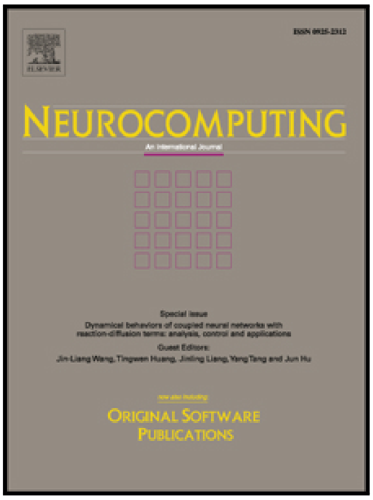
Robust deep auto-encoding Gaussian process regression for unsupervised anomaly detection Communicated by Dr Zhanyu Ma

Journal Pre-proof


NEUROCOMPUTING
An International Journal
Special issue
Dynamical behaviors of coupled neural networks with reaction-diffusion terms: analysis, control and applications
Guest Editors:
Jin-Liang Wang, Tingwen Huang, Jinling Liang, Yang Tang and Jun Hu
ORIGINAL SOFTWARE PUBLICATIONS

Robust deep auto-encoding Gaussian process regression for unsupervised anomaly detection

Jinan Fan, Qianru Zhang, Jialei Zhu, Meng Zhang, Zhou Yang, Hanxiang Cao

PII: S0925-2312(19)31346-3
DOI: https://doi.org/10.1016/j.neucom.2019.09.078
Reference: NEUCOM 21333

To appear in: *Neurocomputing*

Received date: 31 January 2019
Revised date: 12 August 2019
Accepted date: 23 September 2019

Please cite this article as: Jinan Fan, Qianru Zhang, Jialei Zhu, Meng Zhang, Zhou Yang, Hanxiang Cao, Robust deep auto-encoding Gaussian process regression for unsupervised anomaly detection, *Neurocomputing* (2019), doi: https://doi.org/10.1016/j.neucom.2019.09.078

Robust deep auto-encoding Gaussian process regression for unsupervised anomaly detection

Jinan Fan[a], Qianru Zhang[a], Jialei Zhu[a], Meng Zhang[a,*], Zhou Yang[a], Hanxiang Cao[a]

[a]*National ASIC System Engineering Technology Research Center, Southeast University, Nanjing, China*

Abstract

Unsupervised anomaly detection (AD) is of great importance in both fundamental machine learning researches and industrial applications. Previous approaches have achieved great advance in improving the performance of unsupervised AD model recently. However, there are still some thorny issues unsolved, especially the problem of efficiency degradation when dealing with high-dimensional data and the inability to maintain robustness when dealing with contaminated data, which have not been addressed simultaneously in the existing models. In our work, we propose a novel hybrid unsupervised AD method, which first integrates convolutional auto-encoder and Gaussian process regression to extract features and to remove anomalies from noisy data as well. Our model behaves more effectively at modeling high-dimension data and more robust to variation of the anomaly rate in dataset. We evaluate its performance on four publicly benchmark datasets and show the state-of-the-art performance against competitive methods.

Keywords: anomaly detection, deep auto-encoder, Gaussian process, high-dimension data, data contamination

1. Introduction

Anomaly detection is a fundamental task in machine learning [1]. According to Hawkins, an anomaly is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism [2]. Anomaly detection is defined as an analysis task to detect data whose patterns deviate form normal data [3]. Typically, the task of AD algorithms is to build a model by learning data and those data far away from this framed model are regarded as anomaly. Various kinds of approaches, such as neighbor based models, statistical based models and deep based models, can be used for anomaly detection. In many practical application domains including fraud detection [4], intrusion detection [5], medical diagnosis [6] and many others, the unsupervised AD methods are particularly suited when no label information is available [7]. Although fruitful progresses have been made in the last several years, conducting robust anomaly detection on multi- or high-dimensional data without human supervision remains a challenging task [8]. So effective ways are required to detect anomalies in large quantities of high-dimension data with great robustness to different levels of data contamination. In our work, we focus on locating the anomaly observations in the noisy data which includes both normal data and abnormal data in an unsupervised way.

It is more difficult to detect anomalies as the dimensionality of data grows higher since any input sample could be a rare event with low probability to be observed [9]. Solutions widely adopted to the issue are to reduce the dimensionality of input and then to detect anomalies in the corresponding low dimension space.There are many ways for feature reduction, such as Principal Component Analysis(PCA), kernel PCA, auto-encoder(AE), Non-negtive Matrix Factorization(NMF), Genetic programming, Deep Believe Network(DBN), and other demension reduction methods which can be seen in [10]. Among them, PCA [11] and AE [12] are the two commonly used methods. PCA based dimension reduction methods retains the data information with larger eigenvalues and discards the data information with smaller eigenvalues. AE is a popular deep learning-based data dimensionality reduction method. Although there is more computation cost by AE based dimension reduction methods, it have been demonstrated to be very successful in doing feature reduction in anomaly detection, such as [3, 13, 14]. Data contamination [15] is a problem that cannot be resolved

[*] Corresponding author
Email address: zmeng@seu.edu.cn (Meng Zhang)

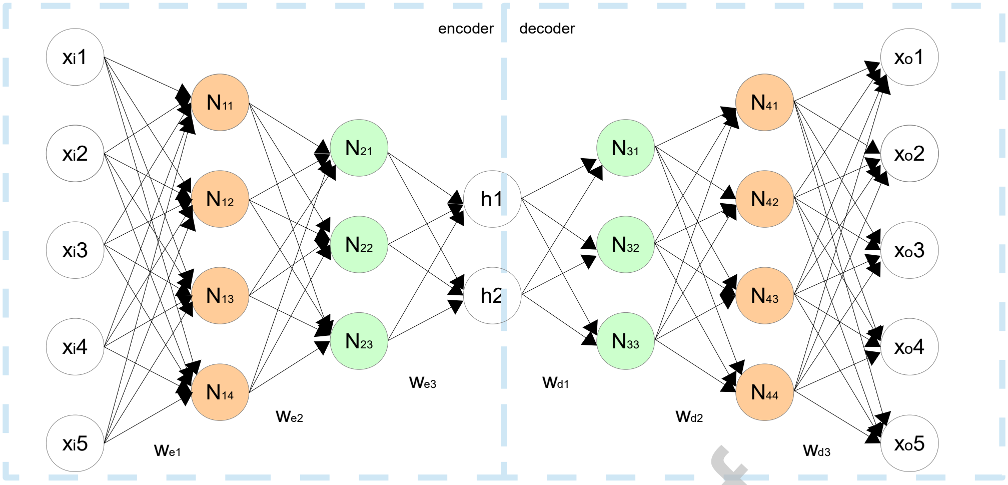


Figure 1: Illustration of auto-encoder.

by dimension reduction methods. Unfortunately, the performance of classicial unsupervised AD models degrades quickly as the number of anomalies grows in data. Because noise (anomaly data) distributes differently from the majority of the target (normal data), which challenges AD model and results in worse performance. Some existing new methods [16, 17] reduce the influence of data contamination. Nevertheless, these models only improve little performance and cannot escape from the curse. We address the problem by introducing Gaussian process regression (GPR) model in an iterative way since GPR can give a reliable estimate of regression problems with a small amount of learning samples [18]. We utilize this characteristic in an iterative way to learn information from normal and anomaly data which are labeled by the model in the last training iteration and to mitigate the influence of varying anomaly rate. Anomaly rate is defined as the ground truth ratio between the number of anomaly data to the number of whole dataset which includes both normal and anomaly data, which is calculated by $\boldsymbol{Num_N}/(\boldsymbol{Num_N}+\boldsymbol{Num_Y})$,where $\boldsymbol{Num_N}$, $\boldsymbol{Num_Y}$ represent the number of anomaly data and normal data of the whole dataset, respectively. In this work, we present a Deep auto-encoding Gaussian process regression (DAGPR) model for the unsupervised anomaly detection. Our approach possesses the following merits:

- It is robust to anomaly rate (the proportion of outliers in the entire noisy dataset) ranging from 0.1 to 0.5 and achieves state-of-the-art performance.
- It is feasible for high-dimension input data.
- It converges within around 20 iterations on the verification datasets.

