Anomaly detecting and ranking of the cloud computing platform by multi-view learning

Jing Zhang

Abstract Anomaly detecting as an important technical in cloud computing is applied to support smooth running of the cloud platform. Traditional detecting methods based on statistic, analysis, etc. lead to the high false-alarm rate due to non-adaptive and sensitive parameters setting. We presented an online model for anomaly detecting using machine learning theory. However, most existing methods based on machine learning linked all features from difference sub-systems into a long feature vector directly, which is difficult to both exploit the complement information between sub-systems and ignore multi-view features enhancing the classification performance. Aiming to this problem, the proposed method automatic fuses multi-view features and optimize the discriminative model to enhance the accuracy. This model takes advantage of extreme learning machine (ELM) to improve detection efficiency. ELM is the single hidden layer neural network, which is transforming iterative solution of the output weights to solution of linear equations and avoiding the local optimal solution. Moreover, we rank anomies according to the relationship between samples and the classification boundary, and then assigning weights for ranked anomalies, retraining the classification model finally. Our method exploits the complement information between sub-systems sufficiently, and avoids the influence from imbalance dataset, therefore, deal with various challenges from the cloud computing platform. We deploy the privately cloud platform by Openstack, verifying the proposed model and comparing results to the state-of-the-art methods with better efficiency and simplicity.

F. Author
School of computer and information technology, Liaoning Normal University, Liushu south street NO.1 Ganjingzi district, Dalian and 116018, China
Tel.: +86-0411-85992418
Fax: +86-0411-85992418
E-mail: zhangjing_0412@163.com

1 Introduction

Cloud computing makes possible auto-scaling and using resources at all time, moreover, avoid waste or out of expectation when developers deploy applications[1, 2]. However, multiple anomalies in the cloud platform become both the major bottleneck for high available and the primary cause to impede development, it is important to how to build efficient anomaly detecting models. Previous studies in anomaly detecting focus on statistical analysis of running curve, which is computing and comparing curves and setting the threshold values to find anomalies by operators. The kind of methods produces both low accuracy and the high false alarm ratio due to adjust parameters by manually in unknown data distribution. Therefore, the anomaly detecting problem in cloud computing is often formulated as a data association problem based on machine learning. Machine learning models that contains data drive, non-linear fitting and incremental provide preconditions for the problem of detection defining and solving.

In detecting-by-learning techniques, the performance depends on ten detection accuracy and the challenge from the high-dimension vector that consists of multiple sub-systems [3]. Information redundancy and noisy from different features will result in failures of detecting. Therefore, feature extraction and dimensionality reduction as efficient means enhance the detecting performance in cloud computing based on virtualization. Sub-space learning extracts principal features by building the optimum projection space such as principal component analysis (PCA) [4], locality preserving projections (LPP) [5], linear discriminant analysis (LDA) [6]. In order to improve expression of features, independent component analysis (ICA) [7] is proposed. Robust dimensionality reduction methods by sparse representation and low-rank learning including robust PCA model [8], sub-space recover model [9], etc. Above methods are applied in information selection in cloud computing due to both high-integrity date description and high-effective computing. Guan, etc. exploit PCA to obtain the most relevant components, and then adaptive kalman filter (AKF) regard as classifier to enhance the detecting performance [10]. Fu, etc., utilize PAC to extract features of the cloud platform, which is different to the research from Guan, and it is selecting features in indicator vectors by mutual information method, and then extracting the most important features on selected features based on PCA [11]. Lan, etc., analyze in such a way as to distinguish between PCA and ICA in cloud computing detecting, and accept that ICA achieves higher and faster performance in experiments [12]. However, above sub-space methods only contain the global structure and ignore the local structure in original datasets. Local projection remaining model achieves more accurate describe of data [13, 14]. In order to solve the problem that linear method induces high computational complexity and limited expression original information, Fujimaki, etc., rebuild the state space by inducing kernel model [15]. Faschi, etc., achieve the feature extraction model based on regression metric analysis, which is used to find anomaly in cloud platform [16]. However, many of these methods are limited due to all of features from

sub-systems are linked to a long feature vector ignoring the complementary information from difference sub-systems. Moreover, above methods does not consider to optimize the discriminative model to enhance the performance. Finding effective and robust learning models relies on appropriate discriminative model building processing of the anomaly detecting. Deng, etc. utilize SMO network to predict anomaly in Iaas cloud platform obtaining accurate results [17]. Sauvanaud, etc. exploit the dynamic index set both to update anomaly class and to compute the classification of the mean value adapting the changing of data distribution [18]. Fu, etc. first, describe the normal of the system based on Bayesian model; second, confirm the anomaly and obtain labeled samples; finally, semi-supervised model based on decision trees is used to predict anomaly in future [19]. Lan, etc. proposed the automatic recognition method for large scale distributed systems, which is using unsupervised model to detect abnormal nodes increasing the performance [20]. In order to reduce complexity of the algorithm, Wang, etc. find K-nearest nodes based on R-tree index method [21]. Unsupervised models overcome the difficult that samples tag labels by manual and achieve better performance in various of application fields [22-34], but these models high dependent on the distribution of samples. Moreover, it is an important influence how to design the metric method. Therefore, anomaly detecting methods by supervised have gained widespread concern. Beak, etc. tag samples and then utilize classification model to find anomaly [35]. Wang, etc. utilize the entropy model transforming samples into time series to enhance the accuracy [36]. Tan, etc. build on anomaly early-warning system by mixing between the Markov Model and Enhanced Bayesian Networks [37]. Yao, etc. proposed the accompanied detection model based on C4.5 classification model, which is defining both log-primary from all of samples and log-accompanied from abnormal samples. In this method, log-primary are used to train the classification model, and log-accompanied are used to recognize the type of anomalies [38]. Liu, etc. exploit SMO network to achieve automatic recognition and detection [39]. However, above methods may not be sufficiently for the definition of detecting-by-learning. For this reason, first, ignore the complementary information from multiple sub-systems. Moreover, divide into two single steps including extraction feature and classification, which is resulting from the failure supervised information in extraction feature processing. Second, the data distribution is imbalance from the cloud platform, which is reducing the detecting accuracy. Third, it is different from the traditional classification problem of anomaly detecting is the first sequence steps of anomaly handing, and the most suitable handing way is used to approach anomaly ranked. Therefore, the target that the anomaly detecting problem is defined as data associate based on detecting-by-learning is obtain anomaly set and anomaly ranking by learning models. Overview of our algorithm is illustrated in Figure 1. We pose anomaly detecting as a data learning problem, which is solved by multi-view learning model. Our method achieves both automatic fuses multi-view features from multiple sub-systems of the cloud platform and obtains optimized discriminative model by improved extreme learning machine. In order to handle anomaly with dis-

tinguished methods and consider imbalance problem, we proposed the novel method to rank the set of anomaly, and then using ranking results to optimize the classification model to enhance robust under imbalance distribution. The proposed model based on ELM has the following characteristics:

