construct the final predictive model. The bad quality (For examples, incomplete, inaccurate and so on) of labels will cause a bad generalization/performance of the obtained learning model. However, in practice, data labeling is a high time and resources consuming process, so it is very difficult to achieve all the ground-truth labels for a big training sample set. In [1], Zhou classified (weakly) supervised learning into three types: 1) incomplete supervision (Only a small parts of training samples are given with labels whereas the other remain unlabeled), 2) inexact supervision (The given labels are coarse-grained), and 3) inaccurate supervision (The given labels are not always ground-truth).

For inaccurate supervision, it means that some label information may suffer from errors [1]. A common situation is that labels of training samples are affected by random noise. And a noticeable and necessary step consists in cleansing/filtering the training samples themselves, what is similar to outlier or anomaly detection. There are many kinds of typical researches carried on handling (detecting/correcting) of mislabeled samples [2,3]: 1) voting filtering. For example, [4] adopted a set of algorithms to construct classifiers which play as a filter for training samples through the majority and consensus votes, 2) measures and thresholds. For example, [5] defined an information entropy on the probability of the instance belonging to each class label. Then it showed that a sample with entropy lower than a given threshold, but with error prediction result, is identified as mislabeled samples and replaced its original label with the predictive label, 3) graph-based methods. For examples, [6-8] regarded training samples as nodes in a graph. First, they created a relative neighbourhood graph (RNG) G by weighing the distance between each pair of nodes. The edges linking two nodes (i.e., training samples) with different labels are named cut edge. Then, they calculated a cut edge weight statistic, which is the sum of the edge weights of the detected training sample to its neighbors with different labels. Finally, according to the value of the cut edge weight statistic, they classified training samples into three types: good, doubtful and bad samples. The suspected and bad samples can be either removed or relabeled. In addition, there are other methods, such as 4) local (k-nearest neighbors)-based methods [9], 5) single model-based methods [10], 6) ensemble-based methods [11], and so on. However, detecting/correcting mislabeled samples in a training sample set is seldom easy. The aforementioned work achieves a certain degree of success.

In this paper, we focus on improving the quality of the training samples by detecting/handling mislabeled samples prior to apply learning algorithms, thereby increasing the predictive model's accuracy/generalization. And our study is for inaccurate supervision with a high-dimensional feature space. The main method adopted in this paper is based on graph theory.

Motivation. Note that the work of [6-8] mainly depend on the construction of RNG G. While the edges in G are based on the distance of vertices (training samples). Therefore, the neighborhood information will become less trustworthy in sparse high-dimensional feature space [1] (See distance matrix of samples from CNAE-9 in Table 5 in Appendix A). Meanwhile, for real datasets, the number of neighbors of some vertices in G could be small (See the experimental data about RNG in Tables 1 and 2 in Section 6). For such a situation, it will be inaccurate if we directly use the normal approximation of the cut edge weight statistic to calculate its p-value (See equation~(3.1) in Section 3). What's more, for handling detected doubtful samples, [6-8] directly dropped the doubtful sample when at least one of its neighbors has a different label. However, we think that such a detected doubtful sample may be a sample near/on the boundary.

Main results. To address these issues, we employ principal component analysis (PCA) before constructing neighbourhood graphs. Then we use k-RNG graph proposed in [12,13] to describe the neighbourhood of training samples, it can overcome the aforementioned shortfall of the small number of neighbors. Next, we propose an algorithm which can detect mislabeled samples in high dimension (See Algorithm 1 in Section 3). Then, we give an extended relative neighbourhood graph: ϵ-scalar relative neighbourhood graph (ϵ-SRNG). And we study the relationship among RNG, k-RNG and ϵ-SRNG. Further, based on ϵ-SRNG, we propose a PCA-ϵ-SRNG detection algorithm (See Algorithm 3 in Appendix B). In order to correct mislabeled samples, a fat location correction/deletion algorithm is given (See Algorithm 2 in Section 5). Finally, we perform and explore our three algorithms on datasets CNAE-9 and MNIST (See the experimental results in Tables 1 and 2 in Section 6). In addition, we employ z-test (confidence intervals) to make a decision to reject or fail to reject null hypothesis as pointed out in [14,15].

Outline. The remainder of this paper is organized as follows. In Section 2, we introduce some basic notations and definitions for later discussions. In Section 3, we review some mislabeled samples detection algorithms, then we present an improved mislabeled samples detection algorithm in high dimension: PCA-k-RNG. In Section 4, we give a notion of ϵ-scalar relative neighbourhood graph and analyze its relationship with RNG and k-RNG. Furthermore, we propose a PCA-ϵ-SRNG detection algorithm. In Section 5, we propose a scalar-adapted fat location correction/deletion algorithm for correcting/deleting mislabeled samples. In Section 6, we perform our proposed algorithms on two real datasets CNAE-9 and MNIST and discuss and study the results from experiments. We conclude the paper in Section 7.

Notations and definitions

In this section, we introduce some notations, assumptions, definitions and tools for later use. Note that among them, some are from [6-8,12,13,16-19] for self-containedness.

- Let $(\mathcal{Z}, \mathrm{P})$ be a probability space. P is the (unknown) distribution (or probability measure). $\mathcal{Z}$ alone is called the sample space, and $\mathcal{Z}$ has the structure $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$and $\mathcal{Y}$ are called the input and output spaces, respectively.
- Let $S = \{z_i = (x_i, y_i), z_i \in \mathcal{Z}, x_i \in \mathcal{X}, y_i \in \mathcal{Y}, i = 1, 2, \ldots, n, n \in \mathbb{N}\}$ be a finite set of labeled samples, and $y_i, i \in \mathbb{N}$ may be inaccurate/incorrect, and assume that these samples are independent and identically distributed (i.i.d.) according to P. Here, we suppose that $\mathcal{X} \subseteq \mathbb{R}^d$, $\mathcal{Y} \subseteq \mathbb{R}$, and $d \in \mathbb{N}$. For $x_1 \in \mathcal{X}$, we can write x_1 as $x_1^j, j = 1, 2, \ldots, d$ which are the features of the sample x_1.
- Let n_i be the number of neighbors of the sample vertice x_i.
- Let p_j be the probability of the label of the x_i 's neighbor x_j is the same to x_i 's label.
- Let w_{ij} be the weight between the sample vertice x_i and its neighbor x_j.
- Let w_{ij^N} be the weight between the sample vertice x_i and its neighbor x_j with a different label from x_i's label.
- For the simplicity, in this paper we consider the classification case, i.e., $\mathcal{Y} = \{1, 2, ..., r\}, r \in \mathbb{N}$.

Definition 2.1: [18] [Relative Neighbourhood Graph (RNG)] Let V be a set of points in $\mathbb{R}^d$ (with $d \in \mathbb{N}$ the number of features). The RNG of V is a graph with the vertices set V, and the set of edges of the RNG of V are exactly those pairs (x_i, x_j) of points for which

$$d(x_i, x_j) \leqslant \max\{d(x_i, x_h), d(x_j, x_h)\}, \forall h \neq i, j, i \neq j \tag{2.1}$$

where $d(\cdot,\cdot)$ denotes the distance between two points in $\mathbb{R}^d$.