2. Related work

Unsupervised anomaly detection is a challenging problem in many practical application domains when no labeling information is available [7]. Tremendous efforts have been devoted to unsupervised anomaly detection [16], and the existing methods can be grouped into three categories: neighbor based, statistical based and deep based.

2.1. Neighbor based

Neighbor based algorithm is a very classical approach to detect abnormality as it distinguishes anomaly data from the normal data using spatial information such as distance and density. Distance measure can be used for anomaly detection that data point with large distance would be defined as anomaly data. Recent progress has been made on distance based AD algorithms including methods proposed by Aytekin et al. [19], Zhang et al. [20], etc. Besides distance, density is another measurement metrics. For example, Local Outlier Factor(LOF) [21] computes the local density deviation of a given data point with respect to its neighbors and the samples that have a substantially lower density than their neighbors are considered as outliers. Isolation forest (IF) [22] is an effective density based method for unsupervised AD problems [23, 24]. Other methods [25, 26] cluster data and then use some criteria to give anomaly scores to every data point. A threshold is given and data whose anomaly score is higher than the threshold will be regarded as anomaly data. The criteria for thresholds includes distance between data and its nearest cluster center, or the number of clusters surrounding to this data point constrained by a constant distance. The computation of these models is decided by the scale of datasets and they are suitable for small and medium datasets.

2.2. Statistical based

Statistical anomaly detection models have an assumption that dataset obeys a specific probability distribution and use statistical methods to learn this probability distribution [27, 28]. A data point will be classified as anomaly data if its probability produced by learned probability distribution is lower than a threshold. Recently, kernel density estimation (KDE) [29] has been employed for building anomaly detection models, and proven to efficiently model normal data with unknown underlying distributions [30–32]. But KDE needs further improvement such as feature engineering for high-dimensional, data rich scenarios due to limited computational scalability and the curse of dimensionality [33]. One class support vector machines (OCSVM) [34] aims to map the data vectors from the input space to the feature space by means of a non-linear kernel function, and then a smooth surface or boundary is found in feature space that separates the image vectors into normal and anomalous measurements. It is widely used for unsupervised anomaly detection methods, such as [13, 35, 36]. Statistical Mixture model can be used for anomaly detection as well. For example, [37] proposed a method for fault detection in Hard Disk Drives based on a Gaussian Mixture Model. [38] gives an infinite inverted Dirichlet mixture model that performs good for object detection. [39] proposed Bayesian generalized inverted-Dirichlet mixture model (GiDMM) with single lower-bound approximation that can be used for anomaly detection. Besides, [40] firstly uses topic modeling for group anomaly detection. It allows groups to select their topic distributions from a dictionary of multinomials, which is learned from data to define what is normal.

2.3. Deep based

Deep learning is also used for detecting anomalies in recent years. Auto-encoder, a kind of specified structured deep neural network (DNN), is widely used in the field of anomaly detection. DNN based methods can be categorized into two classes: full deep methods and mixed methods.

Full deep methods are based on deep auto-encoder (DAE), such as AE, variational auto-encoder (VAE) [41], and deep convolutional auto-encoder (DCAE) [42]. When reconstructing data from features, the majority of data that have similar patterns can be reconstructed accurately while data do not contain these patterns has large reconstruction error. That is a characteristic of AE which is widely exploited for anomaly detection. Deep based AD methods [43, 44] train AE to extract features of normal data and use reconstruction error as anomaly score. Data with large anomaly score will be classified as anomaly data.

In addition, anomaly detection models such as neighbor based methods and statistical based methods can be combined with features and reconstruction errors from DAE . For example, Aytekin et al. [45] uses DAE as a feature extractor and then applies k-means algorithm to the extracted features to detect anomalies. Those points that are far away from the center of the class are regarded as anomalies. Ru et al. [33] propose a deep based anomaly detection method named Deep Support Vector Data Description (Deep SVDD), It jointly trains a deep neural network while optimizing a data-enclosing hypersphere in output space. Points which fall outside corresponding sphere are deemed anomalous.

3. Background

3.1. Deep auto-encoder

DAE , as one of the most popular methods for anomaly detection, is a multi-layer neural network which is suitable for data compression or feature extraction. Figure 1 shows a structure of DAE with five hidden layers. It consists of two parts: encoder and decoder. Usually the dimension of encoder's output are smaller than that of encoder's input, and the dimension of decoder's output should be equal to that of encoder's input. Its training principle is to make the output of decoder as close as possible to the input of encoder. As no labels are needed during training, it is an unsupervised learning manner.

DAE is fully connected between layers. The relationship between adjacent layer-to-layer of the encoder part is shown in equation (1). The relationship between adjacent layer-to-layer of the decoder part is shown in equation (2), where **h** represents output of previous layer and the first layer is $\boldsymbol{x_{in}}$; **W**, **b** and **R**(•) represents weight vector, bias vector and nonlinear activation function, respectively.

$$\boldsymbol{O_e} = \boldsymbol{R}(\boldsymbol{W_e} \bullet \boldsymbol{h} + \boldsymbol{b_e}) \tag{1}$$

$$\boldsymbol{O_d} = \boldsymbol{R}(\boldsymbol{W_d} \bullet \boldsymbol{h} + \boldsymbol{b_d}) \tag{2}$$

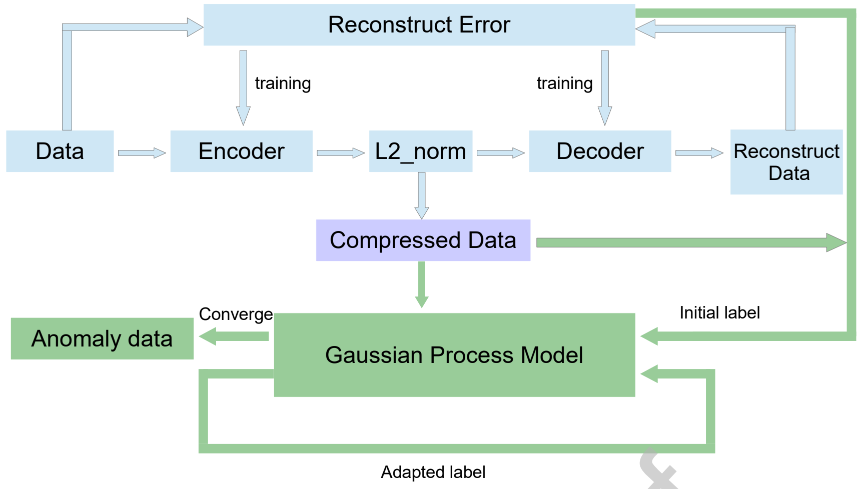


Figure 2: Framework of Deep Auto-encoding Gaussian Process Regression model

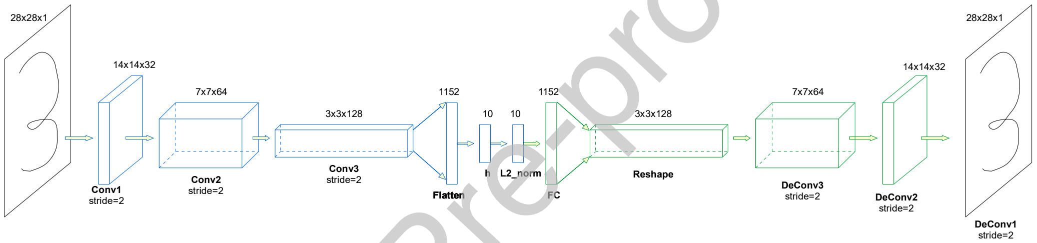


Figure 3: Structure of DCAE

The encoder gradually maps the input x_{in} to O_e in a non-linear manner by equation (1), completing the compression and feature extraction of the data, while the decoder gradually maps the feature O_e back to the original input O_d in a non-linear manner, which is a reverse process of the encoder by equation (2). The difference between the input of the encoder and the output of the decoder is called reconstruction error. With the training progress of DAE, the reconstruction error decreases. There are two reasons why DAE is introduced to our method: one is that DAE is a good feature extractor to compress high dimensional data, and consequently subsequent calculations can be reduced to a large extent. The other one is that after the DAE training is completed, its reconstruction error can be used to distinguish normal data from anomaly data in various ways. In our case, it can provide rough information that helps to initially label data which is later used for GPR. It is DAE's characteristic that reconstruction error can provide clues for anomaly detection: statistically speaking, DAE learns regular data faster than irregular data so that reconstruction error of normal data is smaller than that of abnormal data when DAE is trained to a certain extent [46].

