Received: 24 August 2018 | Revised: 14 November 2018 | Accepted: 5 January 2019

DOI: 10.4218/etrij.2018-0475

ORIGINAL ARTICLE

ETRI Journal WILEY

UFKLDA: An unsupervised feature extraction algorithm for anomaly detection under cloud environment

GuiPing Wang | **JianXi Yang** | **Ren Li**

College of Information Science and Engineering, Chongqing Jiaotong University, Chongqing, China

Correspondence
GuiPing Wang, College of Information Science and Engineering, Chongqing Jiaotong University, Chongqing, China.
Email: w_guiping@163.com

Funding information
This research is supported by the National Natural Science Foundation of China (Grant No. 61572090), Chongqing Research Program of Basic Research and Frontier Technology (Grant Nos. cstc2015jcyjBX0014 and cstc2016jcyjA0304), and the Scientific and Technological Research Program of Chongqing Municipal Education Commission (Grant Nos. KJ1755492 and KJQN201800705).

In a cloud environment, performance degradation, or even downtime, of virtual machines (VMs) usually appears gradually along with anomalous states of VMs. To better characterize the state of a VM, all possible performance metrics are collected. For such high-dimensional datasets, this article proposes a feature extraction algorithm based on unsupervised fuzzy linear discriminant analysis with kernel (UFKLDA). By introducing the kernel method, UFKLDA can not only effectively deal with non-Gaussian datasets but also implement nonlinear feature extraction. Two sets of experiments were undertaken. In discriminability experiments, this article introduces quantitative criteria to measure discriminability among all classes of samples. The results show that UFKLDA improves discriminability compared with other popular feature extraction algorithms. In detection accuracy experiments, this article computes accuracy measures of an anomaly detection algorithm (i.e., C-SVM) on the original performance metrics and extracted features. The results show that anomaly detection with features extracted by UFKLDA improves the accuracy of detection in terms of sensitivity and specificity.

KEYWORDS
Anomaly detection, cloud computing, feature extraction, kernel method, linear discriminant analysis

1 | INTRODUCTION

A variety of faults may cause performance degradation or even downtime of virtual machines (VMs) under the cloud computing environment. For example, a software fault in a VM may lead to a high CPU utilization rate; a memory leaking in a VM may result in a decrease in the amount of free memory and an increase in the CPU utilization rate; a read or write operation to a malfunctioning physical disk may cause huge I/O time and long CPU idle time. Anomaly detection plays an important role in detecting anomalous VMs before real failures occur, thus improving the dependability [1,2] of the cloud platform.

In literature, a variety of monitoring data, including system logs [3,4], system call [5], and network traffic [6,7], are widely adopted to detect anomalous VMs under the cloud environment. However, the anomalies of VMs are complicated and diversified, especially in the cloud environment. Therefore, it is difficult to accurately detect anomalous VMs from a single kind of monitoring data.

Usually, performance degradation or even downtime of VMs appears gradually along with anomalous consumption of physical or virtual resources (such as CPU, memory, I/O, and network resources). This is reflected in various performance metrics of the subsystems of VMs. To better characterize the state of a VM, all possible performance metrics (dozens or even hundreds) are collected [8,9]. However, much correlation and redundancy exist among performance metrics. Some metrics may even be mixed with measurement noise.

This may reduce the accuracy of anomaly detection. In contrast, high-dimensional datasets increase the complexity of data processing and require higher CPU times. Therefore, the original high-dimensional performance metric dataset should first be transformed into a low-dimensional space, while reserving the most useful information at the same time. This leads to the requirement for feature extraction techniques.

In brief, feature extraction techniques for anomaly detection under the cloud environment face the following challenges:

1. *Unlabeled sample dataset*. For a production cloud platform or a newly deployed one, it is usually difficult to obtain the labels of samples. Therefore, unsupervised techniques are preferred.
2. *Non-Gaussian data*. As stated in the experiments in Section 5, the collected performance metrics data under cloud environment usually do not follow the Gaussian distribution. Therefore, feature extraction algorithms based on the Gaussian distribution assumption, for example, PCA, LDA, UFLDA [10], cannot guarantee excellent performance.
3. *The extracted features should avail anomaly detection*. Feature extraction should not only reduce data dimensionality but also enlarge differences among all classes of samples. That is, the samples within a class after feature extraction should assemble together as close as possible, whereas the samples that belong to different classes should separate as far as possible to avail subsequent anomaly detection. Therefore, to quantitatively evaluate the performance of feature extraction techniques, quantitative criteria should be introduced to measure the discriminability among all classes of samples.

To overcome the above challenges, this article proposes a feature extraction algorithm based on unsupervised fuzzy linear discriminant analysis with kernel (termed UFKLDA). The main contributions of this article are listed as follows: 1) It proposes a more effective unsupervised feature extraction algorithm, that is, UFKLDA, and proves its correctness. 2) It introduces quantitative criteria to measure the discriminability among all classes of samples. And 3) it presents two sets of experiments conducted on artificial and real datasets to prove the effectiveness of UFKLDA.

2 | RELATED WORKS

The dependability of a system is defined as the comprehensive abilities of the system to deliver service(s) that can justifiably be trusted [1]. To understand the dependability of cloud platforms, Guan et al [8] investigate the impact of virtualization on cloud dependability. They propose a cloud dependability analysis (CDA) framework with mechanisms to characterize the failure behavior in cloud computing infrastructures.

VMs are an important carrier for cloud services. They are isolated from one another in the cloud platform. Deliberate or nondeliberate faults, including hardware faults [11], software defects [12], misconfiguration [13], and attacks [14], may lead to performance degradation or even downtime of VMs, thereby lowering the dependability of the cloud platform.

Detecting anomalous states of VMs before failures occur is important to improve the dependability of cloud platforms. Generally, to better characterize and understand the state of a system, all possible state variables (i.e., performance metrics) are collected. For distributed systems, it is necessary to collect performance metrics from all subsystems (CPU, memory, I/O, and network) per physical or virtual node [15]. For example, 518 performance metrics in total are collected every minute in [8].

However, the most important performance metrics are usually unknown in practice, or they cannot be measured directly. Therefore, the first issue before anomaly detection is to extract the most effective features from the original performance metrics. Feature extraction also reduces the dimensionality of datasets and, therefore, accelerates data processing.

Davis and Clark [16] review data preprocessing techniques in anomaly-based network intrusion detection. They refer to feature extraction as feature construction, which aims to create additional features with a better discriminative ability than that of the initial performance metric set. However, they do not define or introduce quantitative criteria to measure such discriminative ability.

In large-scale cloud systems, component failures are a norm rather than an exception [17]. Smith et al. [17] collect health-related performance metrics of every node to detect anomalous nodes. These metrics are collected from different layers, including hardware, hypervisor, operating system, and application. A feature extraction technique based on Bayesian network and principal components analysis (PCA) are used to extract the main features that affect the health state of each node. However, they do not verify the effectiveness of the adopted feature extraction technique compared with other popular techniques in literature.

Lan et al. [15] address anomaly identification in large-scale systems. They introduce independent component analysis (ICA)-based feature extraction to convert multidimensional performance metric data into a lower dimensional space for quick and better analysis. One limitation of their study is that the number of independent components in ICA is set based on a criterion used by PCA, which may discard some useful information in the data.

For anomaly detection in hyperspectral imagery, Zhao et al. [18] propose a kernel-based ICA algorithm, which

combines kernel principal component analysis (KPCA) and ICA. Kernel ICA implements nonlinear feature extraction and makes the extracted features mutually independent. However, the intention of introducing the kernel method is not clarified, and the effectiveness of kernel ICA is not verified.

In a production cloud computing system, the state of a physical or virtual node is usually not absolutely normal or abnormal; that is, the boundary between normal and abnormal states is not obvious. Therefore, in the case of unlabeled sample datasets, the ideas in fuzzy mathematics can be introduced into linear discriminant analysis (LDA), which is a supervised algorithm. Li et al [10] extend LDA to the unsupervised situation. They propose a clustering algorithm, that is, fuzzy linear discriminant clustering (FLDC), as well as an unsupervised feature extraction algorithm, namely, unsupervised fuzzy linear discriminant analysis (UFLDA). However, UFLDA works well only when the samples in a cluster show a Gaussian-like distribution.

To improve the accuracy of anomaly-based intrusion detection, Gan et al. [19] proposed a combined algorithm based on partial least square (PLS) feature extraction and core vector machine (CVM) algorithms. However, they do not verify the effectiveness of the combined algorithm compared with other popular techniques.

For kernel-based feature extraction methods, Zamani et al. [20] discuss the choice of the kernel function and its parameters considering the classification information and error for the mapping features. They propose kernel principal components analysis (KPCA) using a nonlinear combination of kernels based on genetic programming and the classification error fitness function. Through experiments, they verify that the proposed KPCA outperforms conventional KPCA in terms of clustering validity indices and classification accuracy.