Remark 2.2: Inequality~(2.1) in Definition 2.1 is equivalent to

$\{x | \max\{d(x_i, x), d(x_j, x)\} < d(x_i, x_j), x \neq x_i, x_j \text{ and } x_i \neq x_j\} = \emptyset$

Let

$$Lens(x_i, x_j) = \{x | \max\{d(x_i, x), d(x_j, x)\} < d(x_i, x_j), x \neq x_i, x_j \text{ and } x_i \neq x_j\} \tag{2.2}$$

where $Lens(x_i, x_j)$ is called a *lune* or *lens* [17,19]

Definition 2.3: [12,13,16] [k-Relative Neighbourhood Graph (k-RNG)] Let V be a set of points in $\mathbb{R}^d$ (with $d \in \mathbb{N}$ the number of features). The k-RNG of V is a graph with the vertices set V, and the set of edges of the k-RNG of V are exactly those pairs (x_i, x_j) of points for which

$$|Lens(x_i, x_j) \cap V| < k \tag{2.3}$$

When k=1, it is RNG which is first introduced by Toussaint in [18]. [12] extended the definition of relative neighbors used in RNG to define a general k-RNG based on the *lens* function~(2.2). [16] used k-RNG to study medoid estimation, outlier identification, classification and clustering.

For a given sample vertice x_i in the neighborhood graph, the total weights of its neighbors with different labels is

$$\hat{\mu}_i = \sum_{j=1}^{n_i} w_{ij^N} \tag{2.4}$$

the expectation of the total weights of its neighbors with different labels is

$$\mu_i = \sum_{j=1}^{n_i} w_{ij}(1 - p_j) \tag{2.5}$$

and the variance of the total weights of its neighbors with different labels is

$$\sigma_i^2 = \sum_{j=1}^{n_i} w_{ij}^2 (1 - p_j) p_j \tag{2.6}$$

In later discussions, we will construct neighbourhood graphs (i.e., k-RNG and ϵ-SRNG which will be introduced in Definition 4.1 in Section 4) from training samples, and detect mislabeled samples based on the constructed neighbourhood graphs.

Mislabeled samples detection in high dimension

In this section, we will improve the work in [6-8] and propose a mislabeled samples detection algorithm in high dimension feature space: PCA-ϵ-SRNG detection algorithm.

Consider that mislabeled samples perturb the generalization of learning models, [6-8] proposed a mislabeled samples identification and handling method based on graph theory. The method is based on the construction of RNG on training samples, and they computed a cut edge weight statistic, which is the sum of the edge weights of the detected training sample to its neighbors with different labels. Then they judged the detected training sample (Good, doubtful or bad) by calculating the p-value.

However, the method will become powerless when the feature space of samples is high-dimensional. We will illustrate the calculation of distance matrix on the CNAE-9 dataset (99.22% of this dataset is filled with zeros) from the UCI machine learning repository (See Table 5 in Appendix A). And note that the method in [6-8] used the normal approximation to compute the cut edge weight statistic when the number of neighbors of the detected training sample is “great enough” and the weights are not too unbalanced. However, for a RNG constructed from training samples in real datasets, the number of their neighbors may be small (See the experimental data about RNG in Tables 1 and 2 in Section 6). A general rule is that the number of samples n is “great enough” if

$$np \geqslant 5 \text{ and } n(1-p) \geqslant 5 \tag{3.1}$$

where p is the parameter in binomial distribution $B(n,p)$ [20].

To address these problems, in this paper, our method is as follows:

- We use PCA to reduce dimensions prior to construct neighbourhood graph.
- Then, we construct k-RNG from training samples. Here, we compare k-RNG with RNG in Subfigures (a)-(b) in Figure 1. The nodes in k-RNG have more neighbors than RNG,

and it will make the general rule about “great enough” more likely satisfied (It also is shown by the experimental data in Tables 1 and 2 in Section 6). Then we use the normal approximation to compute.

- Finally, we adopt z-test (Confidence intervals) to detect mislabeled samples.

We denote H_0 as the null hypothesis: $\hat{\mu}_i$ equals the hypothesized mean μ_i. In order to use z-test, we need the following statistics:

- Weight w_{ij}: we compute it as one over $d(x_i, x_j)$. If $d(x_i, x_j) = 0$, we let $w_{ij} = \infty$.
- Probability p_j : as in [8], we use the global proportion of the label of x_i as an estimation of p_j.
- Confidence interval:

$$\left[\mu_i - z_{1-\alpha/2}\frac{\sigma_i}{\sqrt{n_i}}, \mu_i + z_{1-\alpha/2}\frac{\sigma_i}{\sqrt{n_i}}\right]$$

where μ_i and σ_i are computed by equations~(2.5) and~(2.6). We need to calculate $\hat{\mu}_i$ by equation (2.4) and determine whether it belongs to the above interval.

Therefore, we have the following PCA-k-RNG detection algorithm (See Algorithm 1).

```
Algorithm 1: PCA-$k$-RNG Detection
   Input: the set of samples $S$, the threshold of the percentage of variance $\beta$, the significance level $\alpha$ and the neighbors $k$
   Output: an index set of $S$ in which samples are marked as mislabeled or not
1  construct the covariance matrix $M$ of $S$;
2  compute eigenvalues $\lambda_i$ and eigenvectors $v_i, i = 1, 2, \ldots, d$ of $M$;
3  pick the $k$ largest eigenvectors according to the percentage of variance
       $\min_{d'} \frac{\sum_{i=1}^{d'} \lambda_i}{\sum_{j=1}^{d} \lambda_j} \geqslant \beta$
4  let $V \in \mathbb{R}^{n\times d'}$ be the matrix containing vectors $v_i, i = 1, 2, \ldots, d'$ as columns;
5  for $i = 1, 2, \ldots, n$, let $x_i \in \mathbb{R}^{d'}$ be the vector corresponding to the $i$-th row of $V$;
6  compute the distance matrix $D$ of vectors $\{y_i\}_{i=1}^{d'}$;
7  construct $k$-relative neighbourhood graph $G$ of $D$;
8  for $i = 1$ to $n$ do
       • write the null hypothesis $H_0$ ($\hat{\mu}_i = \mu_i$)
         and alternative hypothesis $H_a$ ($\hat{\mu}_i \neq \mu_i$)
       • specify the level of confidence $1 - \alpha$
       • determine the standardized test statistic
             $z = \frac{\hat{\mu}_i - \mu_i}{\sigma_i/\sqrt{n_i}}$
       • determine the critical values $-z_{1-\alpha/2}$ and $z_{1-\alpha/2}$
       • determine the rejection regions
       • make a decision to reject $H_0$ or fail to reject $H_0$
9  end
10 return the detected results;
```

ϵ -scalar relative neighbourhood graph and PCA- ϵ -SRNG detection algorithm

In this section, we will proceed to improve the Algorithm 1 presented in Section 3.

From Subfigures (a)-(c) in Figure 1, it can see that the number of edges among sample vertices increases with k. What's more, the number of edges on sample vertices among different clusters also increases quickly with k. One behind reason is that k-RNG only requires the neighbors of sample vertices to be at most k while never considering how far a neighbor is. Meanwhile, such requirement will increase the calculation burden. Ideally, it should link sample vertices according to some similar scalar. Note the limitations of k-RNG, we extend RNG from the view of the metric, and give the following definition.