1. We provide an online method to detect anomaly by multi-view learning based on ELM without manual intervention.
2. The proposed model achieves that multi-view features automatic fuse from multiple sub-systems according to supervised information by iterating to minimize the train error, which is exploit the complementary information substantially and obtain the optimal solution space under currently features.
3. Ranking the set of anomaly by proposing the novel model for post-processing, and then generate weight to retrain the classification model to enhance the robustness for imbalance distribution.
4. Through the proposed model by learning, we manage various challenges from high-speed data stream, high-dimension index set, imbalance distribution anomaly, and so on.

For the rest of this paper, we introduce ELM in section 2. In section 3 and section 4, we proposed multi-view model to obtain the set of anomaly, ranking anomaly and optimal the classification model by means of adapting weights from ranking results. In section 5, we utilize collected data from the private cloud platform to evaluate the proposed method and comparing it with existing detecting techniques.

2 Preliminaries: ELM and OSELM

2.1 ELM

In order to facilitate the understanding of our method, this section briefly reviews the related concepts and theories of ELM and developed OSELM.

Extreme learning machine is improved by single hidden layer neural network (SLFNs): assume given N samples (X, T), where $X = [x_1, x_2, ..., x_N]^T \in \mathbb{R}^{d \times N}$, $T = [t_1, t_2, ..., t_N]^T \in \mathbb{R}^{\tilde{N} \times N}$, and $t_i = [t_{i1}, t_{i2}, ..., t_{im}]^T \in \mathbb{R}^m$. The method is used to solve multi-classification problems, and thereby the number of network output nodes is $m(m \geq 2)$. There are $\widetilde{N}$ hidden layer nodes in networks, and activation function $h(\cdot)$ can be Sigmoid or RBF: $\sum_{i=1}^{\widetilde{N}} \beta_i h(a_i x_j + b_j) = o_j$ where $j = 1, \cdots, \tilde{N}$, $a_j = [a_{j1}, a_{j2}, \cdots, a_{jd}]^T$ is the input weight vector, and $\beta_j = [\beta_{j1}, \beta_{j2}, ..., \beta_{jm}]^T$ is the output weight vector. Moreover, a_j, b_j can be generated randomly, which is known by. Written in matrix form: $H\beta = T$, where $H_i = [h_1(a_1 x_1 + b_1), \cdots, h_N(a_{\widetilde{N}} x_1 + b_{\widetilde{N}})]$. Moreover, the solution form of $H\beta = T$ can be written as: $\hat{\beta} = H^{\dagger} T$, where $H^{\dagger}$ is the generalized inverse matrix of H. ELM minimize both the training errors and the output

weights. The expression can be formulated based on optimization of ELM:

$$\text{Minimize} : \frac{1}{2}\|\beta\|_2^2 + C\frac{1}{2}\sum_{i=1}^{N}\|\xi_i\|_2^2$$

$$\text{Subject to} : t_i\beta \cdot h(x_i) \geq 1 - \xi_i, i = 1,...,N \qquad (1)$$

$$\xi_i \geq 0, i = 1,...,N$$

where $\xi_i = \begin{pmatrix} \xi_{i,1} \cdots \xi_{i,m} \end{pmatrix}$ is the vector of the training errors. We can solve the above equation based on KKT theory by Lagrange multiplier, and can obtain the analytical expression of the output weight: $\hat{\beta} = H^T(\frac{I}{C} + HH^T)^{-1}T$. The output function of ELM is: $f(x) = h(x)\hat{\beta} = h(x)H^T(\frac{I}{C} + HH^T)^{-1}T$.

2.2 OSELM

The above model is used to solve classification problem for static batch data. Aiming to this problem Rong et al. proposed an increment classification model OSELM [51]. It is an online solving algorithm based on ELM. The model trains the $\Delta N(\Delta N \geq 1)$ chunk of new samples to obtain new model, then uses matrix calculation with the original model. Through the above calculation, the new output weight matrix $\hat{\beta}_{N+\Delta N}$ is obtained. When the new ΔN chunk arrives, the hidden output weight matrix is updated. The expression is listed as follows:

$$H_{N+\Delta N} = \begin{bmatrix} h(x_1;a_1,b_1) & \cdots & h(x_1;a_{\tilde{N}},b_{\tilde{N}}) \\ \vdots & & \vdots \\ h(x_N;a_1,b_1) & \cdots & h(x_N;a_{\tilde{N}},b_{\tilde{N}}) \\ h(x_{N+1};a_1,b_1) & \cdots & h(x_{N+1};a_{\tilde{N}},b_{\tilde{N}}) \\ \vdots & & \vdots \\ h(x_{N+\Delta N};a_1,b_1) & \cdots & h(x_{N+\Delta N};a_{\tilde{N}},b_{\tilde{N}}) \end{bmatrix} = \begin{bmatrix} h_1 \\ \vdots \\ h_N \\ h_{N+1} \\ \vdots \\ h_{N+\Delta N} \end{bmatrix} = \begin{bmatrix} H_N \\ H_{\Delta N} \end{bmatrix}$$

where $h_{N+k} = \begin{bmatrix} h(x_{N+k};a_1,b_1) \cdots h(x_{N+k};a_{\tilde{N}},b_{\tilde{N}}) \end{bmatrix}^T (k = 1,\ldots,\Delta N)$ is the kth new sample corresponding the vector. Therefore, the output vector is $T_{\Delta N} = \begin{bmatrix} t_{N+1} \cdots t_{N+\Delta N} \end{bmatrix}^T$. Therefore, the incremental expression of the output weight is obtained:

$$\hat{\beta}_{N+\Delta N} = (H_N^T H_N + H_{\Delta N}^T H_{\Delta N})^{-1}(H_N^T T_N + H_{\Delta N}^T T_{\Delta N}) \qquad (2)$$