Nonlinear feature extraction algorithms, such as manifold learning [21], wavelet [22], principal curves [23], genetic programming [24], and artificial neural networks [25], are also studied in literature. The principles of these nonlinear algorithms are usually complicated. Additionally, these algorithms are usually difficult to implement. Therefore, this article will not focus on these nonlinear algorithms. Notwithstanding all this, through introducing the kernel method and choosing kernel functions, the proposed UFKLDA can also implement nonlinear feature extraction.

Compared with the above studies, this article aims at introducing kernel methods to improve the insufficiency of UFLDA [10] in dealing with non-Gaussian sample data, therefore proposing UFKLDA and proving its correctness. In addition, this article defines quantitative criteria to measure discriminative ability among all classes of samples, and presents experiments conducted on artificial datasets and real datasets to verify the effectiveness of UFKLDA.

3 | REVIEW OF FEATURE EXTRACTION ALGORITHMS

3.1 | Preliminaries and notations

Definition 1 Performance metric: For a physical or virtual node, any individually measurable variable (e.g., CPU utilization rate, available memory size, I/O time, network traffic) is defined as a performance metric.

Definition 2 Feature: For a dataset containing multiple performance metrics, each dimensionality of the data after feature extraction is called a feature.

Definition 3 Feature extraction: Feature extraction transforms a dataset containing multiple performance metrics into a lower dimensional space with fewer features to implement faster data analysis.

Despite fewer extracted features, the information most relevant to the problem is reserved after feature extraction. Moreover, the data presented in a low-dimensional subspace are easier to be separated into different classes [15]. Therefore, feature extraction can improve the accuracy of anomaly detection.

The following definition is based on definitions in literature [1,15].

Definition 4 Anomaly and anomaly detection: An anomaly refers to the state of a detected system (or a node in the system) that deviates from the expected normal state of the system (or most of other nodes in the system). Anomaly detection is the function of detecting anomalous states of a system or anomalous nodes in a system.

To distinguish different symbols, this article follows the notations as given below:

1. An italic lowercase letter (with or without subscript(s)) (e.g., x, x_{ij}) represents a scalar value.
2. A bold and italic lowercase letter (with or without subscript(s)) (e.g., $\boldsymbol{x}$, $\boldsymbol{x}_i$) represents a vector. All vectors in this article are column vectors. In addition, a vector in the Hilbert space is represented as a bold lowercase letter (e.g. $\mathbf{x}$).
3. A bold and italic uppercase letter (e.g., $\boldsymbol{X}$) represents a matrix. A matrix in the Hilbert space is represented as a bold uppercase letter (e.g., $\mathbf{X}$).
4. An italic uppercase letter (e.g., X) represents a random variable.
5. A bold and italic uppercase letter (e.g., $\boldsymbol{X}$) represents a random vector. Random vectors are also column vectors.

Assume that an observed VM has n metrics, X_i, $i = 1, \ldots, n$. Each metric can be considered as a random variable. The n metrics constitute a random vector $\boldsymbol{X}$. Further, assume that a total of l samples of all VMs in a monitoring domain are obtained in a certain time period. These l samples

constitute an n-by-l original sample matrix, $\boldsymbol{X}_{n\times l}$, as shown in (1).

$$\boldsymbol{X}=\begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{bmatrix}, \quad \boldsymbol{X}_{n\times l}=\begin{bmatrix} x_{11} & x_{12} & x_{13} & \cdots & x_{1l} \\ x_{21} & x_{22} & x_{23} & \cdots & x_{2l} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & x_{n3} & \cdots & x_{nl} \\ \uparrow & \uparrow & \uparrow & & \uparrow \\ \boldsymbol{x}_1 & \boldsymbol{x}_2 & \boldsymbol{x}_3 & \cdots & \boldsymbol{x}_l \end{bmatrix}. \quad (1)$$

In $\boldsymbol{X}_{n\times l}$, each row represents all the sample data of a performance metric X_i and each column represents the sampling values of all performance metrics of a VM in a point-in-time, that is, $\boldsymbol{x}_i$ is a sample of $\boldsymbol{X}$, $i = 1, \ldots, l$.

3.2 | Linear discriminant analysis

The study of LDA dates back to Fisher's classic article published in 1936 [26]. Mika et al. [27] first introduce the kernel method into LDA and propose kernel LDA (KLDA). The basic idea of LDA is to search for the vectors maximizing the Fisher criterion function (i.e., (7)) as the best projection directions. The projection on these best directions obtains maximum between-cluster scatter and minimum within-cluster scatter to make the samples within a class assemble together as close as possible, whereas the samples that belong to different classes separate as far as possible at the same time.

Assuming that the l samples can be divided into c classes, $H_i = \{x_1^{(i)}, \ldots, x_{l_i}^{(i)}\} \subset R^n$ is the subset of ith class samples, l_i is the number of samples in ith class, $i = 1, ..., c$, $l = l_1 + l_2 + \cdots + l_c$. The within-cluster scatter matrix ($\boldsymbol{S}_\mathrm{W}$) is defined as:

$$\boldsymbol{S}_\mathrm{W}=\sum_{i=1}^{c}\boldsymbol{S}_i=\sum_{i=1}^{c}\sum_{\mathbf{x}\in H_i}(\boldsymbol{x}-\boldsymbol{m}_i)(\boldsymbol{x}-\boldsymbol{m}_i)^T, \quad (2)$$

where

$$\boldsymbol{S}_i=\sum_{\boldsymbol{x}\in H_i}(\boldsymbol{x}-\boldsymbol{m}_i)(\boldsymbol{x}-\boldsymbol{m}_i)^T, \quad \boldsymbol{m}_i=\frac{1}{l_i}\sum_{\boldsymbol{x}\in H_i}\boldsymbol{x}, \quad i=1,2,\ldots,c, \quad (3)$$

are the within-cluster scatter matrix and mean of the ith class, respectively.

The between-cluster scatter matrix ($\boldsymbol{S}_\mathrm{B}$) is defined as:

$$\boldsymbol{S}_\mathrm{B}=\sum_{i=1}^{c}l_i(\boldsymbol{m}_i-\boldsymbol{m})(\boldsymbol{m}_i-\boldsymbol{m})^T. \quad (4)$$

where

$$\boldsymbol{m}=\frac{1}{l}\sum_{\boldsymbol{x}}\boldsymbol{x}=\frac{1}{l}\sum_{i=1}^{c}l_i\boldsymbol{m}_i, \quad (5)$$

is the mean of all samples.

The total scatter matrix ($\boldsymbol{S}_\mathrm{T}$) is defined as the sum of $\boldsymbol{S}_\mathrm{W}$ and $\boldsymbol{S}_\mathrm{B}$, that is,

$$\boldsymbol{S}_\mathrm{T}=\boldsymbol{S}_\mathrm{W}+\boldsymbol{S}_\mathrm{B}. \quad (6)$$

Based on matrices $\boldsymbol{S}_\mathrm{W}$ and $\boldsymbol{S}_\mathrm{B}$, the Fisher criterion function is defined as

$$J(\boldsymbol{w})=\frac{\boldsymbol{w}^T\boldsymbol{S}_\mathrm{B}\boldsymbol{w}}{\boldsymbol{w}^T\boldsymbol{S}_\mathrm{W}\boldsymbol{w}}. \quad (7)$$

The problem of searching projection directions in LDA is converted into solving the following eigenvalue problem.

$$\boldsymbol{S}_\mathrm{B}\boldsymbol{w}_i=\lambda_i\boldsymbol{S}_\mathrm{W}\boldsymbol{w}_i, \quad i=1,2,\ldots,s. \quad (8)$$

The obtained eigenvectors, $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_s$, constitute transformation matrix $\boldsymbol{P}_{s\times n}$.

3.3 | Unsupervised fuzzy linear discriminant analysis

Based on the suggestion that the division into clusters should be characterized by within-cluster similarity and between-cluster (external) dissimilarity [28], Li et al. propose an unsupervised feature extraction algorithm, namely, UFLDA [10].