3.2. Gaussian Process Regression

The Gaussian process, as a nonparametric model, is an important method in machine learning. Trained with a few samples, it can obtain the prediction results of the whole region and the variance information of the prediction that is used to measure confidence. The Gaussian process model is mainly divided into Gaussian process classification and Gaussian process regression (GPR), and we choose GPR in our model.

There are two reasons why we introduce GPR as the anomaly detector in our model. On one hand, GPR can give a reliable estimate in regression problems with a small number of learning samples [18]. On the other hand, in the previous section, DAE learns regular data faster than irregular data so that reconstruction error of normal data is smaller than that of abnormal data when DAE is trained to a certain extent. We choose the best reconstructed and the worst reconstructed data as the initial normal and abnormal data respectively. Those two subsets of the data are relatively small but much more reliable, which is suitable for training GPR.

The Gaussian process regression can be regarded as a generalization of the multivariate Gaussian distribution, which describes the distribution of functions. We can define a Gaussian process as equation (3). The overall statistical characteristics of the Gaussian process is determined by its mean and covariance, defined as equation (4)(5), where $\boldsymbol{x}$, $\boldsymbol{x'}$ are the input data; $c(\boldsymbol{x}, \boldsymbol{x'})$ represents the covariance of the instance $\boldsymbol{x}$ and $\boldsymbol{x'}$ (or the value calculated by the kernel function if kernel trick is applied to GPR defined as equation (6)).

$$f(\boldsymbol{x}) \sim GP(m(\boldsymbol{x}), c(\boldsymbol{x}, \boldsymbol{x'})) \tag{3}$$

$$m(\boldsymbol{x}) = E[f(\boldsymbol{x})] \tag{4}$$

$$c(\boldsymbol{x}, \boldsymbol{x'}) = E[(f((x)) - m((x)))(f(\boldsymbol{x'}) - m(\boldsymbol{x'})) \tag{5}$$

$$c(\boldsymbol{x}, \boldsymbol{x'}) = kernel(\boldsymbol{x}, \boldsymbol{x'}) \tag{6}$$

Let $D = \{\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}^*, \boldsymbol{y}^*\}$ be the dataset that contains both training data and test data for GPR, where $\boldsymbol{X}$, $\boldsymbol{y}$ are training input data and corresponding labels. $\boldsymbol{X}^*$, $\boldsymbol{y}^*$ are test input data and predicted output of the model. The regression model with Gaussian white noise is shown as equation (7)(8).

$$\boldsymbol{y} = f(\boldsymbol{X}) + \boldsymbol{\varepsilon} \tag{7}$$

$$f(\boldsymbol{X}) = \varphi(\boldsymbol{w}^T)\varphi(\boldsymbol{X}) + \boldsymbol{b} \tag{8}$$

where $f(\boldsymbol{X})$ is the predicted value, φ is some kind of kernel function in an implicit form; $\boldsymbol{y}$ is the observed value, $\boldsymbol{\varepsilon}$ is an independent Gaussian white noise which has mean value of 0 and variance of σ_n^2, i.e. $\boldsymbol{\varepsilon} \sim N(0, \sigma_n^2)$.Since the noise $\boldsymbol{\varepsilon}$ is a white noise independent of $f(\boldsymbol{X})$, when $f(\boldsymbol{X})$ obeys the Gaussian distribution, $\boldsymbol{y}$ also obeys the Gaussian distribution. Suppose the prior probability of $\boldsymbol{w}$ is $\boldsymbol{w} \sim N(0, \boldsymbol{I})$, where $\boldsymbol{I}$ is unit matrix. According to the prior knowledge of $\boldsymbol{y}$ in GP, the joint Gaussian prior distribution formed by $\boldsymbol{y}$ and $\boldsymbol{y}^*$ is established as equation (9)

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{y}^* \end{bmatrix} \sim N(0, \begin{bmatrix} C(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2 \boldsymbol{I} & C(\boldsymbol{X}, \boldsymbol{X}^*) \\ C(\boldsymbol{X}^*, \boldsymbol{X}) & C(\boldsymbol{X}^*, \boldsymbol{X}^*) \end{bmatrix}) \tag{9}$$

where $C(\boldsymbol{X}^*, \boldsymbol{X})$ is the $n \times n$ symmetric positive definite covariance matrix named kernel function matrix. $C(\boldsymbol{X}, \boldsymbol{X}^*)$ which is composed of $c(\boldsymbol{x}, \boldsymbol{x'})$ is the covariance matrix of the training set $\boldsymbol{X}$ and the test dataset $\boldsymbol{X}^*$. $C(\boldsymbol{X}^*, \boldsymbol{X}^*)$ is the covariance matrix of the test dataset $\boldsymbol{X}^*$ itself. Given the test dataset $\boldsymbol{X}^*$ and training set $\{\boldsymbol{X}, \boldsymbol{y}\}$, we can calculate posterior probability $p(\boldsymbol{y}^* | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}^*)$ according to Bayesian basic theory (equation (10)-(12)).

$$p(\boldsymbol{y}^* | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}^*) \sim N(\hat{y}(\boldsymbol{X}^*), \hat{\sigma}(\boldsymbol{X}^*)) \tag{10}$$

$$\hat{y}(\boldsymbol{X}^*) = C(\boldsymbol{X}^*, \boldsymbol{X}^*)(C(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2 \boldsymbol{I})^{-1} \boldsymbol{y} \tag{11}$$

$$\hat{\sigma}(\boldsymbol{X}^*) = C(\boldsymbol{X}^*, \boldsymbol{X}^*) - C(\boldsymbol{X}^*, \boldsymbol{X})(C(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2 \boldsymbol{I})^{-1} C(\boldsymbol{X}, \boldsymbol{X}^*) \tag{12}$$

$\hat{y}(\boldsymbol{X}^*)$, as the mean of the Gaussian process regression, is regarded as the prediction output in our model.

4. Proposed method

In this section, we will introduce our approach named deep auto-encoding Gaussian process regression (DAGPR). It is a hybrid approach that combines a deep convolutional auto-encoder with a Gaussian process regression model. The reasons why we choose DCAE and GPR are explained in Section 3. The structure of the DAGPR is shown in Figure 2. It consists of two parts: DCAE (blue part) and GPR model (green part). DCAE reduces the computation of GPR by compressing the data. And the reconstruction errors of compressed data can be used as a basis to help provide the initial label. The GPR part of our model , as the detector, receives the compressed data and initial labels from the DCAE, aiming to classify the data in the dataset into normal and abnormal classes. GPR is iteratively trained and the partial prediction results of the last training are used as the labels for the next training iteration. Finally, after a few iterations, the GPR model gives the anomaly scores and prediction labels for the total data.