Let $G_0 = (H_N^T H_N)^{-1}$, and the incremental expression G_1 can be written as:

$$G_1^{-1} = G_0^{-1} + H_{\Delta N}^T H_{\Delta N} \qquad (3)$$

According to the equation (4) and (5), the new output weight matrix $\hat{\beta}_{N+\Delta N}$ becomes:

$$\hat{\beta}_{N+\Delta N} = G_1(H_N T_N + H_{\Delta N}^T T_{\Delta N}) = \beta_N + G_1 H_{\Delta N}^T(T_{\Delta N} - H_{\Delta N}\beta_N) \qquad (4)$$

where $G_1 = (G_0^{-1} + H_{\Delta N}^T H_{\Delta N})^{-1}$.

According to the above equation, we formulate the further expression:

$$G_1 = G_0 - G_0 H_{\Delta N}^T (I_{\Delta N} + H_{\Delta N} G_0 H_{\Delta N}^T)^{-1} H_{\Delta N} G_0 \qquad (5)$$

From the above learning process, OSELM trains new model by adjusting the original model according to the equation (7) when the new dynamic samples are arriving.

3 The proposed anomaly detecting and ranking model

In order to obtain anomaly in real-time from data stream of the cloud computing platform, we proposed an incremental detecting model based on multi-view features, ranking abnormal samples that is prerequisite of anomaly handling generate weights to feedback adjustment the classification model at current to enhance the robustness for imbalance samples of anomaly. The workflow of proposed model is show in Fig. 2., and it divides into two parts: local training and online training. In local training processing, the proposed multi-view features model is used to automatic fuse difference features and achieve optimized discriminative model. In online training processing, first, fuse multiple features according to the local learning structure, which reduce time consuming from retraining all of samples including local and online samples. Second, rank anomalies detected to handle differently. Finally, set self-adapting weights for anomalies that are used to adjust the classification model to avoid the influence of imbalance distribution.

3.1 Multi-view features fusion and discriminative optimization

The number of state indicators from difference subsystems in the cloud computing platform belongs to the range from dozens to hundreds, which is composed the high-dimensional feature space. However, the information and noisy between sub-systems will influence the detected accuracy in subsequent calculations and reduce the performance of detecting. Most traditional models link multiple features into the long vector, and then extract principal components from this vector to avoid information redundancy. It is difficult to mining potential and complementary information from multiple features. Aiming above problem, we automatic fuse multiple features and optimize classification model by iterative solving. The proposed method based on ELM that is used to classify samples. However, when the sample contains various features, the ELM model is difficult applied to solve multiple features of the same sample. In this paper, the the proposed fusion method based on ELM is describe as follow:

Given N the different samples, which contain V features of each sample collected in multiple ways. The feature v corresponds to the samples are: $(x_i^{(v)}, t_i^{(v)})$, where $X^{(v)} = \left[x_1^{(v)}, \ldots, x_N^{(v)} \right]^T \in \mathbb{R}^{D \times N}$, $t_i^{(v)} = \left[t_{i1}^{(v)}, \ldots, t_{im}^{(v)} \right]^T \in \mathbb{R}^m$. Meanwhile, the same sample corresponds to the same class, then there

is: $t_i^{(1)} = t_i^{(2)} = t_i^{(V)}$. The output weight β is solved by using the samples that contain combined multi-view features. The optimization equation is as follows:

$$
\begin{aligned}
Minimize : &\; \tfrac{1}{2} \|\beta\|_2^2 + \sum_{v=1}^{V} k^{(v)} \left(\sum_{j=1}^{N} \left\| \varepsilon_j^{(v)} \right\|_2^2 \right) \\
Subject\ to : &\; \left(k \cdot H^{(v)} \right) \cdot \beta = t_i^T - \left(\varepsilon_i^{(v)} \right)^T \\
&\; \sum_{v=1}^{V} k^{(v)} = 1 \ \ k > 0
\end{aligned}
\tag{6}
$$

where k_j is the combined parameter that corresponds to the single-features. $\xi_i^{(v)} = \left(\xi_{i1}^{(v)}, \ldots, \xi_{im}^{(v)} \right)$ is training error vector that corresponds to feature v. β is the output weight vector for different feature space in the equation (2). According to the equation (2), the target is to obtain the minimum value of the training error that combine the different feature space. However, according to the equation, the solution of $[k_1, k_2]$ may be $(0,1)/(1,0)$. In this situation, it will degenerates to the single feature model, and other features are failed. Therefore, we introduction the high power factor r, and define $r \geq 2$. The equation is improved as follows:

$$
\begin{aligned}
Minimize : &\; \tfrac{1}{2} \|\beta\|_2^2 + \sum_{v=1}^{V} k^{r(v)} \left(\sum_{j=1}^{N} \left\| \varepsilon_j^{(v)} \right\|_2^2 \right) \\
Subject\ to : &\; \left(k \cdot H^{(v)} \right) \cdot \beta = t_i^T - \left(\varepsilon_i^{(v)} \right)^T \\
&\; \sum_{v=1}^{V} k^{r(v)} = 1 \ \ k > 0
\end{aligned}
\tag{7}
$$

Iterative computing is used to solve both fusion coefficient and output weights due to generate interaction between feature fusion and optimization of hidden layer output weights. First, solve initial output weights by uniform fusing multi-view features. Second, testing samples of single feature with the help of solved weights. Finally, adjust the fusion coefficient according to errors of testing, moreover, repeating the firstly step. Therefore, we exploit Lagrange multiplier method, the equation is transformed into:

$$
\begin{aligned}
Minimize : &\; \tfrac{1}{2} \|\beta\|_2^2 + \sum_{v=1}^{V} k^{r(v)} \left(\sum_{j=1}^{N} \left\| \varepsilon_j^{(v)} \right\|_2^2 \right) \\
Subject\ to : &\; \left(k \cdot H^{(v)} \right) \cdot \beta = t_i^T - \left(\varepsilon_i^{(v)} \right)^T \\
&\; \sum_{v=1}^{V} k^{r(v)} = 1 \ \ k > 0
\end{aligned}
\tag{8}
$$