The unsupervised within-cluster and between-cluster scatter matrices are defined as

$$\boldsymbol{S}_\mathrm{W}^\mathrm{U}=\sum_{i=1}^{c}\sum_{j=1}^{l}\left[u_{ij}(\boldsymbol{x}_j-\boldsymbol{m}_i)(\boldsymbol{x}_j-\boldsymbol{m}_i)^T\right], \quad (9)$$

and

$$\boldsymbol{S}_\mathrm{B}^\mathrm{U}=\sum_{i=1}^{c}\left[\left(\sum_{j=1}^{l}u_{ij}\right)(\boldsymbol{m}_i-\boldsymbol{m})(\boldsymbol{m}_i-\boldsymbol{m})^T\right], \quad (10)$$

where

$$\boldsymbol{m}_i=\sum_{j=1}^{l}\left(\frac{u_{ij}}{\sum_{k=1}^{l}u_{ik}}\right)\boldsymbol{x}_j, \quad \mu_{ij}=\frac{u_{ij}}{\sum_{k=1}^{l}u_{ik}}. \quad (11)$$

u_{ij} is the membership indicating the degree of sample $\boldsymbol{x}_j$ belonging to the ith class. u_{ij} meets the following constraints:

$$\sum_{i=1}^{c}u_{ij}=1, \quad j=1,\ldots,l, \qquad u_{ij}\in[0,1], \quad i=1,\ldots,c;\ j=1,\ldots,l. \quad (12)$$

μ_{ij} indicates the ratio of u_{ij} to the total memberships of all samples belonging to the ith class.

As the theorem in [10] shows, scatter matrices of LDA ((2) and (4)) are the special cases of the unsupervised scatter matrices of UFLDA ((9) and (10)). That is to say that in the supervised situation, $\boldsymbol{S}_{\mathrm{B}}^{\mathrm{U}}$ and $\boldsymbol{S}_{\mathrm{W}}^{U}$ are the same as $\boldsymbol{S}_{\mathrm{B}}$ and $\boldsymbol{S}_{\mathrm{W}}$, respectively. The problem of searching projection directions in UFLDA is ultimately converted into solving the eigenvalue problem $(\boldsymbol{S}_{\mathrm{rW}}^{\mathrm{U}})^{-1}\boldsymbol{S}_{\mathrm{B}}^{\mathrm{U}}\boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i$ where $\boldsymbol{S}_{\mathrm{rW}}^{\mathrm{U}}$ is a regularization of $S_{\mathrm{W}}^{\mathrm{U}}$.

4 | THE PROPOSED UFKLDA ALGORITHM

4.1 | The principles of UFKLDA

As stated in Section 2, UFLDA works well only when the distribution of clusters shows a Gaussian-like distribution. This is because UFLDA is derived from LDA, and LDA is based on the Gaussian distribution assumption. UFLDA did not improve LDA on this point. Aiming at solving the problem of non-Gaussian data, this article introduces the kernel method into UFLDA and proposes UFKLDA. By introducing the kernel method, UFKLDA can not only effectively deal with non-Gaussian datasets but also implement nonlinear feature extraction.

The basic idea of UFKLDA is (i) first, by using a nonlinear map, Φ, it transforms the sample data $\boldsymbol{x}$ from the input space R^n to the Hilbert space $\mathcal{H}$ so as to make $\Phi(\boldsymbol{x})$ approximately follow the Gaussian distribution; because the Gaussian kernel function is usually chosen, the problem is converted into choosing the only parameter, λ; (ii) next, it uses UFLDA to extract features from mapped sample data. Because $\mathcal{H}$ may be a high-dimensional or even an infinite-dimensional space, the kernel method should be introduced so as to compute the kernel function directly without knowing the map function Φ. In other words, UFKLDA does not need to really map the samples into the $\mathcal{H}$ space.

Let $\{\Phi(\boldsymbol{x}_1), \ldots, \Phi(\boldsymbol{x}_l)\}$ be the mapped sample set; assume $K(\boldsymbol{x}, \boldsymbol{x}')$ be the kernel function associated with map Φ. In Hilbert space $\mathcal{H}$, the between-cluster scatter matrix and within-cluster scatter matrix are defined as:

$$\mathbf{S}_{\mathrm{B}}^{\mathrm{U}\Phi} = \sum_{i=1}^{c}\left[\left(\sum_{j=1}^{l} u_{ij}\right)\left(\mathbf{m}_i^{\Phi} - \mathbf{m}^{\Phi}\right)\left(\mathbf{m}_i^{\Phi} - \mathbf{m}^{\Phi}\right)^T\right], \quad (13)$$

$$\mathbf{S}_{\mathrm{W}}^{\mathrm{U}\Phi} = \sum_{i=1}^{c}\sum_{j=1}^{l}\left[u_{ij}\left(\Phi(\boldsymbol{x}_j) - \mathbf{m}_i^{\Phi}\right)\left(\Phi(\boldsymbol{x}_j) - \mathbf{m}_i^{\Phi}\right)^T\right], \quad (14)$$

where $\mathbf{m}_i^{\Phi} = \sum_{j=1}^{l}\mu_{ij}\Phi(\boldsymbol{x}_j)$ is the mean of ith class of mapped samples and $\mathbf{m}^{\Phi} = \frac{1}{l}\sum_{j=1}^{l}\Phi(\boldsymbol{x}_j)$ is the mean of all mapped samples.

The Fisher criterion of UFKLDA is

$$J^{\Phi}(\mathbf{w}) = \frac{\mathbf{w}^T\mathbf{S}_{\mathrm{B}}^{\mathrm{U}\Phi}\mathbf{w}}{\mathbf{w}^T\mathbf{S}_{\mathrm{W}}^{\mathrm{U}\Phi}\mathbf{w}}. \quad (15)$$

According to the theory of reproducing kernels, any solution $\mathbf{w}$ can be represented as the linear combination of all mapped samples [27], that is,

$$\mathbf{w} = \sum_{i=1}^{l} a_i\Phi(\boldsymbol{x}_i), \quad a_i \in R. \quad (16)$$

The following theorem shows that the Fisher criterion of UFKLDA ((15)) can be transformed into the expression of weight vector $\boldsymbol{\alpha}$, where

$$\boldsymbol{\alpha} = [a_1\ a_2\ \cdots\ a_l]^T. \quad (17)$$

Theorem 1 *The Fisher criterion of UFKLDA can be transformed into*

$$J^{\Phi}(\boldsymbol{\alpha}) = \frac{\boldsymbol{\alpha}^T\mathbf{T}_{\mathrm{B}}^{\mathrm{U}\Phi}\boldsymbol{\alpha}}{\boldsymbol{\alpha}^T\mathbf{T}_{\mathrm{W}}^{\mathrm{U}\Phi}\boldsymbol{\alpha}}, \quad (18)$$

where

$$\mathbf{T}_{\mathrm{B}}^{\mathrm{U}\Phi} = \sum_{i=1}^{c}\left(\sum_{j=1}^{l} u_{ij}\right)(\mathbf{M}_i^{\Phi} - \mathbf{M}^{\Phi})(\mathbf{M}_i^{\Phi} - \mathbf{M}^{\Phi})^T, \quad (19)$$

$$\mathbf{M}_i^{\Phi} = \left[\begin{array}{ccc}\sum_{j=1}^{l}\mu_{1j}K(\boldsymbol{x}_1,\boldsymbol{x}_j) & \cdots & \sum_{j=1}^{l}\mu_{nj}K(\boldsymbol{x}_l,\boldsymbol{x}_j)\end{array}\right]^T, \quad (20)$$

$$\mathbf{M}^{\Phi} = \left[\begin{array}{ccc}\frac{1}{l}\sum_{j=1}^{l}K(\boldsymbol{x}_1,\boldsymbol{x}_j) & \cdots & \frac{1}{l}\sum_{j=1}^{l}K(\boldsymbol{x}_l,\boldsymbol{x}_j)\end{array}\right]^T, \quad (21)$$

$$\mathbf{T}_{\mathrm{W}}^{\mathrm{U}\Phi} = \sum_{i=1}^{c}\sum_{j=1}^{l} u_{ij}(\mathbf{K}_j^{\Phi} - \mathbf{M}_i^{\Phi})(\mathbf{K}_j^{\Phi} - \mathbf{M}_i^{\Phi})^T, \quad (22)$$

$$\mathbf{K}_j^{\Phi} = [\ K(\boldsymbol{x}_1,\boldsymbol{x}_j) \quad K(\boldsymbol{x}_2,\boldsymbol{x}_j) \quad \cdots \quad K(\boldsymbol{x}_l,\boldsymbol{x}_j)\ ]^T. \quad (23)$$

Proof First, to represent the numerator of (18) with $\boldsymbol{\alpha}$, the key operations are to represent $\mathbf{w}^T\mathbf{m}_i^{\Phi}$ and $\mathbf{w}^T\mathbf{m}^{\Phi}$ with $\boldsymbol{\alpha}$.

Because

$$\mathbf{w}^T\mathbf{m}_i^{\Phi} = \left(\sum_{i=1}^{l} a_i\Phi(\boldsymbol{x}_i)\cdot\sum_{j=1}^{l}\mu_{ij}\Phi(\mathbf{x}_j)\right) = \sum_{i=1}^{l}\sum_{j=1}^{l} a_i\mu_{ij}K(\boldsymbol{x}_i,\boldsymbol{x}_j)$$

according to (17) and (20), equation $\mathbf{w}^T\mathbf{m}_i^{\Phi} = \alpha^T\mathbf{M}_i^{\Phi}$ is established.