It is worth noted that Gaussian process regression we adopt is a supervised method while the proposed model DAGPR training is within unsupervised category. During the training process, the unsupervised model only have access to unlabeled data, which results in a problem for training the supervised model GPR that needs labeled data. To solve this problem, we use self-assigned labels as shown later other than ground truth labels for training DAGPR. The ground truth label is only used when evaluating the performance of the proposed model.

```
Algorithm 1 DAGPR
Input: unlabeled dataset, $D = \{x_1, x_2, ..., x_n\}$ and threshold, T
Output: predicted labels L corresponding to D
Construct a convolutional auto-encoder whose construction is shown in Section 5.3 and initialize its parameters;
for M epochs (M is training epoch shown in Table 2) do
    D = shuffle(D);
    for input subset $s_{in}$ ∈ D' with batch size=BS(show in Table 2) do
        Output of decoder: $s_o$ = decoder($s_{in}$);
        Calculate average reconstruction error: $AVRes\_error = \frac{\sum(s_o - s_{in})^2}{n}$;
        Train network with Adam algorithm and loss $function = AVRes\_error$;
    end for
end for
Output of encoder: Com_data = encoder($D$);
Calculate reconstruction error $D\_res\_error = \sum_{i=1}^{n}(x_o - x_i)^2$ corresponding to D;
Concatenate Com_data and D_res_error to form a mixed feature matrix F of D;
Calculate the center of F: C = center(F);
Rank the Euclidean distance which is calculated by $F_i$ and $C$;
Choose initial training data Tr_D from both ends of the distance ranking and give self-assigned Tr_L for them;
for N epochs (N is iteration number of GPR shown in Table 2) do
    Train GPR with Tr_D and Tr_L;
    Predict mean $\mu$ of GPR for Com_data;
    Rank Com_data according to its $\mu$;
    Rechoose Tr_D and Tr_L: according to the ranking, the top 20% of Com_data is inlier training data and assigned inlier label; the last 5% of Com_data is outlier training label and assigned outlier label;
end for
Predict the final label L:
    $L(x) = \begin{cases} Normal & \mu(x_i) \leq \mathbf{T} \\ Anomaly & \mu(x_i) > \mathbf{T} \end{cases}$
```

4.1. DCAE Training

Our algorithm first trains DCAE. We apply L2 normalization to the output of encoder to improve the quality of data compression [3, 19]. The blue portion in Figure 2 is the framework of DCAE. We use mean squared error (MSE) between input and output of DCAE as reconstruction error for training DCAE shown in equation (13).

$$Loss = \frac{1}{n}\sum_{i=1}^{n}(x_o - x_i)^2 \tag{13}$$

x_i and x_o represent the input and output of DCAE, respectively; n is the size of training data. After a training period, we resend the training data to the encoder and apply the L2-normalization to the output of encoder to get the compressed data.

4.2. Self-assigned label generation

Because the Gaussian process is a supervised method, we need to provide labels for the compressed data before sent to GPR. The generation process of training data with corresponding self-assigned labels for GPR are as follows:

The input of GPR is hybrid features F which consists of the output of encoder and the corresponding reconstruction error. The Euclidean distance D_i of each feature F_i to the geometric center of all data features F which is calculated by $(F_1+F_2+...+F_n)/n$ is sorted in ascending order. We consider data with large Euclidean distance being more likely to be anomalous and we assign anomaly label to it. Data with small one is more likely to be normal and we assign normal label to it. Finally, we select a subset of the training data from both ends of the distance ranking as the initial training dataset for training GPR. We use the hybrid features containing the output of encoder and reconstruction error as the input of the GPR for two reasons. First one is that extracted features obtained from DCAE is a good representation of datasets. Second one is that under the condition that AE is trained with majority of normal examples, the reconstruction error is small for normal data and large for anomaly data [47]. This also benefits for the choice of self-assigned labels.

4.3. GPR iterative training

The data at both ends of the distance ranking is more reliable than the one in the middle of the distance ranking. If training can effectively learn from data at both ends of the distance ranking, detection performance of total data can be improved. So GPR is trained in an iterative manner to verify this idea in our model by alternating the following two steps:

- Discriminative labeling: obtain more confident normal and anomalous labeled data

- GPR learning: learn from the data and self-assigned labels and improve the performance of prediction

Discriminative labeling: label for compressed data is either anomaly label $l_i = 1$ or normal label $l_i = 0$. We assume that the characteristics of compressed normal data are similar. We determine the label by optimizing the following target:

$$\min_{l} \frac{\sum_{l_i=0}(\varepsilon_i - c^-)^2 + \sum_{l_i=1}(\varepsilon_i - c^+)^2}{\sum_{l_i}(\varepsilon_i - c)^2} \tag{14}$$

where $l = \{l_1, l_2, ..., l_i, ...\}$ is a collection of l_i, l_i is a variable predicted by the model according to ε_i, ε_i is I-th data's compressed data, c^+, c^-, c are the mean of GPR's output for anomaly data, normal data and total data, respectively. The numerator of equation (14) should be as small as possible to ensure that normal and abnormal data are separated as much as possible. The denominator normalizes the distance through the total data. Because ε_i is a scalar, optimizing equation (14) converts to sorting ε_i. We choose the most confident data near both ends of the sorted sequence to give labels 0 and 1 respectively.

GPR learning: After the labeling step, we use compressed data with corresponding self-assigned labels as training data for GPR. After the training process is completed, we predict labels of all data according to equations (10) (11).

These two steps alternate until the predicted performance no longer changes. In the process, the self-assigned labels of the data become purer (more accurate), which will be shown in the following experiments. Finally, we calculate the mean in equation (14) as the output of our method, and abnormal data can be separated from normal data by a threshold. Details about our algorithm are shown in Algorithm 1.

5. Experiments

Our approach detects anomalies from high dimensional noisy data in an unsupervised way. It is evaluated on four publicly available datasets and is compared with main stream unsupervised methods[1]. Four benchmark datasets are MNIST[2], Fashion MNIST[3], CIFAR-10[4] and STL-10[5], which are illustrated as Table 1. There is versatility of anomalies in some applications such as intrusion detection [48–50], retrieve images from search engines [46, 51, 52]. We target these applications and choose one class from each dataset as normal class while the rest is considered as the anomaly class in the datasets. The metrics of performance measure we use is Area Under the Curve (AUC). It is the area under the receiver operating characteristic curve (ROC), which is used as an evaluation of anomaly detection method. ROC curve is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. Using classification datasets to create one-class classification setups allows us to evaluate the results quantitatively via AUC (Area Under ROC Curve) measure by using the ground truth labels [3, 13, 33, 53–55]. The proposed method is an unsupervised algorithm, so we do not use any ground truth labels during training except for evaluating the performance of the proposed method.

5.1. Datasets

- MNIST: It contains 70000 grayscale handwritten digits from 0 to 9. We select one digit as normal class. It is a 28x28 dimension grayscale image dataset.

- Fashion MNIST: It comprises 70000 grayscale images of fashion products. It has 10 categories with 7000 images per category. its image dimension is 28x28.

- CIFAR-10: It consists of 10 semantic concepts of color images and each concept contains about 6000 images. The image size is 32x32x3.

- STL-10: It is an color image recognition dataset containing 10 classes. We select one class as normal one.The image size is 96x96x3.

5.2. Competitive methods

Our method is compared with anomaly detection methods including one-class support vector machine (OCSVM), local outlier factor (LOF), isolation forest (IF), DCAE and Deep Support Vector Data Description (Deep SVDD).

I. OCSVM

We use the same setting of the OCSVM parameter as in paper [33]: the kernel function is the Gaussian kernel. We use the grid search to select the inverse length scale γ and υ, where $\gamma \in \{2^{-10}, 2^{-9}, 2^{-8}, 2^{-7}, 2^{-6}, 2^{-5}, 2^{-4}, 2^{-3}, 2^{-2}, 2^{-1}\}$, and $\upsilon \in \{0.1, 0.01\}$.