And then, ferivation both variables β, k and Lagrange multipliers a, b, and obtain output weights display expression as follow:

$$Minimize: \frac{1}{2}\|\beta\|_2^2 + \sum_{v=1}^{V} k^{r(v)}\left(\sum_{j=1}^{N}\left\|\varepsilon_j^{(v)}\right\|_2^2\right)$$
$$Subject\ to: \left(k \cdot H^{(v)}\right) \cdot \beta = t_i^T - \left(\varepsilon_i^{(v)}\right)^T \qquad (9)$$
$$\sum_{v=1}^{V} k^{r(v)} = 1 \ k > 0$$

, and the fusion coefficient:

$$Minimize: \frac{1}{2}\|\beta\|_2^2 + \sum_{v=1}^{V} k^{r(v)}\left(\sum_{j=1}^{N}\left\|\varepsilon_j^{(v)}\right\|_2^2\right)$$
$$Subject\ to: \left(k \cdot H^{(v)}\right) \cdot \beta = t_i^T - \left(\varepsilon_i^{(v)}\right)^T \qquad (10)$$
$$\sum_{v=1}^{V} k^{r(v)} = 1 \ k > 0$$

In this paper, we improve above model to incremental detecting method. According to data generation from the cloud computing platform, the training process is divided into offline and online:

Offline training: sampling multi-view features from the cloud platform, training fusion model from accumulating data, and solve both β and k according to equation (9) and (10).

Online: the solved offline coefficient is used to fuse new dataset $x_{\Delta N}$, and obtain the updated output matrix. According to the equation (b), combine with the matrix $H_{\Delta N}$ to solve the incremental output weight matrix as follow:

$$\beta_{N+\Delta N} = S_1(H_N T_N + H_{\Delta N}^T T_{\Delta N}) = \beta_N + S_1 H_{\Delta N}^T (T_{\Delta N} - H_{\Delta N}\beta_N) \quad (11)$$

where $S_1^{-1} = S_0^{-1} + H_{\Delta N}^T H_{\Delta N}$. Therefore, our model achieves anomaly detecting in the cloud computing platform at real-time.

3.2 Anomaly ranking and model adjusting

Anomaly detecting is the precondition of anomaly handling to ensure high efficient running of the cloud computing platform. However, anomalies can be divided into difference degrees according to difference threats. Aiming to this problem, we proposed the novel method to ranking anomalies and the result of ranking is used to adjust the classification model to solve the imbalance problem. The workflow of the proposed model is shown in Fig. 2, including 4 steps: 1) training the classification model by the original training dataset, moreover, obtaining the output hidden weight and the output matrix; 2) according to statistical analysis, we find that it is closely related between the output matrix and the location of sample (verify it in the next paragraph). The classification matrix $T_{n \times c}$ can be obtained from the equation
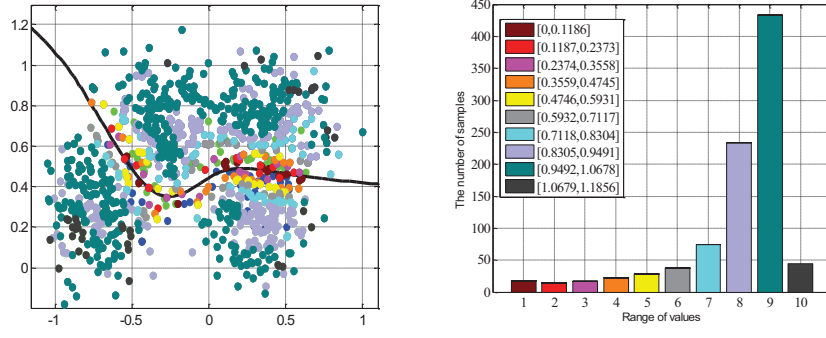
Fig. 1: Statistical analysis of samples distribution

$H_{n\times m}\beta_{m\times c} = T_{n\times c}$, where the category number is the dimension of T, and the maximum value of any row of the matrix $f_max(T_{i\times c})$ is the class of the sample. We exploit the matrix T to describe the location of each sample, achieving the sequence l by ranking $f_max(T_{i\times c})$. The sequence $f_pos(f_max(T_{i\times c}))$ correspond the location sequence, which is smaller value corresponding the closer distance between the samples and the classification boundary; 3) weight samples ranked by $f_max(T_{i\times}c)/\sum_{i=1}^{N} f_max(T_{i\times c})$, due to the small number for abnormal samples, weighted samples can adjusting the adaptability of classification to imbalance distribution; 4) weighted samples are used to retrain the classification model to enhance the robustness.

In order to verify step 2, we obtain the statistics result of all of samples from the testing dataset. First, rank the vector $f_max(T_{i\times c})$; and then, divide the vector $f_max(T_{i\times c})$ into ten ranges according to values, and count the number for each range. The more intuitive is shown in Fig. 3, the right of figure 3 is the histogram that is the samples number from then evenly-distributed ranges, and the left of figure 3 is the scatter-plots that is corresponding the histogram. From the figure 3, we can know locations from three kinds of samples, in 1th type, the number of 20% samples in front of the vector l_head is rather less and samples close to the classification boundary. In 2th type, the number of 20% samples in rear of the vector l_rear is rather less and samples far from the classification boundary. In 3th type, remaining samples that account for the largest proportions and these samples locate in the medium position of the classification region. Therefore, above conclusion that is the vector $f_max(T_{i\times c})$ representing the location of the sample is verified by analysis.

4 Experiments

4.1 Experiments setting

In order to measure the performance of the proposed anomaly detecting method, we deploy the experiment environment based on OpenStack, the framework is

shown in Fig.4. Moreover, the specific is set as the following: (1) The operating system is Ubuntu16.4.01 in the virtual service of the master and services, and we install the virtualization hypervisor KVM+Qemu. (2) The manage component is OpenStack in the cloud computing platform. (3) Performance indicator vectors of virtual services are collected from Prometheus. (4) In order to simulate the working state of each virtual service, Sysbench and Webbench are used benchmarking frameworks. We choose 5 computers as work nodes, and deploy 3 virtual services in each node.