Similarly, because

$$\mathbf{w}^T\mathbf{m}^{\Phi} = \left(\sum_{i=1}^{l} a_i\Phi(\boldsymbol{x}_i)\cdot\frac{1}{l}\sum_{j=1}^{l}\Phi(\boldsymbol{x}_j)\right) = \frac{1}{l}\sum_{i=1}^{l}\sum_{j=1}^{l} a_iK(\boldsymbol{x}_i,\boldsymbol{x}_j)$$

according to (17) and (21), equation $\mathbf{w}^T\mathbf{m}^{\Phi} = \boldsymbol{\alpha}^T\mathbf{M}^{\Phi}$ is established. Therefore, the numerator of (18) can be transformed as follows.

$$\begin{aligned}
&\mathbf{w}^T\mathbf{S}_{\mathrm{B}}^{\mathrm{U}\Phi}\mathbf{w}\\
&= \sum_{i=1}^{c}\left[\left(\sum_{j=1}^{l}u_{ij}\right)\mathbf{w}^T(\mathbf{m}_i^{\Phi}-\mathbf{m}^{\Phi})(\mathbf{w}^T(\mathbf{m}_i^{\Phi}-\mathbf{m}^{\Phi}))^T\right]\\
&= \sum_{i=1}^{c}\left[\left(\sum_{j=1}^{l}u_{ij}\right)\boldsymbol{\alpha}^T(\mathbf{M}_i^{\Phi}-\mathbf{M}^{\Phi})(\boldsymbol{\alpha}^T(\mathbf{M}_i^{\Phi}-\mathbf{M}^{\Phi}))^T\right]\\
&= \sum_{i=1}^{c}\left[\left(\sum_{j=1}^{l}u_{ij}\right)\boldsymbol{\alpha}^T(\mathbf{M}_i^{\Phi}-\mathbf{M}^{\Phi})(\mathbf{M}_i^{\Phi}-\mathbf{M}^{\Phi})^T\boldsymbol{\alpha}\right]\\
&= \boldsymbol{\alpha}^T\mathbf{T}_B^{U\Phi}\boldsymbol{\alpha}.
\end{aligned}$$

The denominator of (18) also can be transformed into the expression of $\boldsymbol{\alpha}$ according to the above methods; therefore, equation $\mathbf{w}^T\mathbf{S}_{\mathrm{W}}^{\mathrm{U}\Phi}\mathbf{w} = \boldsymbol{\alpha}^T\mathbf{T}_{\mathrm{W}}^{\mathrm{U}\Phi}\boldsymbol{\alpha}$ is established.

According to the above theorem, the problem of solving projection vector $\mathbf{w}$ can be transformed as solving the weight vector $\boldsymbol{\alpha}$, where the latter can be further transformed as solving the eigenvectors of matrix $(\mathbf{T}_{\mathrm{W}}^{\mathrm{U}\Phi})^{-1}\mathbf{T}_{\mathrm{B}}^{\mathrm{U}\Phi}$.

In practice, matrix $\mathbf{T}_{\mathrm{W}}^{\mathrm{U}\Phi}$ is regularized by $\mathbf{T}_{\mathrm{rW}}^{\mathrm{U}\Phi} = r\mathbf{T}_{\mathrm{W}}^{\mathrm{U}\Phi} + (1-r)\mathrm{diag}\left(\mathbf{T}_{\mathrm{W}}^{\mathrm{U}\Phi}\right)$, where $\mathrm{diag}(\mathbf{T}_{\mathrm{W}}^{\mathrm{U}\Phi})$ is the diagonal part of matrix $\mathbf{T}_{\mathrm{W}}^{\mathrm{U}\Phi}$ and $r\in[0, 1]$ is a regularization parameter. Finally, the original problem is transformed into solving the eigenvectors of matrix $(\mathbf{T}_{\mathrm{rW}}^{\mathrm{U}\Phi})^{-1}\mathbf{T}_{\mathrm{B}}^{\mathrm{U}\Phi}$.

4.2 | The proposed UFKLDA

The first key step in UFKLDA is to find the optimal matrix $\boldsymbol{U}$ under the criterion $\max_{\boldsymbol{U}}\mathrm{tr}\left[\left(\mathbf{T}_{\mathrm{rW}}^{\mathrm{U}\Phi}\right)^{-1}\mathbf{T}_{\mathrm{B}}^{\mathrm{U}\Phi}\right]$ by using the interior-point optimization method, as detailed in [10]. $\boldsymbol{U}$ is a $c \times l$ matrix that contains all the membership degrees. The second step is to solve the eigenvectors of matrix $(\mathbf{T}_{\mathrm{rW}}^{\mathrm{U}\Phi})^{-1}\mathbf{T}_{\mathrm{B}}^{\mathrm{U}\Phi}$, thereby constructing the transformation matrix with former s eigenvectors. The concrete steps of UFKLDA are listed as follows.

Algorithm *UFKLDA*

Input: l unlabeled n-dimensional samples, $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, ..., $\boldsymbol{x}_l$; the number of extracted features, s.

Output: The transformation matrix $\boldsymbol{P}_{s\times n}$; the sample matrix after feature extraction $\boldsymbol{Y}_{s\times l}$.

Step 1: Initialize the membership matrix $\boldsymbol{U} = [u_{ij}]_{1\le i\le c, 1\le j\le l}$ with random values from [0, 1] such that element u_{ij} of $\boldsymbol{U}$ satisfies $\sum_{i=1}^{c} u_{ij} = 1, j = 1, \ldots, l$.

Step 2: Choose kernel function $K(\boldsymbol{x}, \boldsymbol{x}')$.

Step 3: Take $\max_{\boldsymbol{U}}\mathrm{tr}[(\mathbf{T}_{\mathrm{rW}}^{\mathrm{U}\Phi})^{-1}\mathbf{T}_{\mathrm{B}}^{\mathrm{U}\Phi}]$ as the objective function, use the interior-point optimization method to find the optimal $\boldsymbol{U}$ that satisfies

$$\sum_{i=1}^{c} u_{ij} = 1,\ j = 1,\ \ldots, l \quad \text{and}$$
$$u_{ij}\in[0,1],\ i = 1, \ldots, c;\ j = 1,\ \ldots, l.$$

Step 4: Compute $\mathbf{T}_{\mathrm{B}}^{\mathrm{U}\Phi}$ and $\mathbf{T}_{\mathrm{W}}^{\mathrm{U}\Phi}$ with the optimal $\boldsymbol{U}$ obtained in Step 3 and regularize $\mathbf{T}_{\mathrm{W}}^{\mathrm{U}\Phi}$ as $\mathbf{T}_{\mathrm{rW}}^{\mathrm{U}\Phi}$.

Step 5: Solve the eigenvector problem $(\mathbf{T}_{\mathrm{rW}}^{\mathrm{U}\Phi})^{-1}\mathbf{T}_{\mathrm{B}}^{\mathrm{U}\Phi}\boldsymbol{w}_i = \lambda_i\boldsymbol{w}_i$ with $\lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_s$.

Step 6: Let the transformation matrix be $\boldsymbol{P}^T = [\boldsymbol{w}_1\ \boldsymbol{w}_2\ \ldots\ \boldsymbol{w}_s]$.

Step 7: Transform the sample matrix $\boldsymbol{X}_{n\times l}$ from n-dimensional space to s-dimensional space by the transformation $\boldsymbol{Y}_{s\times l} = \boldsymbol{P}_{s\times n} \times \boldsymbol{X}_{n\times l}$ and obtain $\boldsymbol{Y}_{s\times l}$.

5 | EXPERIMENTS AND ANALYSES

This section first introduces artificial datasets and real datasets adopted in the experiments (Section 5.1). Then, it describes the experimental methods (Section 5.2). Finally, it presents two sets of experiments and corresponding analyses (Section 5.3).

5.1 | Datasets

5.1.1 | Artificial datasets

The advantage of artificial datasets is that it is possible to generate random data following specified probability

distribution according to the requirement of experiments. The research team of Professor Ludmila Kuncheva presents a framework for generating data to simulate changing environments [29] and provides several artificial datasets [30]. These datasets are widely used in literature for testing algorithms in feature selection, feature extraction, classification, and clustering. The scatter diagrams of these datasets are shown in Figure 1A to 1H. All these datasets have a certain degree of nonlinear structure. Therefore, they are very challenging for linear feature extraction algorithms.