II. LOF

[1] We provide our code at https://github.com/fanjinan

[2] https://keras.io/datasets/#mnist-database-of-handwritten-digits

[3] https://keras.io/datasets/#fashion-mnist-database-of-fashion-articles

[4] https://keras.io/datasets/#cifar10-small-image-classification

[5] https://cs.stanford.edu/ acoates/stl10/

Table 1: Dataset Description

Datasets	Dimensions	points
MNIST	784(28*28)	60000
Fashion MNIST	784(28*28)	60000
CIFAR-10	3072(32*32*3)	60000
STL-10	27648(96*96*3)	13000

Table 2: Parameter Settings for DAGPR

Datasets	Mini-batch size of DCAE	Training epoch of DCAE	Number of iteration
MNIST	256	200	10
Fashion NNIST	256	400	20
CIFAR-10	256	1000	20
STL-10	64	300	10

The number of neighbors is a critical parameter for LOF. In the experiment, we set it to 200, because this setting satisfies the requirement that it should be greater than the minimum number of the anomaly classes and should be less than the maximum number of samples that may be local outliers. In addition, in order to improve the performance of the LOF, we set the contamination parameter of LOF as the true anomaly rate in the code.

III. IF

As the number of basic estimators increases, the IF will gradually converge. In our experimental tests, we found that 500 estimators are sufficient to converge the IF. In addition, we set the data contamination parameter as the ground truth anomaly rate for better results.

IV. DCAE

DCAE shares the same structure and hyper parameter settings as the deep convolutional auto-encoder in our method. It uses the reconstruction error of the data as the anomaly score.

V. Deep SVDD

Stochastic gradient descent (SGD) is used for optimization with learning rate 10^{-5} for 250 epochs. It chooses υ from $\upsilon \in \{0.01, 0.1\}$, where υ is an upper bound on the fraction of outliers and a lower bound on the fraction of samples being outside or on the boundary of the hypersphere. We did not implement experiments with Deep SVDD and the results are borrowed from the paper.

5.3. Data Preprocessing and parameter settings

5.3.1. Data pre-processing

Input image is normalized between [0,1] before sent to the anomaly detection models. It is noted that due to the different image sizes of the verification datasets, the input and output dimensions of the DCAE in our model also change accordingly.

5.3.2. DCAE parameters

The DCAE adopted in our method as shown in Figure 3 shares the same structure in paper [56] except that we apply L2 normalization to the output of the encoder. This structure has seven layers including three convolutional layers, a dense layer as output of encoder and three deconvolution layers. The sizes of filter in encoding part are 5x5, 5x5, 3x3, respectively. Convolutions are applied with 2x2 strides and with ReLU activations. Then through a dense layer of size 10 with L2 normalization, we can obtain 10-dimensional features. The decoding part is the reverse of the encoding process. Filters of sizes 3x3, 5x5, 5x5 are used for deconvolution part and ReLU function is applied for each layer except the last layer. We adopt Adam [57] to compute the equation (13) and optimize our algorithm. The learning rate of Adam algorithm is set as 0.001 for all datasets. The mini-batch and training epoch for training DCAE is set as Table 2.

5.3.3. Iterative GPR parameters

In GPR parameter setting, our kernel function is the linear combination of Gaussian kernel, linear kernel and constant kernel showing in equation (15),

$$k(x, x') = \theta_0 exp(-\frac{\theta_1}{2} x - x'^2) + \theta_2 + \theta_3 x^T x' \qquad (15)$$

where $\boldsymbol{\theta_0}$- $\boldsymbol{\theta_3}$ are all constants. This kind of kernel is widely used for GPR [58]. We do not use any ground truth label for GPR training. But in the process of iterative training, we have the labels getting from the previous iteration. The normal label is set as 0 while the anomaly label is set as 1. We find that GPR converges after a few iterations as shown in the following experiments. Therefore, the number of iterations of GPR is set as Table 2 shows, which is an empirical value.

5.4. Results and discussion

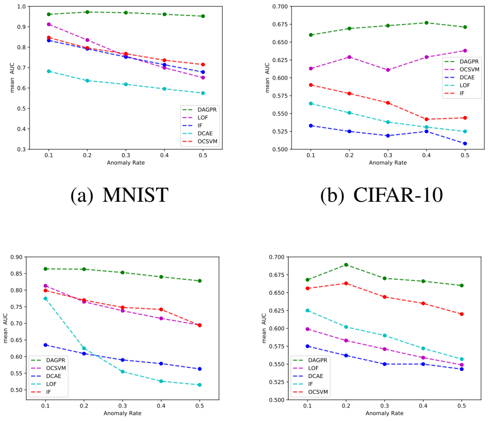
To evaluate the general performance of the proposed method, experiments are conducted on four publicly available datasets and comparison is made in terms of AUC among five common used methods in anomaly detection. Furthermore, we analyze the influence of anomaly rate and compare its robustness with other methods. Finally, we present the convergence analysis and explore the effectiveness of GPR in our method.

5.4.1. General Performance on MNIST, CIFAR-10, Fashion MNIST, STL-10

We have evaluated our model on four high-dimension datasets whose dimensions are shown in Table 1. The results on MNIST, CIFAR-10, Fashion MNIST and STL-10 datasets are shown from Table 3 to Table 6. Each result is the average of five runs.

In general, DAGPR is superior to other classical anomaly detection methods with average AUCs 0.961, 0.669, 0.865, 0.684 on the four datasets. In Table 3, although LOF and Deep SVDD have slightly higher AUC than that of our model when inlier number is 1, 6, and 9, our model shows overall better performance. For example, AUC for all inlier number situations are higher than 0.930. The average AUC of the DAGPR is better than that of the Deep SVDD, which is the best result we can find on the MNIST dataset. Results on CIFAR-10 are presented in Table 4. DAGPR shows good performance when the normal categories are Airplane, Automobile, Ship and Truck. Although our performance is not particularly satisfactory for the rest of normal categories, DAGPR achieves the best overall performance with an average AUC of 0.669. From Table 5 we can see that DAGPR gets better performance when the normal categories are T-shirt, Trouser, Pullover, Coat, Sandal, Sneaker and Ankle boot. Table 6 shows the result on STL-10 dataset which has the highest dimension of 27648. DAGPR achieves the best performance when the normal categories are Airplane, Bird, Cat, Monkey, Ship and Truck. Its average AUC is 0.680, which is greater than that of competive methods.

The detection performance of model is susceptible to the curse of dimensionality and DCAE is chosen to address this problem. The analysis of DCAE in our proposed method cannot be conducted straightforwardly by removing it from the DAGPR since DCAE not only reduces feature dimension, but also provides the initial subset training dataset for GPR. Therefore, to further illustrate our model is suitable for dealing with high dimensional anomaly detection problems, a hybrid model named DASVM is introduced for comparison and results on MNIST, CIFAR-10, Fashion MNIST and STL-10 are shown in Table 7. DASVM is a hybrid of DCAE and OCSVM. It reduces dimension first and detects anomalies with the reduced features, which shares the similar philosophy with the proposed method. The DCAE and OCSVM parameters settings in DASVM are the same as DCAE and OCSVM in section 5.3. DASVM performs better than OCSVM on all datasets. For example, in Table 7, DASVM's average AUC is 0.910 on MNIST dataset, while OCSVM is 0.847. DASVM performs better on CIFAR-10 with average AUC 0.647 compared with OCSVM's average AUC 0.629 as shown in Table 7. This indicates that OCSVM performs better with DCAE implementing dimensionality reduction. Thus, taking DCAE as an approach to reduce features has positive effects on the performance of proposed model.