4.2 Dataset collecting and descriping

In this paper, we grab the state data of the virtual service from Metric Agent service to Retrieval in real-time. In order to avoid network congestion and to ensure transmission efficiency of computing data, we collect samples under 10s frequency, and sustained collection 3 hours. Therefore, we obtain the total items is 16200 where offline samples 20% of all samples (3240 items). Online detecting sustains 36 minutes where the number of training samples is 1620 and the number of testing samples is 1620. We collect 4 kinds of state data from virtual services including CPU, memory, disk I/O, network-service. Each item is described as follows ¡time-stamp, host, attribute-set (CPU), attribute-set (memory), attribute-set (disk-I/O), attribute-set (network-service). The specific description of attributes is shown in table 1.

CPU anomaly: run computation programs in virtual services to achieve the very high CPU utilization rate. In this paper, the CPU anomaly is repressed as cpu_Calculation. I/O anomaly: creating, writing, reading a large number of files in virtual services achieve system I/O anomalies. In this paper, the I/O anomaly is repressed as io_Operate. Network anomaly: send a large number of requiring to achieve network anomalies injection and induce the high occupancy rate of network resources of virtual services. In this paper, the Network anomaly is repressed as net_Operate. Memory anomaly: reading/writing the fixed-size memory block to increase the memory load. In this paper, memory anomaly include two classes: memory_Read and memory_Write.

4.3 Experiment results and analysis

Fig.2. is the histogram of sample size, and compares the positive and negative of dataset due to datasets contained anomalies are imbalance distribution. In order to verify the performance of anomaly detecting in cloud computing platform, we utilize ROC curve to visual represent experiment results, therefore, we obtain both (false positive rate, RateFP) and (true positive rate, RateTP) of each sample, the defined as follows:

$$Rate_T P = E_T P / E_P; Rate_F P = E_F P / E_N \tag{12}$$

Table 1: The attributes description (CPU and Memory) of sub-systems

CPU attributes set	Description	Memory attributes set	Description
node_ cpu_idle	Free percentage	node_ memory	Available memory
node_ cpu_iowait	Wating I/O time	node_ memory_Buffers	Block device cache
node_ cpu_softing	Response software interruption time	node_ memory_Cached	Character device cache
node_ cpu_system	Proportion of Kernel Operations	node_ memory_Swapd	Number of Use Spaces
node_ cpu_user	User Process Propo	node_ memory_MemTotal	Total physical memory
node_ cpu_nice	Change process	node_ memory_Memfree	Idle number
node_ cpu_irq	Response to hardware interrupt time	node_ memory_Slab	Kernel uses memory
node_ cpu_cs	Process switching time	node_ memory_Sheme	Process shared memory
node_ cpu_running	Number of Runnable Tasks	node_ memory_VmallocTotal	Virtual Machine Memory Volume
node_ vcpu_run	Virtual Machine Runtime	node_ memory_VmallocRate	Virtual Machine Memory Utilization Rate
node_ cpu_runrate	Virtual Machine Utilization Rate	node_ memory_VmallocMax	AMaximum occupancy of virtual machines

Table 2: The attributes (I/O and Network) description of sub-systems

| I/O attributes set	Description	Network attributes set	Description
node_ disk_await	I/O waiting time	node_ network	The amount of data received per second
node_ disk_svc_time	I/O service time	node_ network_transmit_bytes	The amount of data sended per second
node_ disk_read_time_ms	Number of readings per second	node_ network_receive_packets	Packages received per second
node_ disk_write_time_ms	Number of writing per second	node_ network_transmit_packets	The amount of data sended per second
node_ disk_sectors_written	Reading sector count	node_ network_trLoss_packets	Number of Packets Lost on Acceptance
node_ disk_sectore_written	Writing sector count	node_ network_trLoss_packets	Number of Packets Lost When Sending
node_ disk_io_time_weighted	Percentage of operating time	node_ netstat_TcpExt_TCPOFOQueue	TCP sequence
node_ disk_bytes_read	Reading bit number	node_ netstat_TcpExt_TCPOrigDataSent	TCP traffic
node_ disk_bytes_written	Writing bit number	node_ netstat_TcpExt_TCPLos	TCP untraffic
node_ disk_Vread_time	Number of virtual block reads per second	node_ Vnetwork_receive_bytes	Virtual Network Accepts Data Volume
node_ disk_Vwrite_time	Write times per second for virtual blocks	node_ Vnetwork_transmit_bytes	The amount of data sent by virtual network
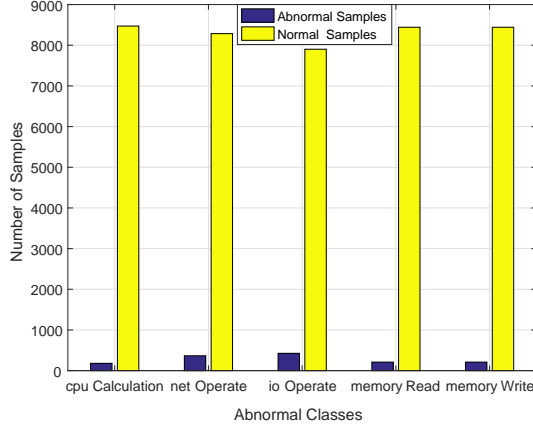
Fig. 2: Please write your figure caption here

where the number of positive is E_P, and the number of negative is E_N. $E_T P$ and $E_T N$ that are the number of classification results are used to compose the coordinate of ROC curve. ROC curve describe the performance by plotting points and linking these points as a curve. When the location of the point is closer to 1, we can know that the classification achieves more performance. In this paper, we choose 3 models to compare and analysis the effectiveness of the proposed model. (1) OrigiF-KNN model, link all features from all subsystems from the cloud computing platform directly, and then KNN model is used to detect anomalies. (2) PCA-KNN model, utilize PCA to reduce the dimension of the linked feature, and then KNN model is used as the classification model. (3) KernelPCA-KNN model, utilize kernel PCA to reduce the dimension of the linked feature, and then KNN model is used as the classification model. Fig.6. shows experiment results including our model and above 3 models, where blue curves are detection results from normal samples, and yellow curves are detection results from anomalies. From figure 7, the proposed method enhance the detecting accuracy due to fully utilizes the complementary information from difference subsystems. Figure 8 is comparison between difference models when cpu_calculation is injected in the cloud computing platform. We can learn that Origi-KNN obtains the unsatisfied result due to the linked feature contains a lot of noisy and redundancy information to distribute the detecting result. KernelPAC-KNN by non-linear mapping obtains higher performance comparing with PCA-KNN. Fig. 9. shows difference detecting results under io_Operate anomalies, our model combines with the supervised information to achieve multi-view features automatic fusion, and optimize the solution space of the classification model to enhance the performance. When net_Operate anomalies are injected in the cloud computing platform, curves of Origi-KNN and KernelPCA-KNN are closer to the diagonal position of ROC, then detecting accuracies of both models are random from figure 9. The proposed model
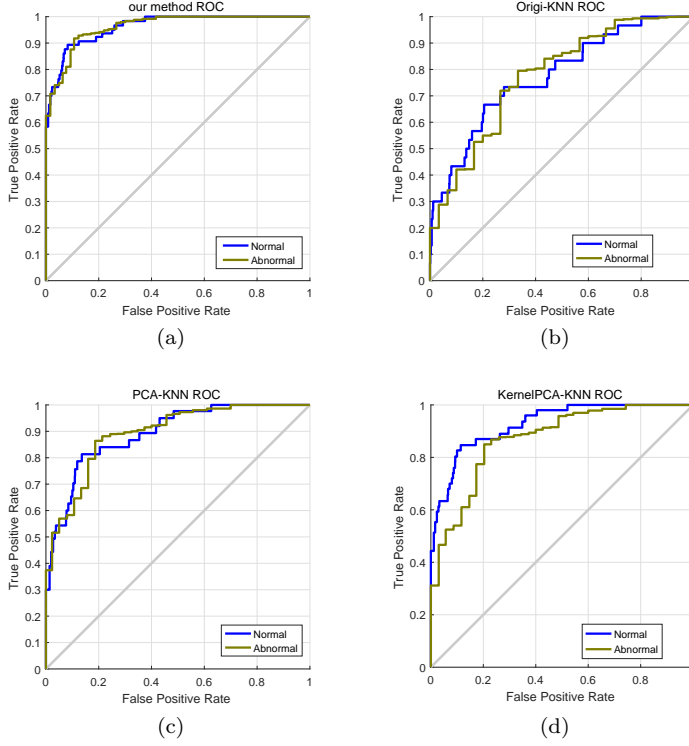
Fig. 3: Comparison of ROC curves of cpu_Calculation anomalies

obtain the satisfied performance. In figure 10, we inject both memory_Read and memory_Write in the cloud computing platform. PCA-KNN, KernelPCA-KNN and our model achieve better results under the kind of anomaly.

5 Conclusion

We presented an effective method for anomaly detecting of the cloud computing platform by building the incremental machine learning model. Our problem is formulated as the binary classification problem in real-time, whose solution is learned through a improve multiple features ELM model. The proposed model automatic fuses multiple features from difference sub-systems and obtains the optimized classification solution by minimizing the training error sum; ranked anomalies are determined by the relation between samples and the classification boundary, and weighting samples ranked retrain the classification model. We can deal with various challenges in anomaly detecting, such as imbalance distribution, high dimensional features and others, effectively through multi-
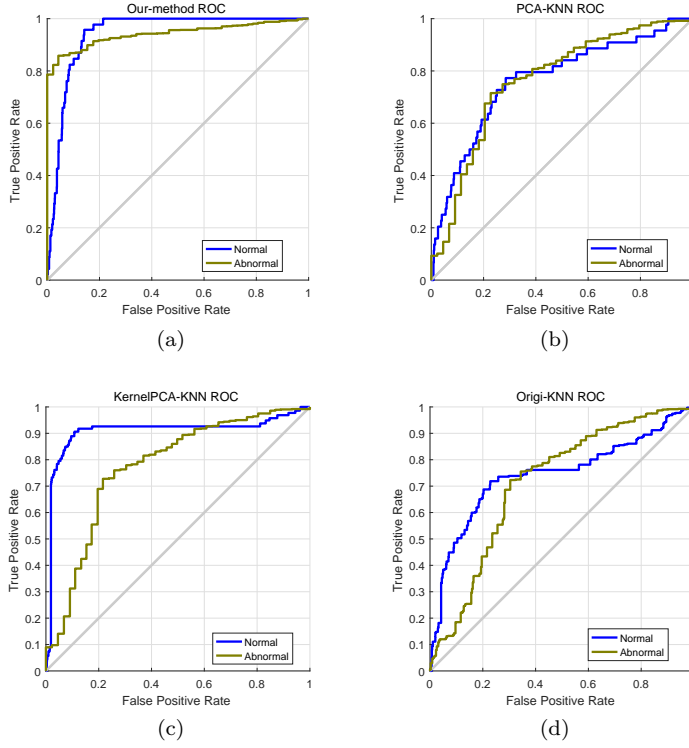
Fig. 4: Comparison of ROC curves of io_Operate anomalies

view learning and feed adjusting. Our model is fast and generalize well to many other sequences from the cloud computing platform.