5.1.2 | Real datasets

In the cloud environment, the performance or state of VMs can be characterized by performance metrics. This article collects samples of VMs from an organization-wide cloud platform consisting of 60 physical computing nodes. 0–4 VMs are deployed on each node. This number is dynamically changed according to the deployment assigned by the management server. This article collects 53 performance metrics, which indicate the heath state of a VM. These performance metrics can be classified into five categories: computation, storage, disk I/O, process, and network. The histograms of four optionally chosen performance metrics are given in Figure 2. It is verified that the performance metrics usually do not follow the Gaussian distribution.

In addition, to construct real datasets including abnormal samples and normal ones, four types of anomalies are injected to 0–30 randomly selected VMs in the cloud platform, which are listed as follows:

1. Anomaly in computation resource consumption (ComAnomaly): Besides the normal computation threads on the selected VMs, the extra injected threads compete for the CPU resource with the normal threads.
2. Anomaly in memory resource consumption (MemAnomaly): Besides the normal threads, the extra injected threads generate memory leaking, which continue consuming memory without releasing it.
3. Anomaly in disk I/O operation (DiskAnomaly): Besides the normal threads, the extra injected I/O intensive threads keep reading and writing a large number of bytes from local disks.
4. Anomaly in network access (NetAnomaly): Several users run LoadRunner from their own physical personal computers to simultaneously access a web application server deployed on a randomly selected VM, generating a large number of HTTP connections to simulate anomalous network behavior.

Based on the above methods, the required real datasets are constructed for experiments.

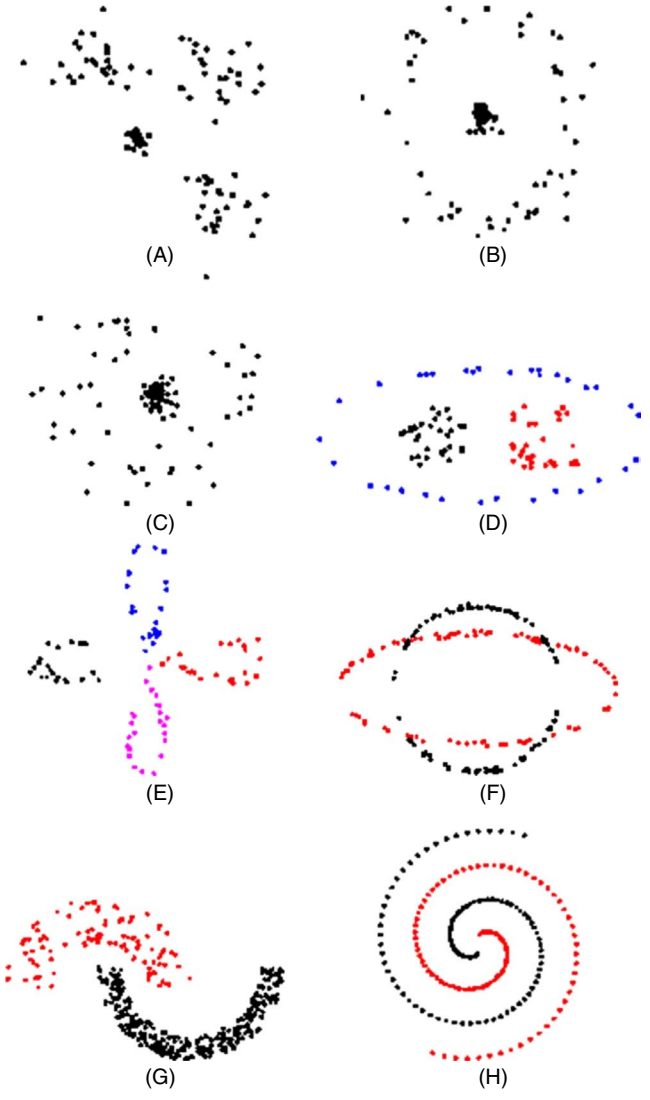

FIGURE 1 Eight artificial datasets: (A) four-Gauss, (B) easy doughnut, (C) difficult doughnut, (D) boat, (E) petals, (F) saturn, (G) half-ring, and (H) two spirals

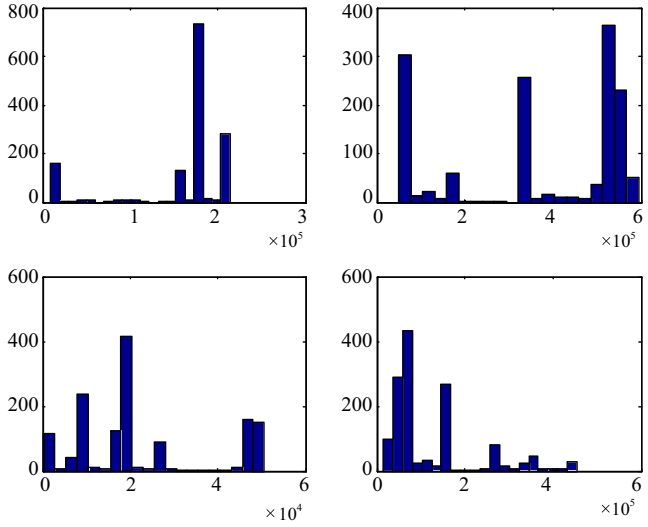

FIGURE 2 Histograms of four performance metrics

5.2 | Experimental methods

It is necessary for feature extraction algorithms to ensure a high dimensionality reduction rate. If the number of extracted features is specified to be the same value for different algorithms, the reduction rates are the same. Another target of feature extraction algorithms is to extract a set of most effective features for classification (e.g., anomaly detection) from multiple performance metrics. This may achieve good discriminability among all classes of samples characterized by these features. Therefore, first it is necessary to define quantitative criteria to measure the discriminability among all classes of samples.

5.2.1 | Quantitative criteria

By reference to the idea of total, between-cluster, and within-cluster scatter matrices (i.e., $\boldsymbol{S}_{\mathrm{T}}$, $\boldsymbol{S}_{\mathrm{B}}$, and $\boldsymbol{S}_{\mathrm{W}}$) in LDA, this article defines the following three square distances:

$d(\boldsymbol{X})$: total average square distance,
$d_{\mathrm{W}}(\boldsymbol{X})$: within-cluster average square distance, and
$d_{\mathrm{B}}(\boldsymbol{X})$: between-cluster average square distance.

For the sample matrix before feature extraction, $\boldsymbol{X}_{n\times l}$, let $\boldsymbol{x}_k^{(i)} \in H_i$, $\boldsymbol{x}_t^{(j)} \in H_j$ be the samples of the ith and jth classes, respectively, $\delta(\boldsymbol{x}_k^{(i)}, \boldsymbol{x}_t^{(j)})$ be the square distance between these two samples, p_i, p_j be the priori probabilities of these two classes; then $d(\boldsymbol{X})$ is defined as:

$$d(\boldsymbol{X}) = \frac{1}{2}\sum_{i=1}^{c} p_i \sum_{j=1}^{c} p_j \frac{1}{l_i l_j} \sum_{k=1}^{l_i}\sum_{t=1}^{l_j} \delta\left(\boldsymbol{x}_k^{(i)}, \boldsymbol{x}_t^{(j)}\right). \tag{24}$$

The Euclidean distance is applied in this article, therefore:

$$\begin{aligned}\delta(\boldsymbol{x}_k^{(i)}, \boldsymbol{x}_t^{(j)}) &= \| \boldsymbol{x}_k^{(i)} - \boldsymbol{x}_t^{(j)} \|^2 \\ &= (\boldsymbol{x}_k^{(i)} - \boldsymbol{x}_t^{(j)})^T (\boldsymbol{x}_k^{(i)} - \boldsymbol{x}_t^{(j)}),\end{aligned} \tag{25}$$

where $\boldsymbol{x}_k^{(i)}$and $\boldsymbol{x}_t^{(j)}$ are both n-dimensional column vectors.

In addition, for an n-dimensional column vector $\boldsymbol{x}$, the following equation holds:

$$\| \boldsymbol{x} \|^2 = \boldsymbol{x}^T \boldsymbol{x} = \mathrm{tr}(\boldsymbol{x}\boldsymbol{x}^T). \tag{26}$$

The following theorem shows that $d(\boldsymbol{X})$ can be computed as the trace of $\boldsymbol{S}_{\mathrm{T}}$.