(a) MNIST

(b) CIFAR-10

(c) Fashion MNIST

(d) STL-10

Figure 4: Anomaly rate analysis on (a)MNIST (b)CIFAR-10 (c)Fashion MNIST (d)STL-10 dataset; AUCs are average over all concepts in dataset

5.4.2. Influence of different anomaly rates

We extend the above experiments to test DAGPR's robustness to anomaly rate ranging from 0.1 to 0.5. The results are shown in Fig 4.

In general, we can see from Fig 4 that AUC values of DAGPR (green points) are the largest in different datasets and different anomaly rates. For ex-

Table 3: Average AUCs on MNIST with anomaly rate 0.1

inlier	DAGPR	LOF	IF	DCAE	OCSVM	Deep SVDD
0	**0.993**	0.868	0.880	0.631	0.890	0.980
1	0.993	0.972	0.993	0.977	0.978	**0.997**
2	**0.979**	0.841	0.697	0.596	0.731	0.917
3	**0.959**	0.890	0.772	0.546	0.794	0.919
4	**0.949**	0.887	0.867	0.700	0.879	**0.949**
5	**0.934**	0.922	0.737	0.557	0.772	0.885
6	0.970	**0.976**	0.883	0.746	0.859	0.983
7	**0.951**	**0.951**	0.903	0.820	0.908	0.946
8	**0.943**	0.853	0.721	0.489	0.806	0.939
9	0.938	0.956	0.875	0.757	0.886	**0.965**
Average	**0.961**	0.912	0.833	0.682	0.847	0.948

Table 4: Average AUCs on CIFAR-10 with anomaly rate 0.2

inlier	DAGPR	LOF	IF	DCAE	OCSVM	Deep SVDD
Airplane	**0.751**	0.633	0.633	0.612	0.596	0.617
Automobile	**0.737**	0.380	0.413	0.358	0.653	0.659
Bird	0.595	**0.673**	0.659	0.638	0.658	0.508
Cat	0.564	0.472	0.481	0.540	0.590	**0.591**
Deer	0.692	0.705	**0.732**	0.657	0.719	0.609
Dog	0.572	0.469	0.499	0.500	0.524	**0.657**
Frog	0.692	0.673	**0.707**	0.558	0.681	0.677
Horse	0.531	0.488	0.529	0.440	**0.705**	0.673
Ship	**0.767**	0.613	0.668	0.630	0.615	0.759
Truck	**0.793**	0.408	0.468	0.319	0.551	0.731
Average	**0.669**	0.551	0.579	0.525	0.629	0.648

Table 5: Average AUCs on Fashion MNIST with anomaly rate 0.1

inlier	DAGPR	OCSVM	DCAE	IF	LOF
T-shirt	**0.858**	0.834	0.633	0.791	0.788
Trouser	**0.981**	0.891	0.949	0.908	0.870
Pullover	**0.873**	0.808	0.643	0.706	0.860
Dress	0.839	0.826	0.691	0.847	**0.883**
Coat	**0.901**	0.820	0.579	0.774	0.870
Sandal	**0.844**	0.721	0.489	0.824	0.568
Shirt	0.678	0.784	0.587	0.627	**0.787**
Sneaker	**0.970**	0.924	0.699	0.911	0.704
Bag	0.736	0.679	0.417	0.645	**0.741**
Ankle boot	**0.965**	0.840	0.660	0.797	0.679
Average	**0.865**	0.813	0.635	0.783	0.775

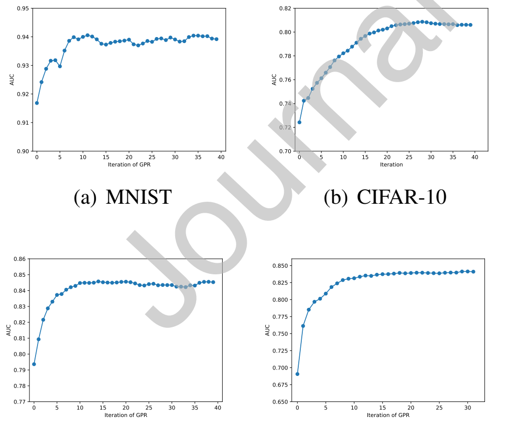
ample, it can be shown from Fig 4 (**b**) that compared with other methods, the proposed method shows superior performance with AUC larger than 0.650 under different anomaly rates, while the best performance of the compared method is OCSVM with the best AUC around 0.630. In addition, it can be seen from Fig 4 that as the anomaly rate changes, AUC value of DAGPR changes the least, and AUC value of the comparative method generally decreases as the anomaly rate increases. This implies that DAGPR is more robust to the changes of anomaly rate when it ranges from 0.1 to 0.5.

Table 6: Average AUCs on STL-10 with anomaly rate 0.2

inlier	DAGPR	OCSVM	DCAE	IF	LOF
Airplane	**0.783**	0.378	0.457	0.430	0.543
Bird	**0.767**	0.652	0.685	0.687	0.641
Car	0.583	**0.598**	0.515	0.583	0.578
Cat	**0.806**	0.423	0.407	0.507	0.544
Deer	0.612	**0.628**	0.610	0.624	0.620
Dog	0.687	**0.766**	0.698	0.747	0.658
Horse	0.572	0.570	0.537	0.562	**0.733**
Monkey	**0.587**	0.552	0.466	0.565	0.536
Ship	**0.666**	0.603	0.545	0.636	0.594
Truck	**0.774**	0.665	0.699	0.680	0.622
Average	**0.684**	0.584	0.562	0.602	0.607

Table 7: Average AUCs on OCSVM and DASVM

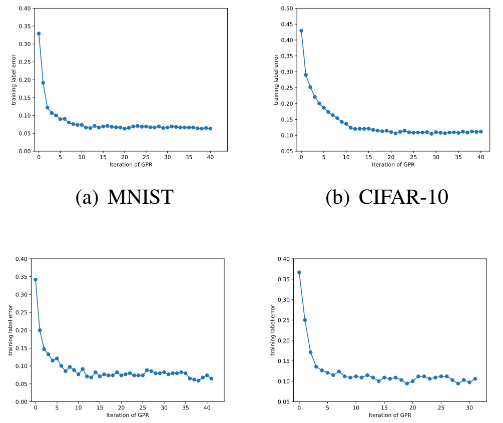
MNIST			CIFAR-10			Fashion MNIST			STL-10		
inlier	OCSVM	DASVM	inlier	OCSVM	DASVM	inlier	OCSVM	DASVM	inlier	OCSVM	DASVM
0	0.890	0.922	Airplane	0.596	0.721	T-shirt	0.834	0.803	Airplane	0.543	0.648
1	0.978	0.973	Automobile	0.653	0.597	Trouser	0.891	0.949	Bird	0.641	0.674
2	0.731	0.840	Bird	0.658	0.654	Pullover	0.808	0.840	Car	0.578	0.605
3	0.794	0.908	Cat	0.590	0.568	Dress	0.826	0.815	Cat	0.544	0.704
4	0.879	0.885	Deer	0.719	0.683	Coat	0.820	0.848	Deer	0.620	0.619
5	0.772	0.887	Dog	0.524	0.598	Sandal	0.721	0.799	Dog	0.658	0.708
6	0.859	0.910	Frog	0.681	0.711	Shirt	0.784	0.788	Horse	0.733	0.604
7	0.877	0.909	Horse	0.705	0.578	Sneaker	0.924	0.923	Monkey	0.536	0.611
8	0.806	0.936	Ship	0.615	0.718	Bag	0.679	0.842	Ship	0.594	0.660
9	0.886	0.930	Truck	0.551	0.637	Ankle boot	0.840	0.876	Truck	0.622	0.627
Average	0.847	0.910	Average	0.629	0.647	Average	0.813	0.849	Average	0.607	0.646



(a) MNIST (b) CIFAR-10

(c) Fashion MNIST (d) STL-10

Figure 5: The anomaly detection performance (AUC) in learning process of GPR iteration on (a)MNIST (b)CIFAR-10 (c)Fashion MNIST (d)STL-10 dataset



(a) MNIST (b) CIFAR-10

(c) Fashion MNIST (d) STL-10

Figure 6: Self-assigned label error in GPR training process on (a)MNIST (b)CIFAR-10 (c)Fashion MNIST (d)STL-10 dataset

5.4.3. Convergence analysis

In this part, we study the convergence performance of our proposed algorithm. The number of iterations of GPR is considered as an indication of the convergence speed. The convergence curves of DAGPR on all datasets are shown in Fig 5. As the number of GPR iterations increases, the anomaly detection performance of DAGPR increases rapidly until convergence and the trend is universally observed in the experimrnts. On dataset MNIST, Fashion MNIST and STL-10, the performance converges at around 10^{th} iteration, while for CIFAR-10, it converges at around 20^{th} iteration.