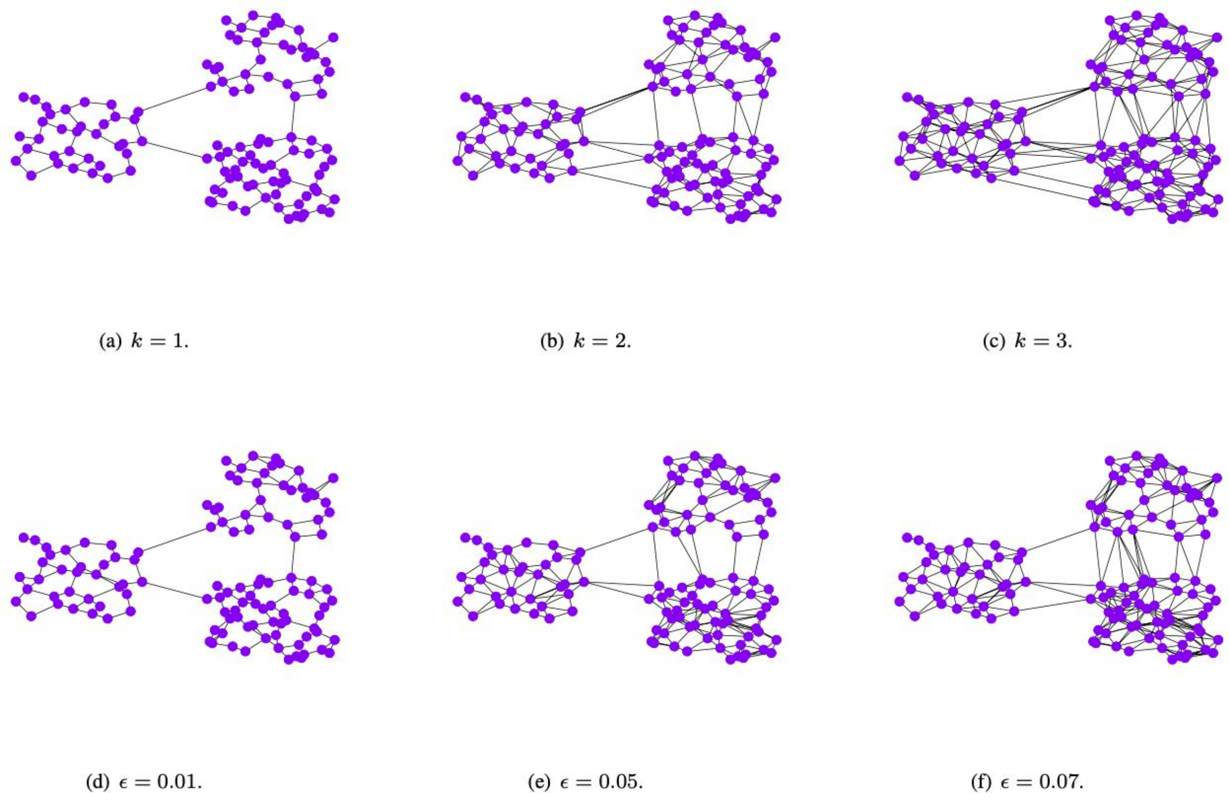


Figure 1. Subfigures (a)-(c) are about k-RNG. When k = 1, k-RNG is RNG. And Subfigures (d)-(f) are about ϵ-RNG.

Definition4.1[ϵ-Scalar Relative Neighbourhood Graph (ϵ-SRNG)])] Let V be a set of points in $\mathbb{R}^d$ (with $d \in \mathbb{N}$ the number of features). There exists an $\epsilon \geqslant 0$, such that, the ϵ-SRNG of V is a graph with vertices set V, and the set of edges of the ϵ-SRNG of V are exactly those pairs(x_i, x_j) of points for which

$$d(x_i, x_j) \leqslant \max\{d(x_i, x_h), d(x_j, x_h)\} + \epsilon, \forall h \neq i, j, i \neq j \tag{4.1}$$

where $d(\cdot,\cdot)$ denotes the distance between two points in $\mathbb{R}^d$.

Remark 4.2: Actually, ϵ-SRNG is scalar-dependent (vertice pair-dependent), while k-RNG is independent of the scalar, because k-RNG requires the *lune* of a pair of vertices to have at most k neighbors. In Figure 2, it compares the relationhip among RNG, k-RNG and ϵ-SRNG. Form Subfigures (a) and (b), it can see that k-RNG extends RNG by allowing the number of vertices in the *lune* is at most k. Form Subfigures (a) and (c), it can see that ϵ-SRNG does not impose certain conditions on the number of vertices, and just extends the boundary of RNG and shrinks the scope of the *lune* by a scalar ϵ. Thus, ϵ-SRNG is like a fat RNG. That is, the two circles in RNG become two donuts in ϵ-SRNG.

Property 4.3: If we take ϵ=0. Then, ϵ-SRNG is RNG.

Proof. The proof is trivial. Let ϵ=0, inequality (4.1) reduces to inequality (2.1).

In some cases, ϵ-SRNG is equivalent to k-RNG. Before giving the further properties, we introduce the following notation:

$$\begin{aligned} & Lens^{\epsilon}(x_i,x_j) \\ = \ & \{x|\max\{d(x_i,x),d(x_j,x)\}+\epsilon<d(x_i,x_j), x\neq x_i,x_j \text{ and } x_i\neq x_j\}\end{aligned}$$

Property 4.4: ϵ-SRNG is a special k-RNG, where

$$k=\max_{i,j}|\{Lens(x_i,x_j)-Lens^{\epsilon}(x_i,x_j)\}|.$$

Proof. From Subfigure (c) in Figure 2, $Lens^{\epsilon}(x_i,x_j)\}$ is the *lune* colored with green. While $Lens(x_i,x_j)$ is the *lune* bounded by the red dotted line. So, the proof follows.

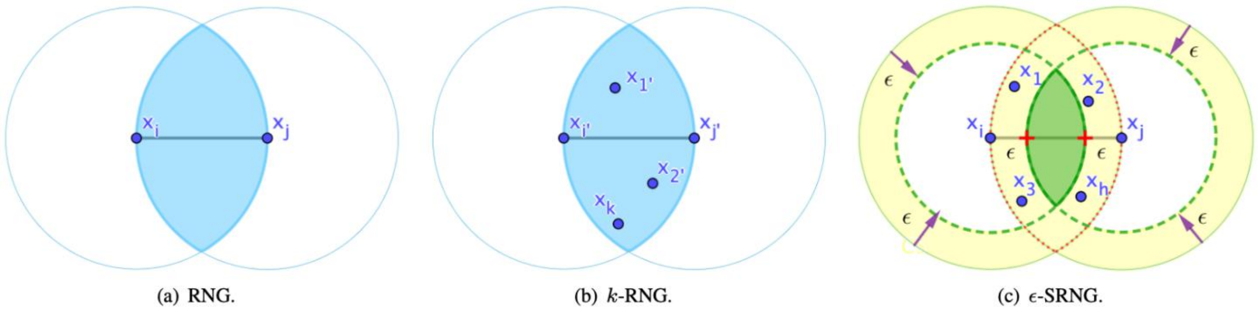


Figure 2. The comparison of RNG, k-RNG and ϵ-RNG.

Property 4.5: In inequality (4.1), if

$$\epsilon\geqslant d(x_i,x_j)/2$$

Then

$$Lens^{\epsilon}(x_i, x_j) = \emptyset$$

Proof. Note that the following set

$$\left\{\max\{d(x_i, x), d(x_j, x)\} < \frac{d(x_i, x_j)}{2}\right\}$$

is empty. Therefore, the result follows.

Property 4.6: If we take

$$\epsilon = \max_{ij}\{d(x_i, x_j)\}/2$$

Then, ϵ-SRNG and k-RNG are equivalent.

Proof. By the condition and Property 4.5, $\forall i \neq j$, we have

$$Lens^{\epsilon}(x_i, x_j) = \emptyset$$

Then by **Property 4.4**, let

$$k = \max_{i,j}\{Lens(x_i, x_j)\}$$

we get that ϵ-SRNG is k-RNG.

And use **Property 4.4** again, we have

$$\max_{i,j} |\{Lens(x_i, x_j)\}| = \max_{i,j} |\{Lens(x_i, x_j) - Lens^{\epsilon}(x_i, x_j)\}|$$

Thus, we obtain the result.

Therefore, ϵ-SRNG is a scalar-based description of neighborhood. From Figure 1, it shows that ϵ-SRNG has a more refined quantitative description on the neighbors than k-RNG: ϵ-SRNG describes the intermediate states from RNG to k-RNG with the changing of the scalar ϵ from 0 to $\max_{i,j}\{d(x_i, x_j)\}/2$

Meanwhile, for ϵ-SRNG, it actually links edges according to a priori scalar ϵ. Therefore, if we can choose an appropriate ϵ, there will be more edges in the same cluster, and less edges among different clusters (See Subfigure (e) vs. Subfigure (b)). And it will drop the distant sample vertice and spare the unnecessary computation.