Theorem 2 *$d(\boldsymbol{X})$ equals to the trace of $\boldsymbol{S}_T$, that is,*

$$d(\boldsymbol{X}) = \mathrm{tr}(\boldsymbol{S}_{\mathrm{T}}) = \mathrm{tr}(\boldsymbol{S}_{\mathrm{W}} + \boldsymbol{S}_{\mathrm{B}}). \tag{27}$$

Proof $(\boldsymbol{x}_k^{(i)} - \boldsymbol{x}_t^{(j)})^T (\boldsymbol{x}_k^{(i)} - \boldsymbol{x}_t^{(j)})$ can be transformed into

$$\begin{aligned}&\left[\left(\boldsymbol{x}_k^{(i)} - \boldsymbol{m}_i\right) - \left(\boldsymbol{x}_t^{(j)} - \boldsymbol{m}_j\right) + \left(\boldsymbol{m}_i - \boldsymbol{m}\right) - \left(\boldsymbol{m}_j - \boldsymbol{m}\right)\right]^T \\ &\times \left[\left(\boldsymbol{x}_k^{(i)} - \boldsymbol{m}_i\right) - \left(\boldsymbol{x}_t^{(j)} - \boldsymbol{m}_j\right) + \left(\boldsymbol{m}_i - \boldsymbol{m}\right) - \left(\boldsymbol{m}_j - \boldsymbol{m}\right)\right].\end{aligned}$$

After expansion, 16 terms are obtained, but only 6 terms are nonzero. After merging these 6 terms, $d(\boldsymbol{X})$ can be converted into 5 terms.

$$d(\boldsymbol{X}) = \frac{1}{2l^2}\sum_{i=1}^{c}\sum_{j=1}^{c}\begin{bmatrix} l_j \sum_{k=1}^{l_i} (\boldsymbol{x}_k^{(i)} - \boldsymbol{m}_i)^T (\boldsymbol{x}_k^{(i)} - \boldsymbol{m}_i) \\ + l_i \sum_{t=1}^{l_j} (\boldsymbol{x}_t^{(j)} - \boldsymbol{m}_j)^T (\boldsymbol{x}_t^{(j)} - \boldsymbol{m}_j) \\ + l_i l_j (\boldsymbol{m}_i - \boldsymbol{m})^T (\boldsymbol{m}_i - \boldsymbol{m}) \\ - 2 l_i l_j (\boldsymbol{m}_i - \boldsymbol{m})^T (\boldsymbol{m}_j - \boldsymbol{m}) \\ + l_i l_j (\boldsymbol{m}_j - \boldsymbol{m})^T (\boldsymbol{m}_j - \boldsymbol{m}) \end{bmatrix}.$$

Note that the third term is zero and the first and second terms are equal to

$$\frac{1}{2l}\sum_{i=1}^{c}\sum_{k=1}^{l_i}\left(\boldsymbol{x}_k^{(i)} - \boldsymbol{m}_i\right)^T \left(\boldsymbol{x}_k^{(i)} - \boldsymbol{m}_i\right).$$

The fourth and fifth terms are equal to

$$\frac{1}{2l}\sum_{i=1}^{c} l_i (\boldsymbol{m}_i - \boldsymbol{m})^T (\boldsymbol{m}_i - \boldsymbol{m}).$$

Therefore,

$$\begin{aligned} d(\boldsymbol{X}) &= \frac{1}{l}\sum_{i=1}^{c}\sum_{k=1}^{l_i}\left(\boldsymbol{x}_k^{(i)} - \boldsymbol{m}_i\right)^T \left(\boldsymbol{x}_k^{(i)} - \boldsymbol{m}_i\right) + \frac{1}{l}\sum_{i=1}^{c} l_i \left(\boldsymbol{m}_i - \boldsymbol{m}\right)^T \left(\boldsymbol{m}_i - \boldsymbol{m}\right) \\ &= \frac{1}{l}\sum_{i=1}^{c}\sum_{k=1}^{l_i}\mathrm{tr}\left[\left(\boldsymbol{x}_k^{(i)} - \boldsymbol{m}_i\right)\left(\boldsymbol{x}_k^{(i)} - \boldsymbol{m}_i\right)^T\right] + \frac{1}{l}\sum_{i=1}^{c} l_i \mathrm{tr}\left[(\boldsymbol{m}_i - \boldsymbol{m})(\boldsymbol{m}_i - \boldsymbol{m})^T\right] \\ &= \mathrm{tr}\left[\frac{1}{l}\sum_{i=1}^{c}\sum_{k=1}^{l_i}\left(\boldsymbol{x}_k^{(i)} - \boldsymbol{m}_i\right)\left(\boldsymbol{x}_k^{(i)} - \boldsymbol{m}_i\right)^T\right] + \mathrm{tr}\left[\frac{1}{l}\sum_{i=1}^{c} l_i (\boldsymbol{m}_i - m)(\boldsymbol{m}_i - \boldsymbol{m})^T\right] \\ &= \mathrm{tr}(\boldsymbol{S}_{\mathrm{W}}) + \mathrm{tr}(\boldsymbol{S}_{\mathrm{B}}) = \mathrm{tr}\left(\boldsymbol{S}_{\mathrm{W}} + \boldsymbol{S}_{\mathrm{B}}\right). \end{aligned}$$

■

By reference to the definition of $d(\boldsymbol{X})$, $d_{\mathrm{W}}(\boldsymbol{X})$ is defined as

$$d_{\mathrm{W}}(\boldsymbol{X}) = \frac{1}{2}\sum_{i=1}^{c} p_i \frac{1}{l_i^2}\sum_{k=1}^{l_i}\sum_{t=1}^{l_i}\delta\left(\boldsymbol{x}_k^{(i)}, \boldsymbol{x}_t^{(i)}\right). \tag{28}$$

Similarly, the following equation can be proved.

$$d_{\mathrm{W}}(\boldsymbol{X}) = \mathrm{tr}\left(\boldsymbol{S}_{\mathrm{W}}\right). \tag{29}$$

Therefore, $d_{\mathrm{B}}(\boldsymbol{X})$ can be computed as

$$d_{\mathrm{B}}(\boldsymbol{X}) = d(\boldsymbol{X}) - d_{\mathrm{W}}(\boldsymbol{X}). \tag{30}$$

For the sample matrix after feature extraction, $\boldsymbol{Y}_{s\times l}$, the square distances $d(\boldsymbol{Y})$, $d_{\mathrm{W}}(\boldsymbol{Y})$, and $d_{\mathrm{B}}(\boldsymbol{Y})$ can also be computed in a similar way.

For a given feature extraction algorithm, if the number of extracted features is specified, the smaller the $d_{\mathrm{W}}(\boldsymbol{Y})$, the more closely the projected samples assemble together in each class and the larger the $d_{\mathrm{B}}(\boldsymbol{Y})$, the farther the projected samples that belong to different classes separate. Therefore, the better the discriminability of the projected samples, the better the performance of the feature extraction algorithm.

5.2.2 | Experimental methods

Two sets of experiments are conducted to verify the effectiveness of UFKLDA: discriminability experiments and detection accuracy experiments.

The first set of experiments compares the discriminability abilities of four unsupervised feature extraction algorithms (PCA, UFLDA, UFKLDA, and KICA) on artificial and real datasets. For uniformity, the number of extracted features is the same for all the involved feature extraction algorithms for each dataset. The values of the quantitative criteria (i.e., $d_{\mathrm{W}}(\boldsymbol{Y})$ and $d_{\mathrm{B}}(\boldsymbol{Y})$) achieved by each algorithm are computed and listed to intuitively compare the performances of different algorithms.

The second set of experiments computes and compares the accuracy measures of a same anomaly detection algorithm (i.e., *C*-SVM [31,32]) on the original performance metrics (i.e., no extraction) of real datasets and on the features extracted by the above four feature extraction algorithms. A single kind of anomaly or combination of two kinds of anomalies is injected to randomly selected VMs in the cloud platform to collect and construct datasets. This article adopts the following two accuracy measures to evaluate the performances of these combined anomaly detection mechanisms: (F_{P}, False Positive; F_{N}, False Negative; T_{P}, True Positive; T_{N}, True Negative).

1. *Sensitivity*: The proportion of True Positive (i.e., correctly detected anomalous VMs) to the number of actual anomalous VMs.

$$\mathrm{Sensitivity} = T_{\mathrm{P}}/(T_{\mathrm{P}} + F_{\mathrm{N}}). \tag{31}$$

2. *Specificity*: The proportion of True Negative (i.e., correctly detected normal VMs) to the number of actual normal VMs.

$$\mathrm{Specificity} = T_{\mathrm{N}}/(F_{\mathrm{P}} + T_{\mathrm{N}}). \tag{32}$$

5.3 | Experimental results and analyses

5.3.1 | Discriminability experiments

Two real cloud datasets are generated, each containing 500 and 1,000 samples, respectively. The proportions of all kinds of samples (i.e., normal state and four types of anomalies) are the same on the whole in each dataset. Table 1 lists the experimental results of four unsupervised feature extraction algorithms on these two cloud datasets. The number of extracted features is specified as 16. Therefore, the dimensionality reduction rates of these four algorithms are $(53 - 16)/53 = 69.81\%$. Each cell in the latter four columns contains two values; the former is $d_{\mathrm{W}}(\boldsymbol{Y})$ and the latter is $d_{\mathrm{B}}(\boldsymbol{Y})$. The results in Table 1 show that, among these four algorithms, the performances of PCA and UFLDA are the worst; KICA performs better than these two algorithms; whereas UFKLDA works the best among these four algorithms. Specifically, the achieved $d_{W}(\boldsymbol{Y})$ value of UFKLDA in each group of experiment is the smallest, which means the samples in each class represented by the extracted features assemble most closely, whereas the $d_{\mathrm{B}}(\boldsymbol{Y})$ value is the biggest, which means the distance among each class of samples is the farthest. Therefore, UFKLDA achieves the best discriminability.

The results of these four unsupervised feature extraction algorithms on eight artificial datasets are listed in Table 2. The results show that for the Gaussian-like distribution datasets (i.e., the first five datasets), the performance of UFLDA is evidently superior to that of PCA; whereas, for the non-Gaussian distribution datasets (i.e., the latter three ones),

TABLE 1 Experimental results of four unsupervised feature extraction algorithms on two real cloud datasets

No.	No. of metrics	No. of features	PCA ($d_{\mathrm{W}}(\boldsymbol{Y})$, $d_{\mathrm{B}}(\boldsymbol{Y})$)	UFLDA ($d_{\mathrm{W}}(\boldsymbol{Y})$, $d_{\mathrm{B}}(\boldsymbol{Y})$)	UFKLDA ($d_{\mathrm{W}}(\boldsymbol{Y})$, $d_{\mathrm{B}}(\boldsymbol{Y})$)	KICA ($d_{\mathrm{W}}(\boldsymbol{Y})$, $d_{\mathrm{B}}(\boldsymbol{Y})$)
1	53	16	(26.5577, 7.7328)	(25.4137, 8.0145)	(24.1529, 8.5319)	(24.9137, 8.3823)
2	53	16	(29.0449, 8.1540)	(28.1227, 8.4016)	(27.1729, 8.8115)	(27.8329, 8.5171)

there is no obvious advantage for UFLDA. However, the proposed UFKLDA works better than the other three algorithms on all the eight datasets: its $d_W(\boldsymbol{Y})$ values are the smallest, and its $d_B(\boldsymbol{Y})$ values are the largest; for the first three datasets containing 12 performance metrics, extracting only 1 feature also can assure good discriminability. Especially for non-Gaussian distribution datasets, UFKLDA performs obviously better than UFLDA.

5.3.2 | Detection accuracy experiments

Note that feature extraction algorithms not only reduce data dimensionality and data volume but also enlarge the difference among each class of samples represented by the extracted features. Moreover, they reduce or even eliminate the correlation and redundancy among features. Therefore, feature extraction algorithms are expected to enhance the accuracy of anomaly detection algorithms.

In this set of experiments, all the four feature extraction algorithms extract 16 features from 53 performance metrics of the real datasets. Table 3 lists the experimental results of four single-anomaly tests. Each cell in the last five columns contains two values, the first being sensitivity and the second being specificity. Each experiment is conducted 10 times, and the results are averaged. The results of all experiments are also averaged in the bottommost row. Although the original performance metric set contains all the information for anomaly detection, much correlation and redundancy exist among the performance metrics. Some metrics may even be mixed with measurement noise. Therefore, anomaly detection with all original performance metrics (i.e., no extraction) is not the best choice, which is verified in Table 3. The results also show that, compared with the features extracted by PCA, UFLDA, and KICA, anomaly detection with features extracted by UFKLDA achieves the highest detection accuracy in terms of sensitivity and specificity.

Table 4 lists the experimental results of six combination-anomaly tests. The bottommost row averages the results from all experiments. Compared with single-anomaly tests, the detection accuracies of six combination-anomaly tests are relatively low. This is because the most important features are more difficult to extract under the conditions of combination anomaly. In spite of this, UFKLDA plus SVM still achieves the highest detection accuracy among the five anomaly detection mechanisms.

Form the above two sets of experiments, it is concluded that the proposed UFKLDA outperforms the other three algorithms. The main underlying reasons are listed as follows:

1. UFKLDA inherits the advantage of LDA, that is, the solved projection directions are effective for classification.
2. By introducing kernel methods, UFKLDA not only overcomes the insufficiency of UFLDA in dealing with non-Gaussian sample data but also can implement nonlinear feature extraction.

6 | CONCLUSIONS AND FUTURE WORK

To overcome the insufficiency of UFLDA in dealing with non-Gaussian sample data, this article introduces the kernel method and proposes a feature extraction algorithm based on unsupervised fuzzy linear discriminant analysis with kernel (termed UFKLDA). Experiments on both artificial and real datasets prove the effectiveness of UFKLDA.

In summary, UFKLDA successfully addresses the three challenges stated in Section 1. First, UFKLDA is an unsupervised feature extraction algorithm and, therefore, applicable for unlabeled sample datasets. Second, UFKLDA improves the performance of UFLDA in dealing with non-Gaussian sample data by introducing kernel methods. Third, the experimental results show that UFKLDA achieves the best discriminability among the involved algorithms. The future work of this article will focus on some popular nonlinear feature extraction algorithms. A

TABLE 2 Experimental results of four unsupervised feature extraction algorithms on eight artificial datasets

No.	No. of metrics	No. of features	PCA ($d_W(Y)$, $d_B(Y)$)	UFLDA ($d_W(Y)$, $d_B(Y)$)	UFKLDA ($d_W(Y)$, $d_B(Y)$)	KICA ($d_W(Y)$, $d_B(Y)$)
1	12	1	(0.7115, 0.9307)	(0.6363, 0.9912)	(0.5991, 1.1375)	(0.6193, 1.0549)
2	12	1	(1.6765, 0.0415)	(1.4327, 0.2193)	(1.2916, 0.3961)	(1.3291, 0.3196)
3	12	1	(1.6890, 0.0213)	(1.4217, 0.3007)	(1.2035, 0.6113)	(1.3009, 0.4578)
4	2	1	(0.8666, 0.1543)	(0.7595, 0.4331)	(0.6152, 0.6129)	(0.6630, 0.5451)
5	2	1	(0.4831, 0.8102)	(0.3993, 0.8391)	(0.3121, 0.9765)	(0.3429, 0.9163)
6	2	1	(1.0078, 0.0037)	(0.9911, 0.0149)	(0.6929, 0.4026)	(0.8327, 0.2175)
7	2	1	(0.4989, 0.9335)	(0.4803, 0.9461)	(0.3095, 1.0121)	(0.4017, 0.9326)
8	2	1	(1.0031, 0.0708)	(0.9809, 0.0915)	(0.7127, 0.3723)	(0.8391, 0.2532)

TABLE 3 Experimental results of detection accuracy of four single-anomaly tests

	No extraction + SVM (*Sens.*, *Spec.*)	PCA + SVM (*Sens.*, *Spec.*)	UFLDA + SVM (*Sens.*, *Spec.*)	UFKLDA + SVM (*Sens.*, *Spec.*)	KICA + SVM (*Sens.*, *Spec.*)
ComAnomaly	(0.961, 0.960)	(0.940, 0.940)	(0.968, 0.966)	(0.976, 0.972)	(0.960, 0.952)
MemAnomaly	(0.973, 0.968)	(0.943, 0.938)	(0.972, 0.967)	(0.981, 0.975)	(0.965, 0.961)
DiskAnomaly	(0.972, 0.969)	(0.939, 0.932)	(0.969, 0.965)	(0.978, 0.971)	(0.953, 0.951)
NetAnomaly	(0.952, 0.945)	(0.928, 0.925)	(0.957, 0.951)	(0.966, 0.963)	(0.947, 0.939)
Average	(0.965, 0.961)	(0.938, 0.934)	(0.967, 0.962)	(0.975, 0.970)	(0.956, 0.951)

TABLE 4 Experimental results of detection accuracy of six combination-anomaly tests

	No extraction + SVM (*Sens.*, *Spec.*)	PCA + SVM (*Sens.*, *Spec.*)	UFLDA + SVM (*Sens.*, *Spec.*)	UFKLDA + SVM (*Sens.*, *Spec.*)	KICA + SVM (*Sens.*, *Spec.*)
ComAnomaly & MemAnomaly	(0.929, 0.921)	(0.902, 0.883)	(0.931, 0.925)	(0.942, 0.937)	(0.919, 0.913)
ComAnomaly & DiskAnomaly	(0.941, 0.933)	(0.913, 0.893)	(0.940, 0.937)	(0.948, 0.941)	(0.932, 0.931)
ComAnomaly & NetAnomaly	(0.947, 0.936)	(0.921, 0.913)	(0.941, 0.939)	(0.953, 0.945)	(0.927, 0.922)
MemAnomaly & DiskAnomaly	(0.939, 0.933)	(0.912, 0.907)	(0.942, 0.942)	(0.945, 0.942)	(0.937, 0.932)
MemAnomaly & NetAnomaly	(0.937, 0.932)	(0.905, 0.900)	(0.937, 0.936)	(0.946, 0.940)	(0.933, 0.925)
DiskAnomaly & NetAnomaly	(0.929, 0.922)	(0.897, 0.893)	(0.938, 0.931)	(0.943, 0.940)	(0.929, 0.921)
Average	(0.937, 0.930)	(0.908, 0.898)	(0.938, 0.935)	(0.946, 0.941)	(0.930, 0.924)

comparison between UFKLDA and these nonlinear algorithms is expected in future.