Acknowledgments

References

1. RITTINGHOUSE J W, RANSOME J F. Cloud computing: implementation, management, and security. Boca Raton: CRC press, 2016.
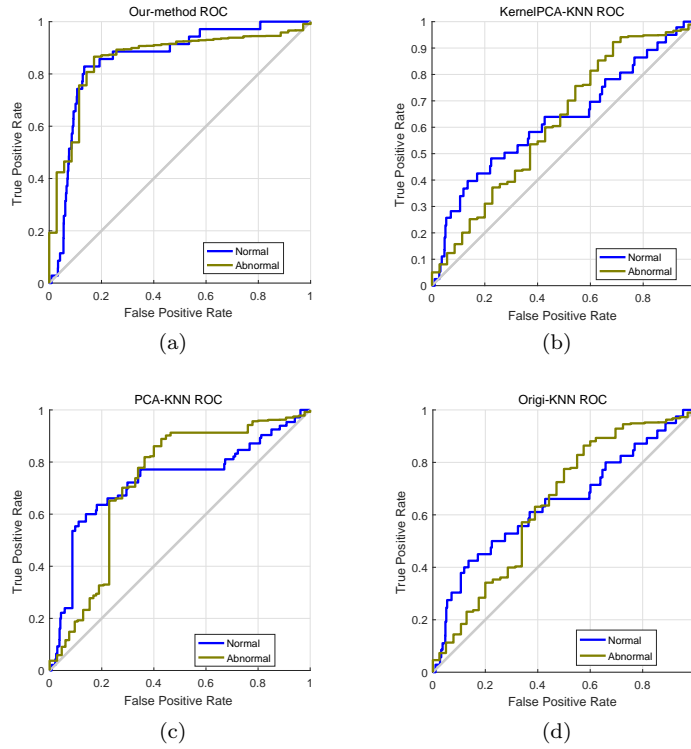
Fig. 5: Comparison of ROC curves of net_Operate anomalies

2. ZHOU ZHEN. Research on Anomaly Detection Strategy and Algorithms Aware of Running Environment for Virtual Machines in the Cloud Platform. Chongqing: Chongqing University. Department of Computer.

3. SALEEM M, RAJOURI J K. Cloud Computing Virtualization. International Journal of Computer Applications Technology and Research, 2017, 6(7): 290-292.

4. VIDAL R, MA Y, SASTRY S S. Principal component analysis. New York: Springer, 2016.

5. WANG R, NIE F, HONG R, et al. Fast and orthogonal locality preserving projections for dimensionality reduction. IEEE Transactions on Image Processing, 2017, 26(10): 5019-5030.

6. WU L, SHEN C, HENGEL A. Deep linear discriminant analysis on fisher networks: A hybrid architecture for person re-identification. Pattern Recognition, 2017, 65: 238-250.

7. DU K L, SWAMY M N S. Independent Component Analysis. London: Springer, 2014.

8. WRIGHT J, GANESH A, RAO S, et al. Robust Principal Component Analysis: Exact Recovery of Corrupted Low-Rank Matrices. Advances in Neural Information Processing Systems, 2009, 87(4):20:3C20:56.

9. G. C. LIU, S. C. YAN. Latent Low-rank Representation for Subspace Segmentation and Feature Extraction // Proc of the IEEE International Conference on Computer Vision. Washington, USA: IEEE, 2011: 1615C1622.

10. GUAN Q, FU S. Adaptive anomaly identification by exploring metric subspace in cloud computing infrastructures // Proc of 2013 IEEE 32nd International Symposium on Reliable Distributed Systems, Washington, USA: IEEE, 2013: 205-214.
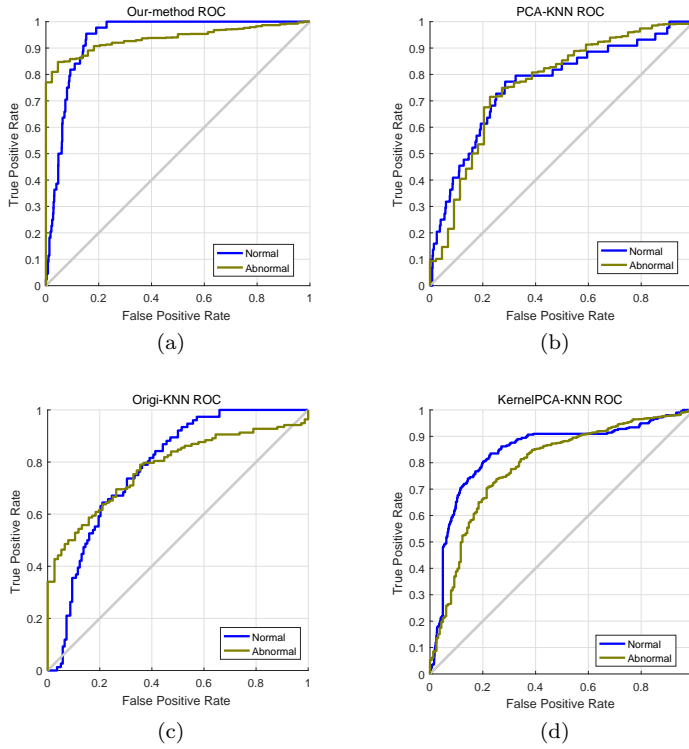
Fig. 6: Comparison of ROC curves of memory_Read anomalies

11. FU S. Performance Metric Selection for Autonomic Anomaly Detection on Cloud Computing Systems // Proc of the Global Telecommunications Conference, Washington, USA: IEEE, 2011: 1-5.

12. LAN Z L, ZHENG Z M, LI Y W. Toward automated anomaly identification in large-scale systems. IEEE Transactions on Parallel and Distributed Systems, 2010, 21(2): 174-187.

13. WARD J L, LUMSDEN S L. Locally linear embedding: dimension reduction of massive protostellar spectra. Monthly Notices of the Royal Astronomical Society, 2016, 461(2): 2250-2256.

14. LIU C, JAJA J, PESSOA L. LEICA: Laplacian eigenmaps for group ICA decomposition of fMRI data[J]. NeuroImage, 2018, 169: 363-373.

15. FUJIMAKI R, YAIRI T, MACHIDA K. An approach to spacecraft anomaly detection problem using kernel feature space // Proc of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining. New York: ACM, 2005: 401-410.

16. FARSHCHI M, SCHNEIDER J G, WEBER I, et al. Metric selection and anomaly detection for cloud operations using log and metric correlation analysis. Journal of Systems and Software, 2017, 137(5):531-549.

17. Dean D J, Nguyen H, Gu X. UBL: Unsupervised behavior learning for predicting performance anomalies in virtualized cloud systems [C]. Proceedings of the 9th ACM International Conference on Autonomic Computing, 2012: 191-200.

18. Sauvanaud C, Silvestre G, Kaaniche M, et al. Data stream clustering for online anomaly detection in cloud applications[C]//Dependable Computing Conference (EDCC), 2015 Eleventh European. IEEE, 2015: 120-131.
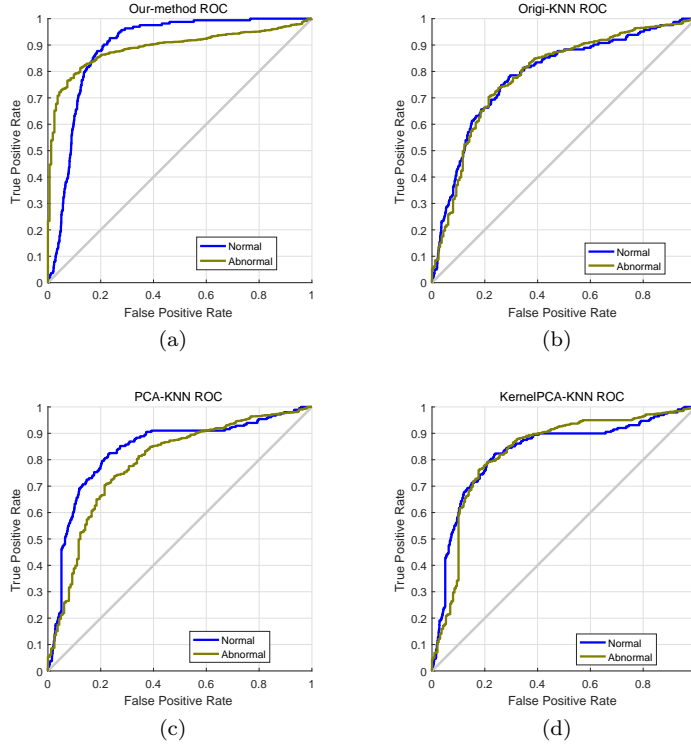
Fig. 7: Comparison of ROC curves of memory_Write anomalies

19. Fu S. Performance metric selection for autonomic anomaly detection on cloud computing systems [C]. 2011 IEEE Global Telecommunications Conference, 2011: 1-5.

20. Lan Z L, Zheng Z M, and Li Y W. Toward automated anomaly identification in large-scal systems [J]. IEEE Transactions on Parallel and Distributed Systems, 2010, 21(2): 174-187.

21. Wang DY. Research and Implementation of Anomaly Detection Technology for Cloud Computin[D]. Shanghai Jiaotong University, 2012.

22. Y. Wang, X. Lin, L. Wu, W. Zhang. Effective Multi-Query Expansions: Collaborative Deep Networks for Robust Landmark Retrieval. IEEE Trans. Image Processing, 26(3):1393-1404, 2017.

23. L. Wu, Y. Wang, L. Shao. Cycle-Consistent Deep Generative Hashing for Cross-Modal Retrieval. IEEE Trans. Image Processing, 28(4):1602-1612, 2019.

24. L. Wu, Y. Wang, L. Shao, M. Wang. 3D PersonVLAD: Learning deep global representations for video-based person re-identification. IEEE Trans. Neural Networks and Learning Systems, 2019.

25. L. Wu, Y. Wang, J. Gao, X. Li. Where-and-When to Look: Deep Siamese Attention Networks for Video-based Person Re-identification. IEEE Trans. Multimedia, 2018.

26. Y. Wang, X. Lin, L. Wu, W. Zhang. Effective Multi-Query Expansions: Robust Landmark Retrieval. ACM Multimedia, 2015.

27. Y. Wang, L. Wu. Beyond low-rank representations: Orthogonal clustering basis reconstruction with optimized graph structure for multi-view spectral clustering. Neural Networks, 103:1-8, 2018.

28. Y. Wang, X. Lin, L. Wu, W. Zhang, Q. Zhang, X. Huang. Robust subspace clustering for multi-view data by exploiting correlation consensus. IEEE Transactions on Image Processing, 24(11):3939-3949,2015.
29. Y. Wang, L. Wu, X. Lin, J. Gao. Multiview spectral clustering via structured low-rank matrix factorization. IEEE Transactions on Neural Networks and Learning Systems, 29(10): 4833-4843, 2018.
30. Y. Wang, W. Zhang, L. Wu, X. Lin, X. Zhao. Unsupervised metric fusion over multiview data by graph random walk-based cross-view diffusion. IEEE Transactions on Neural Networks and Learning Systems, 28(1):57-70, 2017.
31. Y. Wang, W. Zhang, L. Wu, X. Lin, M. Fang and S. Pan. Iterative views agreement: An iterative low-rank based structured optimization method to multi-view spectral clustering. IJCAI 2016: 2153-2159.
32. L. Wu, Y. Wang, X. Li, J. Gao. Deep Attention-based Spatially Recursive Networks for Fine-Grained Visual Recognition. IEEE Transactions on Cybernetics, 2018.
33. L. Wu, Y. Wang, J. Gao, X. Li. Deep Adaptive Feature Embedding with Local Sample Distributions for Person Re-identification. Pattern Recognition, 73:275-288, 2018.
34. L. Wu, Y. Wang, X. Li, J. Gao. What-and-Where to Match: Deep Spatially Multiplicative Integration Networks for Person Re-identification. Pattern Recognition, 76:727-738, 2018.
35. Baek S, Kwon D, Kim J, et al. Unsupervised labeling for supervised anomaly detection in enterprise and cloud networks[C]//Cyber Security and Cloud Computing (CSCloud), 2017 IEEE 4th International Conference on. IEEE, 2017: 205-210.
36. Wang C, Talwar V, Schwan K, Ranganathan P. Online detection of utility cloud anomalies using metric distributions [C]. Proceedings of the 2010 IEEE/IFIP Network Operations and Management Symposium, 2010: 96-103.
37. Tan Y, Nguyen H, Shen Z, Gu X, Venkatramani C, Rajan D. PREPARE: Predictive performance anomaly prevention for virtualized cloud systems[C]. Proceedings of 32nd IEEE International Conference on Distributed Computing Systems, 2012: 285-294.
38. Yao X, Wand HM, etc. Fault Detection Mechanism Based on Adjoint State Tracking in Cloud Computing System[J]. Journal of Computer Science, 2012, 35(5):856-870.
39. Liu J, Chen S, Zhou Z, et al. An Anomaly Detection Algorithm of Cloud Platform Based on Self-Organizing Maps[J]. Mathematical Problems in Engineering, 2016.