To explore why GPR is effective as the iteration goes, we do more insight experiments. Fig 6 shows the self-assigned label error in GPR training process on all datasets. We can see that as the number of iterations of GPR increases, the error rate of training self-assigned labels decreases. More details, our initial labeled error ranges from 0.3-0.45, which means more than half of the labels we assigned are right at the beginning; converged label error ranges from 0.05-0.15, which means around 0.85 to 0.95 labels we assigned are right at the convergence. This phenomenon exists in different normal categories. Therefore, we think the prediction of GPR becomes better by training, and better prediction results in more correctly labeled data for next training of GPR. We think it is the "positive feedback" that makes the iterative process of our GPR effective.

6. Conclusion

In this paper, we introduce a novel unsupervised anomaly detection method that first combines an auto-encoder and a Gaussian process regression to improve anomaly detection performance. The deep convolutional auto-encoder is applied for feature extraction and dimension reduction, and the Gaussian process regression is used to train the model to detect anomalies. Experiments show that our model obtain the best overall performance against the compared unsupervised anomaly detection methods on MNIST, CIFAR-10, Fashion MNIST and STL-10 datasets. It also shows that our method is suitable for high-dimensional input and is robust to the dataset where the anomaly rate varies form 0.1 to 0.5. In the future work, we will be interested in applying our approach to more large-scale datasets for anomaly detection.

ACKNOWLEDGMENTS

This research work was partly supported by the Natural Science Foundation of China and Jiangsu (Project No. 61750110529, 61850410535, 61671148, BK20161147), the Research and Innovation Program for Graduate Students in Universities of Jiangsu Province (Grant SJCX17_0048, SJCX18_0058).

References

References

[1] L. Deecke, R. Vandermeulen, L. Ruff, S. Mandt, M. Kloft, Anomaly detection with generative adversarial networks.

[2] D. M. Hawkins, Identification of outliers, Vol. 11, Springer, 1980.

[3] M. Nicolau, J. McDermott, et al., Learning neural representations for network anomaly detection, IEEE transactions on cybernetics 49 (8) (2018) 3074–3087.

[4] I. Hwang, S. Kim, Y. Kim, C. E. Seah, A survey of fault detection, isolation, and reconfiguration methods, IEEE transactions on control systems technology 18 (3) (2010) 636–653.

[5] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, M. Rajarajan, A survey of intrusion detection techniques in cloud, Journal of network and computer applications 36 (1) (2013) 42–57.

[6] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, G. Langs, Unsupervised anomaly detection with generative adversarial networks to guide marker discovery, in: International Conference on Information Processing in Medical Imaging, Springer, 2017, pp. 146–157.

[7] M. Amer, M. Goldstein, S. Abdennadher, Enhancing one-class support vector machines for unsupervised anomaly detection, in: Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description, ACM, 2013, pp. 8–15.

[8] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, H. Chen, Deep autoencoding gaussian mixture model for unsupervised anomaly detection.

[9] V. Chandola, A. Banerjee, V. Kumar, Anomaly detection: A survey, ACM computing surveys (CSUR) 41 (3) (2009) 15.

[10] L. Van Der Maaten, E. Postma, J. Van den Herik, Dimensionality reduction: a comparative, J Mach Learn Res 10 (66-71) (2009) 13.

[11] I. Jolliffe, M. Lovric, International encyclopedia of statistical science (2014).

[12] D. E. Rumelhart, G. E. Hinton, R. J. Williams, Learning representations by back-propagating errors, nature 323 (6088) (1986) 533.

[13] S. M. Erfani, S. Rajasegarar, S. Karunasekera, C. Leckie, High-dimensional and large-scale anomaly detection using a linear one-class svm with deep learning, Pattern Recognition 58 (2016) 121–134.

[14] D. Rajashekar, A. N. Zincir-Heywood, M. I. Heywood, Smart phone user behaviour characterization based on autoencoders and self organizing maps, in: 2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW), IEEE, 2016, pp. 319–326.

[15] M. Daszykowski, K. Kaczmarek, Y. Vander Heyden, B. Walczak, Robust statistics in data analysisa review: basic concepts, Chemometrics and intelligent laboratory systems 85 (2) (2007) 203–219.

[16] Y. Xiao, H. Wang, W. Xu, J. Zhou, Robust one-class svm for fault detection, Chemometrics and Intelligent Laboratory Systems 151 (2016) 15–25.

[17] S. Yin, X. Zhu, C. Jing, Fault detection based on a robust one class support vector machine, Neurocomputing 145 (2014) 263–268.

[18] R. Herbrich, N. D. Lawrence, M. Seeger, Fast sparse gaussian process methods: The informative vector machine, in: Advances in neural information processing systems, 2003, pp. 625–632.

[19] C. Aytekin, X. Ni, F. Cricri, E. Aksu, Clustering and unsupervised anomaly detection with l2 normalized deep auto-encoder representations, arXiv preprint arXiv:1802.00187.

[20] Y. Zhang, B. Du, L. Zhang, S. Wang, A low-rank and sparse matrix decomposition-based mahalanobis distance method for hyperspectral anomaly detection., IEEE Trans. Geoscience and Remote Sensing 54 (3) (2016) 1376–1389.

[21] M. M. Breunig, H.-P. Kriegel, R. T. Ng, J. Sander, Lof: identifying density-based local outliers, in: ACM sigmod record, Vol. 29, ACM, 2000, pp. 93–104.

[22] F. T. Liu, K. M. Ting, Z.-H. Zhou, Isolation forest, in: 2008 Eighth IEEE International Conference on Data Mining, IEEE, 2008, pp. 413–422.

[23] F. T. Liu, K. M. Ting, Z.-H. Zhou, Isolation-based anomaly detection, ACM Transactions on Knowledge Discovery from Data (TKDD) 6 (1) (2012) 3.

[24] L. Puggini, S. McLoone, An enhanced variable selection and isolation forest based methodology for anomaly detection with oes data, Engineering Applications of Artificial Intelligence 67 (2018) 126–135.

[25] L. Portnoy, Intrusion detection with unlabeled data using clustering, Ph.D. thesis, Columbia University (2000).

[26] H. Du, S. Zhao, D. Zhang, J. Wu, Novel clustering-based approach for local outlier detection, in: Computer Communications Workshops (INFOCOM WKSHPS), 2016 IEEE Conference on, IEEE, 2016, pp. 802–811.

[27] C. Wang, K. Viswanathan, C. Lakshminarayan, V. Talwar, W. Satterfield, K. Schwan, Statistical techniques for online anomaly detection in data centers., in: Integrated Network Management, Citeseer, 2011, pp. 385–392.

[28] N. A. Heard, D. J. Weston, K. Platanioti, D. J. Hand, et al., Bayesian anomaly detection methods for social networks, The Annals of Applied Statistics 4 (2) (2010) 645–662.

[29] E. Schubert, A. Zimek, H.-P. Kriegel, Generalized outlier detection with flexible kernel density estimates, in: Proceedings of the 2014 SIAM International Conference on Data Mining, SIAM, 2014, pp. 542–550.