ACKNOWLEDGMENTS

The authors are grateful to the editor and anonymous reviewers for their valuable comments on this article.

ORCID

GuiPing Wang https://orcid.org/0000-0001-7184-3302

REFERENCES

1. A. Avizienis, et al., *Basic concepts and taxonomy of dependable and secure computing*, IEEE Trans. Dependable Secure Comput. **1** (2004), no. 1, 11–33.
2. W. Abderrahim and Z. Choukair, *The three-dimensional model for dependability integration in cloud computing*, Ann. Telecommun. **72** (2017), no. 5-6, 371–384.
3. H. B. Mi et al., *Toward fine-grained, unsupervised, scalable performance diagnosis for production cloud computing systems*, IEEE Trans. Parallel Distrib. Syst. **24** (2013), no. 6, 1245–1255.
4. A. N. Harutyunyan et al., *Abnormality analysis of streamed log data*, in Proc. IEEE/IFIP Netw. Oper. Manag. Symp.: Manag. Softw. Defined World, Krakow, Poland, May 2014, pp. 1–7.
5. S. Alarifi and S. Wolthusen, Anomaly detection for ephemeral cloud IaaS virtual machines, in *Proc. Int. Conf. Netw. Syst.* (NSS), Madrid, Spain, June 3–4, 2013, pp. 321–335.
6. K. Adamova et al., *Network anomaly detection in the cloud: The challenges of virtual service migration*, in Proc. IEEE Int. Conf. Commun. (ICC), Sydney, Australia, June 2014, pp. 3770–3775.
7. B. L. Dalmazo et al., *Expedite feature extraction for enhanced cloud anomaly detection*, in Proc. IEEE/IFIP Netw. Oper. Manag. Symp. (NOMS), Istanbul, Turkey, Apr. 2016, pp. 1215–1220.
8. Q. Guan, C. C. Chiu, and S. Fu, , in *CDA: A cloud dependability analysis framework for characterizing system dependability in cloud computing*Proc. IEEE Pacific Rim Int. Symp. Dependable Comput. (PRDC), Niigata, Japan, Nov. 2012, pp. 11–20.
9. Q. Guan and S. Fu, *Auto-AID: A data mining framework for autonomic anomaly identification in networked computer systems*, in Proc. IEEE Int. Performance Comput. Commun. Conf. (IPCCC), Albuquerque, NM, USA, Dec. 2010, pp. 73–80.
10. C.-H. Li, B.-C. Kuo, and C.-T. Lin, *LDA-based clustering algorithm and its application to an unsupervised feature extraction*, IEEE Trans. Fuzzy Syst. **19** (2011), no. 1, 152–163.
11. R. Kumar, S. Vijayakumar, and S. A. Ahamed, *A pragmatic approach to predict hardware failures in storage systems using MPP database and big data technologies*, in Proc. IEEE Int. Adv. Comput. Conf., Gurgaon, India, Feb. 2014, pp. 779–788.
12. F. Langner and A. Andrzejak, *Detecting software aging in a cloud computing framework by comparing development versions*, in Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage., Ghent, Belgium, May 2013, pp. 896–899.
13. S. Kikuchi and K. Hiraishi, *Improving reliability in management of cloud computing infrastructure by formal methods*, in Proc. IEEE/IFIP Netw. Oper. Manag. Symp.: Manag. Softw. Defined World, Krakow, Poland, May 2014, pp. 1–7.
14. S. N. Brohi et al., *Identifying and analyzing security threats to virtualized cloud computing infrastructures*, in Proc. Int. Conf. Cloud Comput. Technol., Applicat. Manag., Dubai, United Arab Emirates, Dec. 2012, pp. 151–155.

15. Z. L. Lan, Z. M. Zheng, and Y. W. Li, *Toward automated anomaly identification in large-scale systems*, IEEE Trans. Parallel Distrib. Syst. **21** (2010), no. 2, 174–187.
16. J. J. Davis and A. J. Clark, *Data preprocessing for anomaly based network intrusion detection: A review*, Comp. Secur. **30** (2011), no. 6-7, 353–375.
17. D. Smith, Q. Guan, and S. Fu, *An anomaly detection framework for autonomic management of compute cloud systems*, in Proc. Annu. IEEE Int. Comput. Softw. Applicat. Conf. Workshops, Seoul, Rep. of Korea, July 2010, pp. 376–381.
18. C. H. Zhao, Y. L. Wang, and F. Mei, *Kernel ICA feature extraction for anomaly detection in hyperspectral imagery*, Chin. J. Electron. **21** (2012), no. 2, 265–269.
19. X. S. Gan et al., *Anomaly intrusion detection based on PLS feature extraction and core vector machine*, Knowl.-Based Syst. **40** (2013), 1–6.
20. B. Zamani, A. Akbari, and B. Nasersharif, *Evolutionary combination of kernels for nonlinear feature transformation*, Inf. Sci. **274** (2014), 95–107.
21. D. Lunga et al., *Manifold-learning-based feature extraction for classification of hyperspectral data*, IEEE Signal Process. Mag. **31** (2014), no. 1, 55–66.
22. S.-H. Lee and J. S. Lim, *Parkinson's disease classification using gait characteristics and wavelet-based feature extraction*, Expert Syst. Appl. **39** (2012), no. 8, 7338–7344.
23. F. Zhang, *Nonlinear feature extraction and dimension reduction by polygonal principal curves*, Int. J. Pattern Recognit. Artif. Intell. **20** (2006), no. 1, 63–78.
24. J. G. Moreno-Torres et al., *Repairing fractures between data using genetic programming-based feature extraction: a case study in cancer diagnosis*, Inf. Sci. **222** (2013), 805–823.
25. Y. K. Kwon and B. R. Moon, Nonlinear feature extraction using a neuro genetic hybrid, in *Proc. Int. Conf. Genetic Evolutionary Computat. Conf. (GECCO)*, Washington, DC, USA, June 25–29, 2005, pp. 2089–2096.
26. R. A. Fisher, *The use of multiple measurements in taxonomic problems*, Ann. Eugen. **7** (1936), no. 2, 179–188.
27. S. Mika et al., *Fisher discriminant analysis with kernels*, in Proc. 1999 IEEE Signal Process. Soc. Workshop, Neural Netw. Signal Process., Madison, WI, USA, Aug. 1999, pp. 41–48.
28. S. T. John and C. Nello, *Kernel methods for pattern analysis*, Cambridge University Press, Cambridge, UK, 2004.
29. A. Narasimhamurthy and L. I. Kuncheva, *A Framework for generating data to simulate changing environments*, in Proc. IASTED Int. Multi-Conf.: Artif. Intell. Applicat. (AIAP), Innsbruck, Austria, Feb. 2007, pp. 384–389.
30. L. I.Kuncheva, Artificial data sets, 2007, available at http://pages.bangor.ac.uk/~mas00a/activities/artificial_data.htm.
31. C. Cortes and V. N. Vapnik, *Support-vector networks*, Mach. Learn. **20** (1995), no. 3, 273–279.
32. C. C. Chang and C. J. Lin, *LIBSVM: a library for support vector machines*, ACM Trans. Intell. Syst. Technol. **2** (2011), no. 3, article 27.

AUTHOR BIOGRAPHIES

GuiPing Wang received his BS, MS, and PhD degrees from Chongqing University, P. R. China, in 2000, 2003, and 2015, respectively. Currently, he is an associate professor in the College of Information Science and Engineering, Chongqing Jiaotong University. His research interests include machine learning, dependability analysis and design of distributed systems, and cloud computing. As the first author, he has published over 20 papers in related research areas.

JianXi Yang received his PhD degree from Chongqing Jiaotong University, P. R. China, in 2011. Currently, he is a professor in the College of Information Science and Engineering, Chongqing Jiaotong University, P. R. China. His research interests include state monitoring, fault diagnosis, big data processing, and cloud computing. He has published over 30 papers in related research areas.

Ren Li received his BS, MS, and PhD degrees from Chongqing University, P. R. China, in 2007, 2009, and 2013, respectively. Currently, he is an associate professor in the College of Information Science and Engineering, Chongqing Jiaotong University, P. R. China. His research interests include cloud computing, big data processing, and semantic web techniques. He has published over 10 papers in related research areas.