Based on the definition of ϵ-SRNG, we propose a PCA-ϵ-SRNG detection algorithm. We mainly replace k-RNG in Step 7 of Algorithm 1 with ϵ-SRNG. For saving spaces, the concrete Algorithm 3 can be found in Appendix B. We will expect that Algorithm 3 achieves a similar detection than Algorithm 1 but spares the unnecessary computation.

Fat location correction/deletion algorithm

In the above section, we analyze how to detect mislabeled samples in high-dimensional feature space. However, after identifying mislabeled samples, it needs to correct them. In this section, we will discuss that how to correct the detected mislabel samples.

First, we classify the detected mislabeled samples into four types: let x_i be the detected mislabeled sample,

- Type I: x_i's label is the same to all its neighbors' labels.
- Type II: x_i's label is different from all its neighbors' labels, but its neighbors' labels are the same.
- Type III: x_i's label is different from all its neighbors' labels, and its neighbors' labels are not all the same.
- Type IV: x_i's label is the same to some of its neighbors' labels.

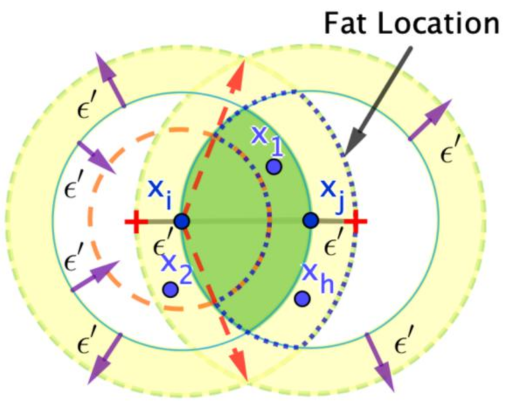


Figure 3. Fat location of x_j from the view of x_i in ϵ-SRNG.

For dealing with Types I-IV, we introduce a notion of "fat location". In Remark 4.2, we have discussed that ϵ-SRNG can be regarded as a fat RNG. In Figure 3, we name the sample vertices in the area bounded by the blue dotted line "fat location" of x_j from the view of x_i, and denote it as $fl(x_i \to x_j)$ or simply fl_{ij}. Actually, fl_{ij} can be expressed as

$$f_{ij} = \{x_k | w_{ik} \geqslant w_{ij} - \epsilon'\} \cap \{x_k | w_{ik} \leqslant w_{ij} + \epsilon'\} \cap \{x_k | w_{jk} \geqslant w_{ji}\}$$

Then, our handling method for the detected mislabeled samples is as follows:

- Type I: delete x_i.
- Type II: relabel x_i with its neighbors' label.
- Type III and IV: roughly speaking, if fat location of x_i from each of its neighbors with different labels is not empty, then we label each fat location according to the labels of its sample vertices. Next, we relabel x_i by majority voting on all labeled fat location. Otherwise, delete x_i.

Finally, we give the detailed correction algorithm in Algorithm 2.

```
Algorithm 2: Fat Location Correction/Deletion
  Input: the detected mislabeled sample $x_i$, $\epsilon$-SRNG, the scale $\epsilon'$ and the threshold of labeling fat location $\alpha$
  Output: the detected mislabeled sample is relabeled with a correcting label or deleted
if the labels of $x_i$'s neighbors are all the same with $x_i$'s label then
    delete $x_i$;
else
    if the labels of $x_i$'s neighbors are all the same then
        correct $x$'s label with the label of its neighbors;
    else if the number of neighbors with the same label to $x_i$'s are more than others then
        delete $x_i$;
    else
        fat_vector=vector();
        foreach $x_j$ in $x_i$'s neighbors do
            if $fl_{ij}$ is not empty then
                sort $fl_{ij}$ in ascending order according to the number of labels;
                if $fl_{ij}[1]$-$fl_{ij}[2] \geqslant \alpha$ then
                    label $fl_{ij}$ with $fl_{ij}[1]$;
                    add $fl_{ij}$ to fat_vector;
                end
            end
        end
        if fat_vector is not empty then
            label fat_vector by majority voting;
            correct $x$'s label with the label of fat_vector;
        else
            delete $x_i$;
        end
end
return the result;
```

Experiments and discussion

In this section, we will perform our three algorithms PCA-k-RNG, PCA-ϵ-SRNG and fat location correction/deletion algorithms on real datasets. Then, we will further analyze our experimental results.

We will explore our algorithms proposed in previous sections on datasets CNAE-9 [21] and MNIST [22], respectively. CNAE-9 is a dataset containing 1080 documents of free text business descriptions (The number of features is 856) of Brazilian companies categorized into a subset of 9 categories cataloged in a table called National Classification of Economic Activities [21]. This dataset is highly sparse (99.22% of the matrix is filled with zeros). MNIST has a training set of 60000 collected handwritten digits each digitized to a 28×28 grayscale (so with dimension 784, and 80.88% of the matrix is averagely filled with zeros) image, as well as a test set of 10000 examples [22]. If we try to directly construct neighborhood graphs from training samples in these datasets, the results are less reliable. For instance, we take the first 45 samples from CNAE-9 to construct the distance matrix and find that the distances of lots of sample pairs with different labels are the same (See Table 5 in Appendix A). Thus, our two algorithms (Algorithms 1 and 3) use PCA prior to construct neighbourhood graphs.

For simplicity of presentation, let

- AE=Average Edges
- $\text{IDR} = \dfrac{\text{Inaccurate (Pollution) Samples Detected}}{\text{Total (Pollution) Samples}}$
- $\text{DR} = \dfrac{\text{Deleted (Pollution) Samples}}{\text{Inaccurate (Pollution) Samples Detected}}$
- $\text{RR} = \dfrac{\text{Relabeled (Pollution) Samples}}{\text{Inaccurate (Pollution) Samples Detected}}$
- $\text{MCR} = \dfrac{\text{Corrected (Pollution) Samples}}{\text{Relabeled (Pollution) Samples}}$

Here, IDR, DR, RR and MCR are the abbreviations of inaccurate samples detection rate, deletion rate, relabeling rate and mislabeling correction rate on original/pollution training sample set, respectively.

In following experiments, we first introduce the noise level from 1% to 20% by mislabeling labels of training samples from CNAE-9 and MNIST, respectively. Then, we perform our algorithms (Algorithms 1, 2, and 3) on these two datasets. Tables 1 and 2 show the experimental results.

Algorithm \ Noise			1%		5%		7%		10%	
			Whole	Pollution	Whole	Pollution	Whole	Pollution	Whole	Pollution
PCA-RNG + Fat Location	AE = 2	IDR	0.005	0.000	0.008	0.050	0.008	0.036	0.013	0.075
		DR	0.000	NaN	0.000	0.000	0.000	0.000	0.000	0.000
		RR	1.000	NaN	1.000	1.000	1.000	1.000	1.000	1.000
		MCR	0.000	NaN	0.000	0.000	0.333	1.000	0.000	0.000
PCA-k-RNG + Fat Location	AE = 55 ($k = 28$)	IDR	0.045	0.750	0.070	0.650	0.070	0.500	0.100	0.625
		DR	0.000	0.000	0.071	0.077	0.000	0.000	0.000	0.000
		RR	1.000	1.000	0.929	0.923	1.000	1.000	1.000	1.000
		MCR	0.167	1.000	0.346	0.750	0.429	0.857	0.325	0.520
PCA-k-RNG + Fat Location	AE = 62 ($k = 32$)	IDR	0.045	1.000	0.073	0.700	0.078	0.643	0.110	0.725
		DR	0.000	0.000	0.035	0.071	0.032	0.000	0.000	0.000
		RR	1.000	1.000	0.966	0.929	0.968	1.000	1.000	1.000
		MCR	0.222	1.000	0.321	0.692	0.467	0.778	0.341	0.517
PCA-ϵ-SRNG + Fat Location	AE = 54 ($\epsilon = 12$)	IDR	0.048	1.000	0.075	0.450	0.080	0.643	0.110	0.575
		DR	0.158	0.000	0.167	0.111	0.094	0.000	0.114	0.000
		RR	0.842	1.000	0.833	0.889	0.906	1.000	0.886	1.000
		MCR	0.250	1.000	0.200	0.625	0.414	0.667	0.231	0.391
PCA-ϵ-SRNG + Fat Location	AE = 65 ($\epsilon = 13.1$)	IDR	0.043	0.750	0.075	0.600	0.078	0.571	0.110	0.525
		DR	0.177	0.000	0.167	0.167	0.097	0.000	0.091	0.000
		RR	0.824	1.000	0.833	0.833	0.903	1.000	0.909	1.000
		MCR	0.214	1.000	0.240	0.600	0.393	0.688	0.225	0.429

Algorithm \ Noise			15%		17%		20%		25%	
			Whole	Pollution	Whole	Pollution	Whole	Pollution	Whole	Pollution
PCA-RNG + Fat Location	AE = 2	IDR	0.015	0.050	0.013	0.059	0.013	0.050	0.005	0.000
		DR	0.000	0.000	0.000	0.000	0.000	0.000	0.000	NaN
		RR	1.000	1.000	1.000	1.000	1.000	1.000	1.000	NaN
		MCR	0.333	0.667	0.200	0.250	0.200	0.250	0.000	NaN
PCA-k-RNG + Fat Location	AE = 55 ($k = 28$)	IDR	0.145	0.567	0.140	0.632	0.160	0.638	0.193	0.610
		DR	0.035	0.029	0.000	0.000	0.000	0.000	0.026	0.033
		RR	0.966	0.971	1.000	1.000	1.000	1.000	0.974	0.967
		MCR	0.232	0.394	0.464	0.605	0.453	0.569	0.440	0.559
PCA-k-RNG + Fat Location	AE = 62 ($k = 32$)	IDR	0.165	0.633	0.160	0.691	0.180	0.700	0.203	0.630
		DR	0.046	0.000	0.016	0.000	0.014	0.000	0.037	0.032
		RR	0.955	1.000	0.984	1.000	0.986	1.000	0.963	0.968
		MCR	0.254	0.421	0.460	0.617	0.465	0.589	0.385	0.492
PCA-ϵ-SRNG + Fat Location	AE = 54 ($\epsilon = 12$)	IDR	0.163	0.567	0.135	0.544	0.155	0.538	0.188	0.550
		DR	0.077	0.029	0.037	0.000	0.048	0.000	0.067	0.018
		RR	0.923	0.971	0.963	1.000	0.952	1.000	0.933	0.982
		MCR	0.217	0.394	0.462	0.649	0.441	0.605	0.343	0.444
PCA-ϵ-SRNG + Fat Location	AE = 65 ($\epsilon = 13.1$)	IDR	0.158	0.500	0.138	0.574	0.153	0.550	0.185	0.520
		DR	0.079	0.000	0.018	0.000	0.033	0.000	0.108	0.039
		RR	0.921	1.000	0.982	1.000	0.967	1.000	0.892	0.962
		MCR	0.190	0.367	0.500	0.692	0.475	0.636	0.349	0.460

Table 1. Algorithms on CNAE-9 (The number of eigenvector chosen is 180, ratio of variance is about 0.974, number of sample is 400, significance level is 0.1, scale $\epsilon' = 5$ and threshold of labeling fat location $\alpha = 25$)

Algorithm		Noise	1%		5%		7%		10%	
			Whole	Pollution	Whole	Pollution	Whole	Pollution	Whole	Pollution
PCA-RNG + Fat Location	AE = 4	IDR	0.008	0.250	0.018	0.200	0.010	0.071	0.030	0.225
		DR	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
		RR	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
		MCR	0.333	0.250	0.571	1.000	0.500	1.000	0.750	1.000
PCA-k-RNG + Fat Location	AE = 59 ($k = 23$)	IDR	0.038	0.500	0.073	0.750	0.085	0.750	0.093	0.625
		DR	0.133	0.000	0.103	0.000	0.088	0.000	0.054	0.000
		RR	0.867	1.000	0.897	1.000	0.912	1.000	0.946	1.000
		MCR	0.154	1.000	0.539	0.933	0.516	0.762	0.629	0.880
PCA-k-RNG + Fat Location	AE = 65 ($k = 26$)	IDR	0.043	0.500	0.080	0.800	0.088	0.714	0.100	0.675
		DR	0.177	0.000	0.125	0.000	0.114	0.000	0.075	0.000
		RR	0.824	1.000	0.875	1.000	0.886	1.000	0.925	1.000
		MCR	0.143	1.000	0.536	0.938	0.484	0.750	0.649	0.889
PCA-ϵ-SRNG + Fat Location	AE = 58 ($\epsilon = 4.8$)	IDR	0.058	0.750	0.098	0.800	0.093	0.679	0.108	0.625
		DR	0.348	0.000	0.205	0.000	0.108	0.000	0.163	0.000
		RR	0.652	1.000	0.795	1.000	0.892	1.000	0.837	1.000
		MCR	0.200	1.000	0.484	0.938	0.424	0.737	0.639	0.920
PCA-ϵ-SRNG + Fat Location	AE = 64 ($\epsilon = 5$)	IDR	0.053	0.750	0.085	0.700	0.088	0.679	0.098	0.550
		DR	0.333	0.000	0.206	0.000	0.143	0.000	0.180	0.000
		RR	0.667	1.000	0.794	1.000	0.857	1.000	0.821	1.000
		MCR	0.214	1.000	0.482	0.929	0.467	0.737	0.625	0.909

Algorithm		Noise	15%		17%		20%		25%	
			Whole	Pollution	Whole	Pollution	Whole	Pollution	Whole	Pollution
PCA-RNG + Fat Location	AE = 4	IDR	0.023	0.117	0.025	0.118	0.033	0.138	0.038	0.090
		DR	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
		RR	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
		MCR	0.556	0.714	0.600	0.750	0.615	0.727	0.200	0.333
PCA-k-RNG + Fat Location	AE = 59 ($k = 23$)	IDR	0.148	0.717	0.130	0.632	0.153	0.663	0.200	0.600
		DR	0.051	0.023	0.077	0.047	0.066	0.038	0.050	0.000
		RR	0.949	0.977	0.923	0.954	0.934	0.962	0.950	1.000
		MCR	0.518	0.691	0.646	0.756	0.649	0.726	0.540	0.683
PCA-k-RNG + Fat Location	AE = 65 ($k = 26$)	IDR	0.148	0.683	0.130	0.632	0.153	0.675	0.198	0.580
		DR	0.068	0.024	0.077	0.047	0.066	0.037	0.051	0.000
		RR	0.932	0.976	0.923	0.954	0.934	0.963	0.949	1.000
		MCR	0.509	0.700	0.646	0.756	0.667	0.731	0.547	0.707
PCA-ϵ-SRNG + Fat Location	AE = 58 ($\epsilon = 4.8$)	IDR	0.140	0.633	0.143	0.632	0.168	0.638	0.193	0.510
		DR	0.089	0.026	0.123	0.070	0.105	0.059	0.052	0.000
		RR	0.911	0.974	0.877	0.930	0.896	0.941	0.948	1.000
		MCR	0.510	0.703	0.620	0.775	0.583	0.729	0.480	0.686
PCA-ϵ-SRNG + Fat Location	AE = 64 ($\epsilon = 5$)	IDR	0.138	0.617	0.140	0.603	0.158	0.588	0.190	0.500
		DR	0.109	0.027	0.143	0.073	0.127	0.064	0.053	0.000
		RR	0.891	0.973	0.857	0.927	0.873	0.936	0.947	1.000
		MCR	0.531	0.722	0.604	0.763	0.564	0.705	0.486	0.700

Table 2. Algorithms on MNIST (The number of eigenvector chosen is 162, ratio of variance is about 0.974, number of sample is 400, significance level is 0.05, scale $\epsilon' = 5$ and threshold of labeling fat location $\alpha = 20$)

From Tables 1 and 2, we plot the curve of the inaccurate samples detected rate in Figure 4. And it can see that except the case $k = 1$, i.e., RNG, the trend of the detected rates generally increases with noise level. However, the curve of detection rates will begin to be lower than the curve of "Detection rate = Noise level" as the noise level is more than about 10%, that is, when our algorithms could not detect all the mislabeled samples. And for some abnormal points such as the values on the noise level 15% in Figure 4, we think that one behind reason is some samples polluted are on the boundary, so they are detected as inaccurate samples.

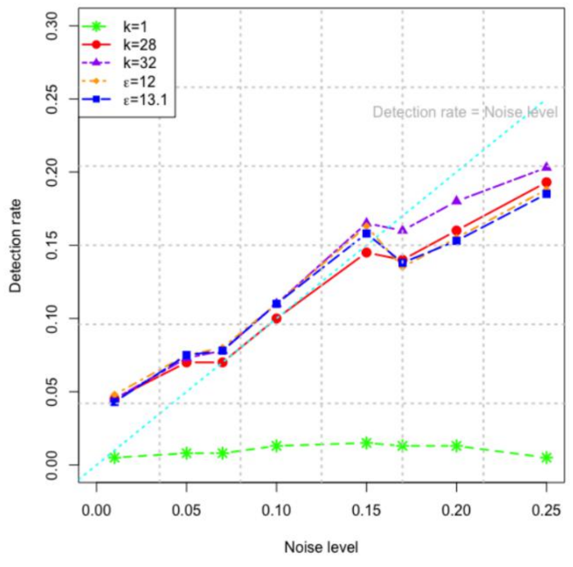
Parameter	Noise	Detected pollution samples over total samples							
		1%	5%	7%	10%	15%	17%	20%	25%
CNAE-9	$k=1$	0.000	0.003	0.003	0.008	0.008	0.010	0.010	0.000
	$k=28$	0.008	0.033	0.035	0.063	0.085	0.107	0.128	0.153
	$k=32$	0.010	0.035	0.045	0.073	0.095	0.117	0.140	0.158
	$\epsilon=12$	0.010	0.023	0.045	0.058	0.085	0.092	0.108	0.138
	$\epsilon=13.1$	0.008	0.030	0.040	0.053	0.075	0.098	0.110	0.130
MNIST	$k=1$	0.003	0.010	0.005	0.023	0.018	0.020	0.028	0.023
	$k=23$	0.005	0.038	0.053	0.063	0.108	0.107	0.133	0.150
	$k=26$	0.005	0.040	0.050	0.068	0.102	0.107	0.135	0.145
	$\epsilon=4.8$	0.008	0.040	0.048	0.063	0.095	0.107	0.128	0.128
	$\epsilon=5$	0.008	0.035	0.048	0.055	0.093	0.103	0.118	0.125

Table 3. The detected pollution samples over total samples for CNAE-9 and MNIST

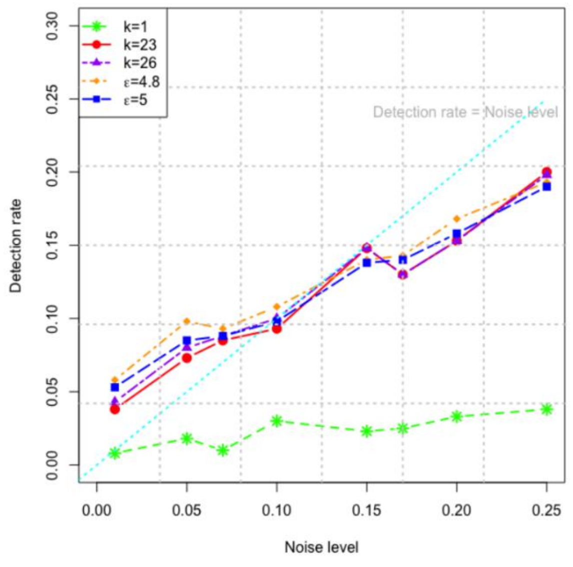
And from Tables 1 and 2, we can compute the detected pollution samples over total samples as follow (The calculation results see Table 3):

$$\frac{\text{Pollution Samples Detected}}{\text{Total Samples}} = \text{IDR on Pollution Samples} \times \text{Noise Level}$$

Then we plot the curves of the detected pollution samples over total samples with the change of noise in Figure 5. It can see that when the noise level is between 1% and 5%, our algorithms can detect nearly all the mislabeled samples we have made. With increasing noise level, our algorithms become less effective. The detection rate is about 15% as the noise level is around 25%, when PCA-k-RNG is slightly better than PCA-ϵ-SRNG. In addition, contrasting with Figure 4, the curves in Figure 5 have a larger deviation from the curve "Detection rate = Noise level". It means that there exist some inaccurate samples detected by our algorithms before we add noise to training samples.



(a) CNAE-9.



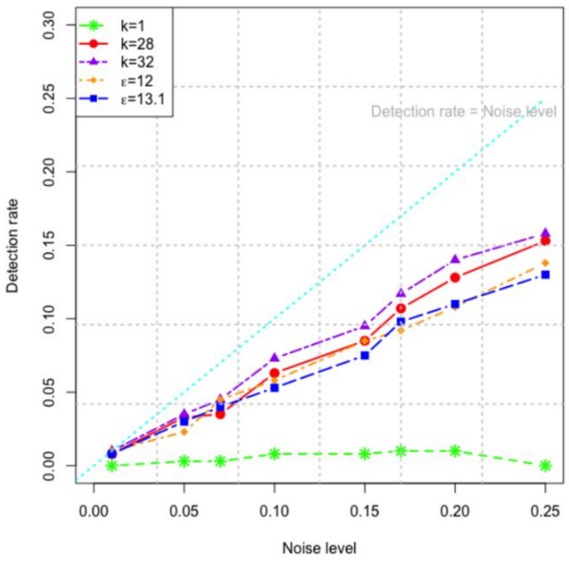
(b) MNIST.

Figure 4. The curves of inaccurate samples detection rate with the change of noise.

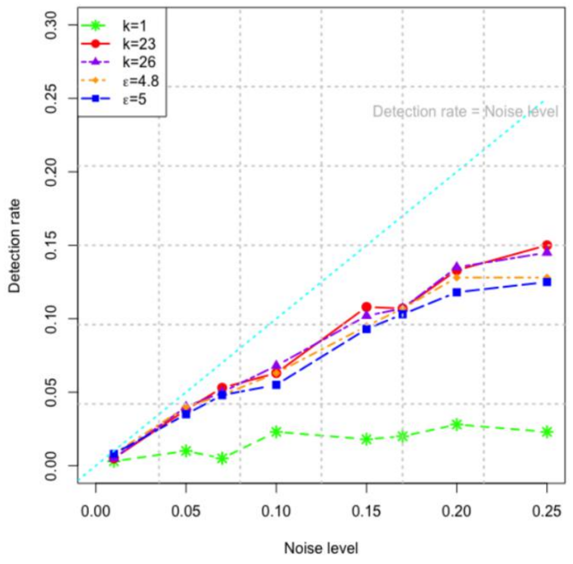
Furthermore, from the definitions of IDR, RR and MCD, we can calculate the correction rate on pollution samples as follow:

$$\begin{aligned} & \text{The Correction Rate on Pollution Samples} \\ = & \frac{\text{Corrected Pollution Samples}}{\text{Total Pollution Samples}} \\ = & \text{IDR} \times \text{RR} \times \text{MCD} \end{aligned}$$

Then, based on the experimental data in Tables 1 and 2, we compute the correction rate on pollution samples in Table 4. And we plot the curves of the correction rate on pollution samples with the change of noise in Figure 6. It shows that the curves of correction rate on pollution samples generally decrease with increasing noise level. The trend of correction rates for k-RNG and ϵ-SRNG-based algorithms are close. While for RNG case, i.e., $k = 1$, the trend is not so obvious which is due to its low correction rate. And for some abnormal points such as the values on the noise level 15% in Subfigure (a) and 1% in Subfigure (b) in Figure 6, we think one behind reason is too much samples are detected as mislabeled (some may be accurate samples), then the correction rate becomes low (See the values on the noise level 15% in Subfigure (a) in Figure 4). Another reason may be some polluted samples are on the boundary, so it is difficult to correct (or detect) when they are polluted/mislabeled (See the purple and red values on the noise level 1% in Subfigure (b) in Figure 4).

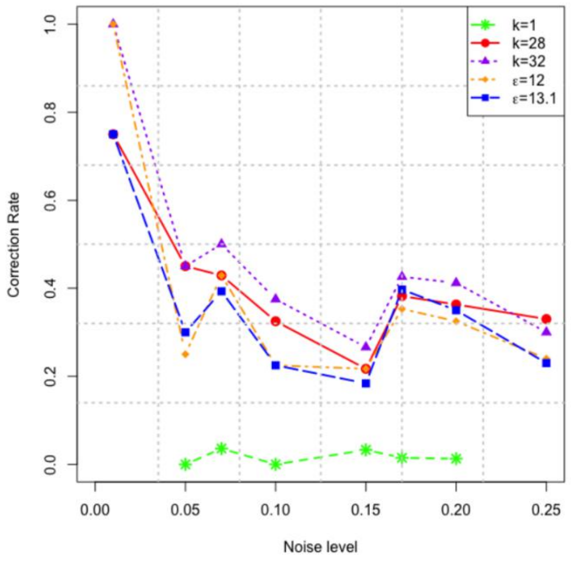


(a) CNAE-9.

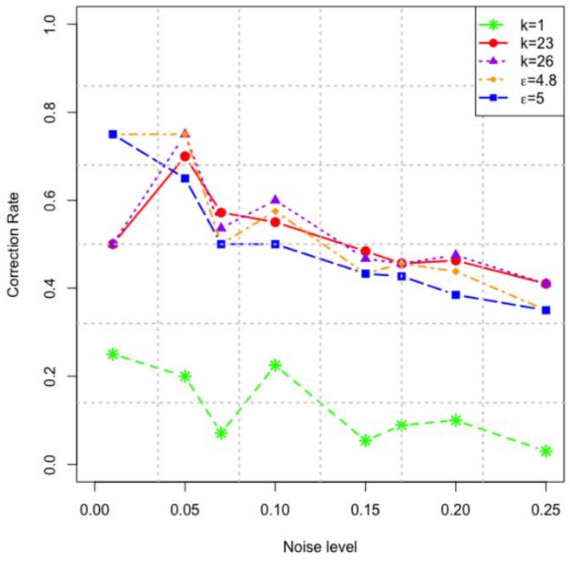


(b) MNIST.

Figure 5. The curves of detected pollution samples over total samples with the change of noise.



(a) CNAE-9.



(b) MNIST.

Figure 6. The curves of correction rate on pollution samples with the change of noise.

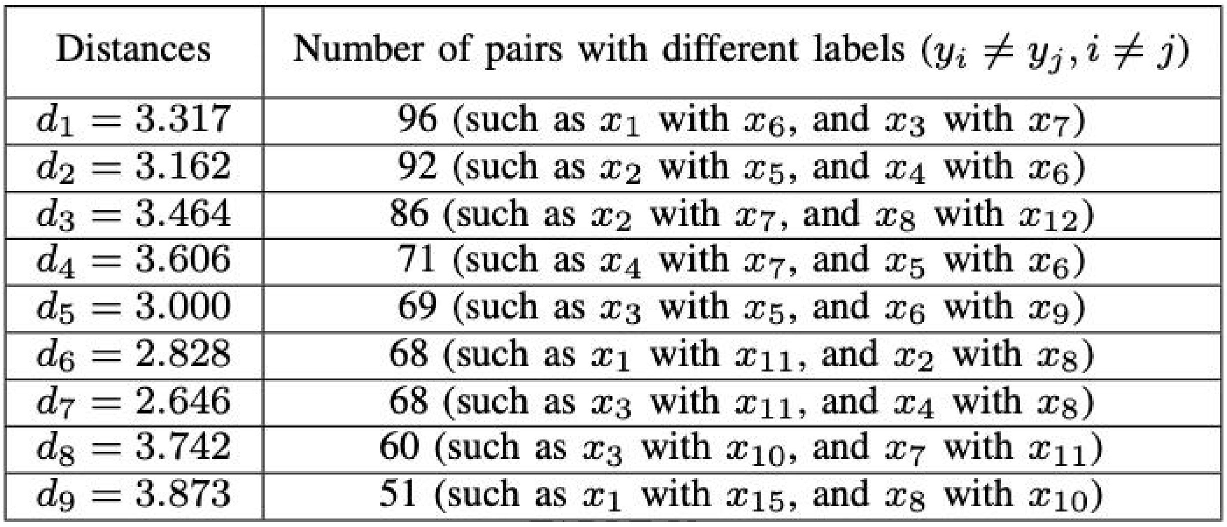
	Noise / Parameter	The correction rate on pollution samples							
		1%	5%	7%	10%	15%	17%	20%	25%
CNAE-9	$k = 1$	NaN	0.000	0.036	0.000	0.033	0.015	0.013	NaN
	$k = 28$	0.750	0.450	0.429	0.325	0.217	0.382	0.363	0.330
	$k = 32$	1.000	0.450	0.500	0.375	0.266	0.426	0.412	0.300
	$\epsilon = 12$	1.000	0.250	0.429	0.225	0.217	0.353	0.325	0.240
	$\epsilon = 13.1$	0.750	0.300	0.393	0.225	0.184	0.397	0.350	0.230
MNIST	$k = 1$	0.250	0.200	0.071	0.225	0.054	0.089	0.100	0.030
	$k = 23$	0.500	0.700	0.572	0.550	0.484	0.456	0.463	0.410
	$k = 26$	0.500	0.750	0.536	0.600	0.467	0.456	0.475	0.410
	$\epsilon = 4.8$	0.750	0.750	0.500	0.575	0.433	0.456	0.438	0.350
	$\epsilon = 5$	0.750	0.650	0.500	0.500	0.433	0.427	0.385	0.350

Table 4. The correction rate on pollution samples for CNAE-9 and MNIST

Conclusion

In this paper, we focus on the detection and correction of mislabeled samples in high dimension. First, based on the k-RNG constructed from training samples, we give a PCA-k-RNG detection algorithm for the high-dimensional feature space. Then, we propose an ϵ-scalar relative neighbourhood graph from the view of the metric and present an alternative high-dimensional detection algorithm: PCA-ϵ-SRNG. And in order to correct the detected mislabeled samples, we propose a fat location correction/deletion algorithm. Finally, we perform and analyze our algorithms on two real datasets. How to adjust the scale and threshold parameters (For instances, ϵ in Algorithm 1, α in Algorithm 2 and k in Algorithm 3. From figures in Section 6, it seems that the smaller ϵ is or the larger k is, the better the detection and correction rates are) and improve algorithms' performance deserves further studying in the future.

Appendix A Distance matrix for samples from CNAE-9



Distances	Number of pairs with different labels ($y_i \neq y_j, i \neq j$)
$d_1 = 3.317$	96 (such as x_1 with x_6, and x_3 with x_7)
$d_2 = 3.162$	92 (such as x_2 with x_5, and x_4 with x_6)
$d_3 = 3.464$	86 (such as x_2 with x_7, and x_8 with x_{12})
$d_4 = 3.606$	71 (such as x_4 with x_7, and x_5 with x_6)
$d_5 = 3.000$	69 (such as x_3 with x_5, and x_6 with x_9)
$d_6 = 2.828$	68 (such as x_1 with x_{11}, and x_2 with x_8)
$d_7 = 2.646$	68 (such as x_3 with x_{11}, and x_4 with x_8)
$d_8 = 3.742$	60 (such as x_3 with x_{10}, and x_7 with x_{11})
$d_9 = 3.873$	51 (such as x_1 with x_{15}, and x_8 with x_{10})

Table 4. Distance matrix for 45 samples from CNAE-9

Appendix B PCA-ϵ-SRNG Detection

```
Algorithm 3: PCA-$\epsilon$-SRNG Detection
  Input: the set of samples $S$, the threshold of the percentage of variance $\beta$, the significance level $\alpha$ and the scale $\epsilon$
  Output: an index set of $S$ in which samples are mislabeled
1 construct the covariance matrix $M$ of $S$;
2 compute eigenvalues $\lambda_i$ and eigenvectors $v_i, i = 1, 2, \ldots, d$ of $M$;
3 pick the $k$ largest eigenvectors according to the percentage of variance
        $\min_{d'} \frac{\sum_{i=1}^{d'} \lambda_i}{\sum_{j=1}^{d} \lambda_j} \geqslant \beta$
4 let $V \in \mathbb{R}^{n \times d'}$ be the matrix containing vectors $v_i, i = 1, 2, \ldots, d'$ as columns;
5 for $i = 1, 2, \ldots, n$, let $x_i \in \mathbb{R}^{d'}$ be the vector corresponding to the $i$-th row of $V$;
6 compute the distance matrix $D$ of vectors $\{y_i\}_{i=1}^{d'}$;
7 construct $\epsilon$-scalar relative neighbourhood graph $G$ of $D$;
8 for $i = 1$ to $n$ do
      • write the null hypothesis $H_0$ ($\hat{\mu}_i = \mu_i$)
        and alternative hypothesis $H_a$ ($\hat{\mu}_i \neq \mu_i$)
      • specify the level of confidence $1 - \alpha$
      • determine the standardized test statistic
            $z = \frac{\hat{\mu}_i - \mu_i}{\sigma_i / \sqrt{n_i}}$
      • determine the critical values $-z_{1-\alpha/2}$ and $z_{1-\alpha/2}$
      • determine the rejection regions
      • make a decision to reject $H_0$ or fail to reject $H_0$
9 end
10 return the detected results;
```

REFERENCES

[1] Z.-H. Zhou, “A brief introduction to weakly supervised learning,” *National Science Review*, vol. 5, no. 1, pp. 44– 53, 2018.

[2] B.Fre nayandM.Verleysen,“Classificationinthepres- ence of label noise: a survey,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 5, pp. 845–869, 2014.

[3] M. Poel, “Detecting mislabeled data using supervised machine learning techniques,” in *Proceedings of the 11th International Conference on Augmented Cognition (AC)*, Vancouver, Canada, Jul. 2017, pp. 571–581.

[4] C. E. Brodley and M. A. Friedl, “Identifying mislabeled training data,” *Journal of Artificial Intelligence Research*, vol. 11, no. 1, pp. 131–167, 1999.

[5] J. Sun, F. Zhao, C. Wang, and S. Chen, “Identifying and correcting mislabeled training instances,” in *Proceedings of Future Generation Communication and Networking (FGCN)*, Jeju, South Korea, Dec. 2007, pp. 244–250.

[6] S. Lallich, F. Muhlenbach, and D. A. Zighed, “Improv- ing classification by removing or relabeling mislabeled instances,” in *Proceedings of the 13th International Sym- posium on Foundations of Intelligent Systems (ISMIS)*, Lyon, France, Jun. 2002, pp. 5–15.

[7] D. A. Zighed, S. Lallich, and F. Muhlenbach, “Separabil- ity index in supervised learning,” in *Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD)*, Helsinki, Finland, Aug. 2002, pp. 475–487.

[8] F. Muhlenbach, S. Lallich, and D. A. Zighed, “Iden- tifying and handling mislabelled instances,” *Journal of Intelligent Information Systems*, vol. 22, no. 1, pp. 89– 109, 2004.

[9] D. W. Aha, D. Kibler, and M. K. Albert, “Instance-based learning algorithms,” *Machine Learning*, vol. 6, no. 1, pp. 37–66, 1991.

[10] R. Ekambaram, S. Fefilatyev, M. Shreve, K. Kramer, L. O. Hall, D. B. Goldgof, and R. Kasturi, “Active cleaning of label noise,” *Pattern Recognition*, vol. 51, no. 3, pp. 463–480, 2016.

[11] J. Cao, S. Kwong, and R. Wang, “A noise-detection based adaboost algorithm for mislabeled data,” *Pattern Recognition*, vol. 45, no. 12, pp. 4451–4465, 2012.

[12] M. S. Chang, C. Y. Tang, and R. C. T. Lee, “20-relative neighborhood graphs are hamiltonian,” *Journal of Graph Theory*, vol. 15, no. 5, pp. 543–557, 1991.

[13] ——, “Solving the euclidean bottleneck matching prob- lem by k-relative neighborhood graphs,” *Algorithmica*, vol. 8, no. 1-6, pp. 177–194, 1992.

[14] J. Ranstam, “Why the p-value culture is bad and confidence intervals a better alternative,” *Osteoarthritis and Cartilage*, vol. 20, no. 8, pp. 805–808, 2012.

[15] R. L. Wasserstein and N. A. Lazar, “The ASA’s state- ment on p-values: Context, process, and purpose,” *The American Statistician*, vol. 70, no. 2, pp. 129–133, 2016.

[16] M. Kleindessner and U. V. Luxburg, “Lens depth func- tion and k-relative neighborhood graph: Versatile tools for ordinal data analysis,” *Journal of Machine Learning Research*, vol. 18, no. 1, pp. 1–52, 2017.

[17] G. T. Toussaint, “Applications of the relative neigh- bourhood graph,” International Journal of Advances in Computer Science and Its Applications, vol. 4, no. 3, pp. 77–85, 2014.

[18] ——, “The relative neighborhood graph of a finite planar set,” *Pattern Recognition*, vol. 12, no. 4, pp. 261–268, 1980.

[19] Z. Liu and R. Modarres, “Lens data depth and median,” *Journal of Nonparametric Statistics*, vol. 23, no. 4, pp. 1063–1074, 2011.

[20] R. V. Hogg, E. Tanis, and D. Zimmerman, Probability and Statistical Inference, 9th ed. Pearson Education, Inc., 2014.

[21] D. Dua and E. K. Taniskidou. “UCI Machine Learning Repository”. Retrieved 5 Jan. 2019. [Online]. Available: http://archive.ics.uci.edu/ml

[22] Y. LeCun and C. Cortes and C. J. C. Burges. “The MNIST database of handwritten digits”. Retrieved 5 Jan. 2019. [Online]. Available: http://yann.lecun.com/ exdb/mnist/