[30] J. Kim, C. D. Scott, Robust kernel density estimation, Journal of Machine Learning Research 13 (Sep) (2012) 2529–2565.

[31] M. Nicolau, J. McDermott, et al., One-class classification for anomaly detection with kernel density estimation and genetic programming, in: European Conference on Genetic Programming, Springer, 2016, pp. 3–18.

[32] M. Nicolau, J. McDermott, et al., A hybrid autoencoder and density estimation model for anomaly detection, in: International Conference on Parallel Problem Solving from Nature, Springer, 2016, pp. 717–726.

[33] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, M. Kloft, Deep one-class classification, in: International Conference on Machine Learning, 2018, pp. 4393–4402.

[34] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, R. C. Williamson, Estimating the support of a high-dimensional distribution, Neural computation 13 (7) (2001) 1443–1471.

[35] D. Wang, D. S. Yeung, E. C. Tsang, Structured one-class classification, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 36 (6) (2006) 1283–1295.

[36] S. Rajasegarar, C. Leckie, J. C. Bezdek, M. Palaniswami, Centered hyperspherical and hyperellipsoidal one-class support vector machines for anomaly detection in sensor networks, IEEE Transactions on Information Forensics and Security 5 (3) (2010) 518–533.

[37] L. P. Queiroz, F. C. M. Rodrigues, J. P. P. Gomes, F. T. Brito, I. C. Chaves, M. R. P. Paula, M. R. Salvador, J. C. Machado, A fault detection method for hard disk drives based on mixture of gaussians and nonparametric statistics, IEEE Transactions on Industrial Informatics 13 (2) (2016) 542–550.

[38] Z. Ma, Y. Lai, W. B. Kleijn, Y.-Z. Song, L. Wang, J. Guo, Variational bayesian learning for dirichlet process mixture of inverted dirichlet distributions in non-gaussian image feature modeling, IEEE transactions on neural networks and learning systems 30 (2) (2018) 449–463.

[39] Z. Ma, J. Xie, Y. Lai, J. Taghia, J.-H. Xue, J. Guo, Insights into multiple/single lower bound approximation for extended variational inference in non-gaussian structured data modeling, IEEE transactions on neural networks and learning systems.

[40] L. Xiong, B. Póczos, J. Schneider, A. Connolly, J. VanderPlas, Hierarchical probabilistic models for group anomaly detection, in: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, 2011, pp. 789–797.

[41] D. P. Kingma, M. Welling, Auto-encoding variational bayes, arXiv preprint arXiv:1312.6114.

[42] J. Masci, U. Meier, D. Cireşan, J. Schmidhuber, Stacked convolutional auto-encoders for hierarchical feature extraction, in: International Conference on Artificial Neural Networks, Springer, 2011, pp. 52–59.

[43] M. Sakurada, T. Yairi, Anomaly detection using autoencoders with nonlinear dimensionality reduction, in: Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis, ACM, 2014, p. 4.

[44] J. An, S. Cho, Variational autoencoder based anomaly detection using reconstruction probability, Special Lecture on IE 2 (2015) 1–18.

[45] K. K. Reddy, S. Sarkar, V. Venugopalan, M. Giering, Anomaly detection and fault disambiguation in large flight data: a multimodal deep auto-encoder approach, in: Annual Conference of the Prognostics and Health Management Society, 2016.

[46] Y. Xia, X. Cao, F. Wen, G. Hua, J. Sun, Learning discriminative reconstructions for unsupervised outlier removal, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1511–1519.

[47] D. C. Le, A. N. Zincir-Heywood, M. I. Heywood, Data analytics on network traffic flows for botnet behaviour detection, in: 2016 IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, 2016, pp. 1–7.

[48] M. H. Ali, B. A. D. Al Mohammed, A. Ismail, M. F. Zolkipli, A new intrusion detection system based on fast learning network and particle swarm optimization, IEEE Access 6 (2018) 20255–20261.

[49] N. Shone, T. N. Ngoc, V. D. Phai, Q. Shi, A deep learning approach to network intrusion detection, IEEE Transactions on Emerging Topics in Computational Intelligence 2 (1) (2018) 41–50.

[50] A. L. Buczak, E. Guven, A survey of data mining and machine learning methods for cyber security intrusion detection, IEEE Communications Surveys & Tutorials 18 (2) (2015) 1153–1176.

[51] W. Liu, G. Hua, J. R. Smith, Unsupervised one-class learning for automatic outlier removal, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 3826–3833.

[52] F. Schroff, A. Criminisi, A. Zisserman, Harvesting image databases from the web, IEEE transactions on pattern analysis and machine intelligence 33 (4) (2010) 754–766.

[53] J. Guo, G. Liu, Y. Zuo, J. Wu, An anomaly detection framework based on autoencoder and nearest neighbor, in: 2018 15th International Conference on Service Systems and Service Man-

agement (ICSSSM), IEEE, 2018, pp. 1–6.

[54] Y. Liu, Z. Li, C. Zhou, Y. Jiang, J. Sun, M. Wang, X. He, Generative adversarial active learning for unsupervised outlier detection, IEEE Transactions on Knowledge and Data Engineering.

[55] C. Wang, Y.-M. Zhang, C.-L. Liu, Anomaly detection via minimum likelihood generative adversarial networks, in: 2018 24th International Conference on Pattern Recognition (ICPR), IEEE, 2018, pp. 1121–1126.

[56] X. Guo, X. Liu, E. Zhu, J. Yin, Deep clustering with convolutional autoencoders, in: International Conference on Neural Information Processing, Springer, 2017, pp. 373–382.

[57] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980.

[58] B. Yu, Anomaly detection algorithm based on gaussian model (In Chinese), thesis.

biographies of all authors：

Jinan Fan received the B.S. degree in from Chongqing University, Chongqing, China, 2017. Currently, he is pursuing the master's degree at National ASIC Center, Southeast University, China. His research interests include anomaly detection, machine learning and deep learning.

Qianru Zhang is a Ph.D. student at National ASIC Center in School of Electronic Science and Engineering, Southeast University, China. She obtained her M.S. in Electrical and Computer Engineering from University of California, Irvine in 2016. Her research interests include digital signal processing, big data analysis and deep learning techniques.

Jialei Zhu is currently pursuing the master's degree at National ASIC Center in School of Integrated circuit engineering, Southeast University, China. Her research interests include deep learning techniques.

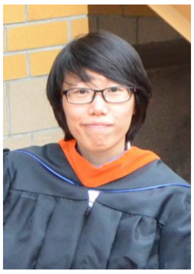
Meng Zhang received the B.S. degree in Electrical Engineering from the China University of Mining and Technology, Xuzhou, China, in 1986, and the M.S. and Ph.D. degrees in Bioelectronics Engineering from Southeast University, Nanjing, China, in 1993 and 2014 respectively.
He is currently a professor in National ASIC System Research Center, College of Electronic Science and Engineering of Southeast University, Nanjing, PR China. He is a faculty adviser of PhD graduates. His research interests include digital signal and image processing, digital communication systems, wireless sensor networks, information security and assurance, cryptography, and digital integrated circuit design, machine learning etc. He is an author or coauthor of more than 50 referred journal and international conference papers and a holder of more than 60 patents, including some PCT, US patents.

Zhou Yang is currently pursuing the master's degree at National ASIC Center in School of Electronic Science and Engineering, Southeast University, China. His research interests include computer vision and big data analysis.

Hanxiang Cao is currently pursuing the master's degree at National ASIC Center in School of Integrated circuit engineering, Southeast University, China. His research interests include deep learning techniques and big data analysis.

biographies and pictures of all authors:

Jinan Fan



Qianru Zhang

Jialei Zhu

Meng Zhang

Zhou Yang

Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: