symmetry

MDPI

Article

A Novel Hybrid Method for KPI Anomaly Detection Based on VAE and SVDD

Yun Zhao , Xiuguo Zhang *, Zijing Shang and Zhiying Cao *

School of Information Science and Technology, Dalian Maritime University, Dalian 116026, China; zhao_yun@dlmu.edu.cn (Y.Z.); shangzj@dlmu.edu.cn (Z.S.)

* Correspondence: zhangxg@dlmu.edu.cn (X.Z.); czysophy@dlmu.edu.cn (Z.C.)

Abstract: Key performance indicator (KPI) anomaly detection is the underlying core technology in Artificial Intelligence for IT operations (AIOps). It has an important impact on subsequent anomaly location and root cause analysis. Variational auto-encoder (VAE) is a symmetry network structure composed of encoder and decoder, which has attracted extensive attention because of its ability to capture complex KPI data features and better detection results. However, VAE is not well applied to the modeling of KPI time series data and it is often necessary to set the threshold to obtain more accurate results. In response to these problems, this paper proposes a novel hybrid method for KPI anomaly detection based on VAE and support vector data description (SVDD). This method consists of two modules: a VAE reconstructor and SVDD anomaly detector. In the VAE reconstruction module, firstly, bi-directional long short-term memory (BiLSTM) is used to replace the traditional feedforward neural network in VAE to capture the time correlation of sequences; then, batch normalization is used at the output of the encoder to prevent the disappearance of *KL* (Kullback–Leibler) divergence, which prevents ignoring latent variables to reconstruct data directly. Finally, exponentially weighted moving average (EWMA) is used to smooth the reconstruction error, which reduces false positives and false negatives during the detection process. In the SVDD anomaly detection module, smoothed reconstruction errors are introduced into the SVDD for training to determine the threshold of adaptively anomaly detection. Experimental results on the public dataset show that this method has a better detection effect than baseline methods.

Keywords: key performance indicator (KPI); anomaly detection; variational auto-encoder (VAE); support vector data description (SVDD)

check for updates

Citation: Zhao, Y.; Zhang, X.; Shang, Z.; Cao, Z. A Novel Hybrid Method for KPI Anomaly Detection Based on VAE and SVDD. *Symmetry* **2021**, *13*, 2104. https://doi.org/10.3390/sym13112104

Academic Editors: Kóczy T. László and István A. Harmati

Received: 11 September 2021
Accepted: 26 October 2021
Published: 5 November 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

In recent years, with the development of technologies, such as machine learning and deep learning, the concept of Artificial Intelligence for IT operations (AIOps) has been proposed. AIOps combines Artificial Intelligence (AI) with operation and maintenance (O and M) to automatically monitor and manage IT services, and improve O and M efficiency. KPI (key performance indicator) anomaly detection is an underlying core technology of intelligent operation and maintenance. Most of the key technologies of intelligent operation and maintenance depend on the results of KPI anomaly detection [1]. In order to provide an efficient and reliable service, KPIs must be monitored in real time to detect anomalies on time. It is necessary for those KPI fluctuations with relatively short durations that must also be accurately monitored to avoid future economic losses. KPI data is a time series data with specific meaning, obtained through periodic sampling in the format of (timestamp, value). KPIs can be roughly divided into two types: service KPIs and machine KPIs. Service KPIs can reflect the scale and quality of web services, such as web page response time, web page visits, number of connection errors, etc. Machine KPIs can reflect the health status of machines (servers, routers, and switches), such as CPU utilization, memory utilization, disk IO, network card throughput, etc. In addition, KPIs also show diversity in shape characteristics, which can be roughly divided into periodic KPIs, stable

KPIs, and continuously fluctuating KPIs. In the actual scenes, the occurrence frequency of anomalies is very low, which leads to extremely unbalanced data samples. Due to the complexity of the business system, it will be constantly updated and upgraded, resulting in the diversity of anomaly types. Because of these characteristics, the precision and recall of existing anomaly detection algorithms are not high, and there are a lot of false positives and false negatives. This not only increases the workload of operation and maintenance personnel, but also makes them unable to find abnormal KPIs timely and accurately.

At present, a series of KPI anomaly detection methods have been proposed by the academia and industry, and these methods are gradually changing from statistical methods to machine learning methods. Deep learning is a subset of machine learning that can automatically learn features from data to achieve good performance and flexibility. As a powerful symmetrical neural network, deep generative models have been widely used in the field of anomaly detection. The learning goal of deep generation models is to narrow the gap between the restored data and the original data as much as possible. Based on the idea that normal data patterns occur frequently and anomalies rarely occur, the "compression restore" process will find the main data patterns instead of restoring the abnormal patterns. Anomaly detection needs to learn the normal pattern of data, so generation models are very suitable. Among them, the representative algorithms are variational auto-encoder (VAE) [2] and generative adversarial network (GAN) [3].

VAE is an unsupervised generative network model, which is composed of encoder and decoder. The encoder maps the input data X to latent variable Z, and the decoder maps the latent variable Z back to X. Generally, the encoder and decoder are the same and share network parameters, so this architecture is called symmetrical [4]. VAE and GAN learn the distribution of normal data, while abnormal data cannot fit this distribution. Anomaly detection is based on the asymmetry of normal data and abnormal data distribution. VAE obtains the distribution of data by variational inference. GAN directly uses the generator to simulate the distribution of data, and the discriminator determines whether the distribution simulated by the generator is good or bad. VAE is less difficult to train and more robust to noise than GAN, so it is more suitable for KPI anomaly detection. However, KPI anomaly detection methods based on VAE still have the following problems:

(1) VAE is not well suited for time series modeling. Previous VAE-based KPI anomaly detection methods [5,6] regard time series as sliding windows, ignoring the time relationship between sliding windows in the encoding process. In order to solve this problem, researchers combine LSTM [7] and VAE. Specifically, LSTM is used to replace the feedforward neural network in VAE, which can extract the characteristics, such as time dependence and correlation between data [8,9]. However, when VAE combines with the strong autoregressive decoder (LSTM), *KL* (Kullback–Leibler) divergence will disappear [10]. Because of the autoregressive of decoder, latent variables in VAE are often ignored and data is reconstructed directly. At this time, the approximate posterior is close to the prior, which causes the *KL* divergence term in the loss function to be reduced to 0. Some studies [10–13] have tried to solve this problem before, but additional parameters or training processes need to be added.

(2) VAE needs to set the threshold for anomaly detection. VAE detects anomalies by comparing the reconstruction results with the original inputs, that is, reconstruction errors. To some extent, the reconstruction error represents an instantaneous measure of anomaly degree. If a threshold is set directly on the reconstruction error, it will lead to a large number of false positives and false negatives. Moreover, for a large number of different types of KPIs, it is difficult to set a unified threshold for reconstruction errors. Early VAE-based anomaly detection studies [5,14] often ignored the importance of threshold selection. Some studies [14,15] adjusted the threshold through cross-validation. However, anomalous samples are rare, and establishing a sufficiently large validation set is a luxury. Other attempts [5,16] only evaluate the best performance of models in the test set, which makes it difficult to reproduce

the results in practical application. Therefore, anomaly detection models need to determine the threshold automatically.

In response to the above-mentioned problems, this paper applies VAE and SVDD to KPI anomaly detection. We use the public data set collected from the real operation and maintenance environment to prove the effectiveness of this method. The main contributions of this paper are summarized as follows:

(1) In order to better capture the time correlation of the KPI time series, the encoder and decoder of the VAE are designed as BiLSTM [17]. Compared with LSTM, BiLSTM processes sequences in both positive and negative directions. Its advantage lies in considering not only past KPI data, but also future KPI data.

(2) It focuses on the problem of the disappearance of *KL* divergence in the loss function during model training, avoiding the strong autoregressive decoder to ignore latent variables and directly reconstruct the data. In this paper, batch normalization [18] is used at the output of the encoder to make the *KL* divergence have a lower bound greater than zero. This method can effectively prevent the disappearance of *KL* divergence without introducing any new model components or modifying targets.

(3) Due to the unpredictability of system behavior, normal behavior can also lead to sharp error peaks. In this paper, EWMA [19] is used to smooth the reconstruction error to suppress frequent error peaks. Simultaneously, the effect of eliminating short-term trends and retaining long-term trends can be achieved, which will minimize false positives and false negatives in the detection process.

(4) In order to solve the threshold adaptation problem of KPI anomaly detection, smoothed reconstruction errors are put into the SVDD [20] for training. The threshold determined by the SVDD has good adaptability and improves the performance of anomaly detection.

2. Related Work

At present, there are few anomaly detection methods for KPI, but, as a kind of time series data, many time series anomaly detection methods are worthy of reference. The existing studies in this section are divided into three categories: traditional statistical methods, supervised machine learning methods, and unsupervised machine learning methods.

The method based on traditional statistics is the earliest method to study time series. The general idea of this method is to make some assumptions about the distribution of data, and then use the statistical inference method to find the anomalies under this assumption. For example, the well-known $3 - \sigma$ [21] criterion assumes that data follow a normal distribution, and, if some values exceed 3 standard deviations, they can be considered outliers. With the development of technology, the ARIMA [22,23] and Holt–Winters [24] methods are proposed. Both of these algorithms use a predictive idea to fit the law of time series. Then, prediction results are compared with actual time series, and anomalies are determined by setting a threshold for prediction errors. However, anomaly detection methods based on traditional statistics usually have simple assumptions about time series. Moreover, experts are required to select detectors for given time series and fine-tune the parameters of detectors based on the training data. Therefore, it does not apply to complex monitoring indicator data in actual O and M scenes.

The method based on supervised machine learning can avoid parameters adjustment in traditional statistical algorithms. Among them, the EGADS [25] framework developed by Yahoo and the Opprentice [26] framework developed by then Tsinghua Netman Laboratory are very representative. EGADS and Opprentice are supervised ensemble learning methods. These two methods use anomaly scores output by various traditional anomaly detection algorithms as features, and use user feedback as labels to train anomaly classifiers, which have achieved good results in KPI anomaly detection. However, supervised methods rely heavily on good manual labeling, which is usually not feasible in practical applications. In addition, the ensemble learning classifier based on multi anomaly detectors also faces some problems, such as a large amount of calculation, imbalance of positive and negative samples, among others. Therefore, unsupervised learning methods have become the main research direction of KPI anomaly detection.

The method based on unsupervised machine learning determines the "normal area" by using a single class label (normal KPI samples). Then, by comparing the difference between the KPI observation value and "normal area", we can infer the abnormal degree of data. Since normal samples are far more numerous than abnormal samples in anomaly detection, the model can still be trained even without labels. However, traditional unsupervised machine learning methods need to spend a lot of time to extract the features of data for anomaly detection, such as OCSVM [27], K-means [28], GMM [29], etc. Since deep generative models can automatically capture complex features from data and have higher accuracy, they have received extensive attention. Donut [5] was the first unsupervised model that applied a deep generative model to KPI anomaly detection. Donut puts forward innovations, such as M-ELBO, MCMC iteration, and missing value zero fillings on the basis of VAE, which has excellent performance on periodic KPIs. Subsequently, Buzz [6] solved the problem that was difficult for donut as it handles more complex data distribution. It measures the distance of data distributions and generates distributions through the Wasserstein distance. In fact, Buzz optimizes the likelihood evidence lower bound of a variant VAE by adversarial training. Buzz has a better detection effect on aperiodic KPIs. However, the KPI anomaly detection method based on VAE does not consider the time dependence of data, which limits its applicability to time series. Although LSTM-VAE [8] solves this problem, it will encounter the problem of *KL* divergence disappearing during training. In addition, the VAE judges anomalies by means of reconstruction, with manual determination of thresholds that have poor adaptability. Some recent studies [30,31] have achieved good results by using models to automatically determine thresholds.

This paper uses the public benchmark data set, with related research works of anomaly detection under the same data set that are introduced as follows. KPI-TSAD [32] is a time series deep learning model based on convolution and long short-term memory (LSTM) neural network, and uses a variational auto-encoder (VAE) oversampling model to solve the imbalanced classification problem. Although the method based on supervised learning has achieved good performance in anomaly detection, it needs a lot of labeled data for training. LSTM-based VAE-GAN [9] regards the long short-term memory (LSTM) network as the encoder, generator, and discriminator of VAE-GAN, and jointly trains the encoder, generator, and discriminator. In the anomaly detection stage, anomalies are detected based on reconstruction errors and discrimination results. However, it needs to accumulate certain data to adjust the threshold of the abnormal score. PAD [33] is a method for robust prediction and unsupervised anomaly detection. The prediction block (LSTM) obtains a clean input from the time series reconstructed by the VAE, making it robust to anomalies and noise. At the same time, because LSTM helps to maintain a long-term sequence pattern, VAE performs better in anomaly detection. ALSR [34] is a machine learning scheme for continuous interval KPI anomaly detection. The anomaly detection scheme is optimized by using the different characteristics of abnormal points in the continuous anomaly interval, so that it has better detection accuracy. FluxEV [35] mainly improves SPOT [36], which is only sensitive to extreme values and therefore cannot detect local fluctuations. The method of moment estimation is used to optimize maximum likelihood estimation in SPOT to improve computational efficiency. This paper mainly aims at solving the problems of KPI anomaly detection based on VAE. However, the works discussed above are different from the problems addressed in this paper. For example, KPI-TSAD [32] solves the problem of data imbalance through VAE. LSTM-based VAE-GAN [9] aims to resolve the problem of errors in the mapping of GAN from real-time space to potential space. PAD [33] considers two aspects: state prediction and anomaly detection. ALSR [34] mainly focuses on anomaly detection in continuous intervals. FluxEV [35] focuses on improving computational efficiency.

3. Anomaly Detection Method

3.1. Method Flow

The problem to be solved in this paper is how to detect anomalies in KPI time series data. In order to solve this problem, a novel hybrid anomaly detection method is proposed. The method flow is shown in Figure 1, which mainly includes data preprocessing, a VAE-based reconstruction module, and SVDD-based anomaly detection module. In the training stage, data preprocessing was firstly carried out, that is, missing value filling and data normalization were performed on the original KPI time series. Then, the VAE reconstruction was carried out, that is, the BiLSTM-VAE model was trained and batch normalization was used to prevent the disappearance of the *KL* divergence. Finally, SVDD anomaly detection was carried out, and reconstruction errors smoothed by EWMA were put into the SVDD for training. The center a and radius R of the SVDD hypersphere were calculated, and the radius R is the threshold of anomaly detection. In the test stage, the test data were preprocessed and input into the trained BiLSTM-VAE model to obtain the reconstructed test data. If the smoothed reconstruction error was less than or equal to the threshold R, it would be judged as normal. If it was greater than the threshold R, it would be judged as abnormal.

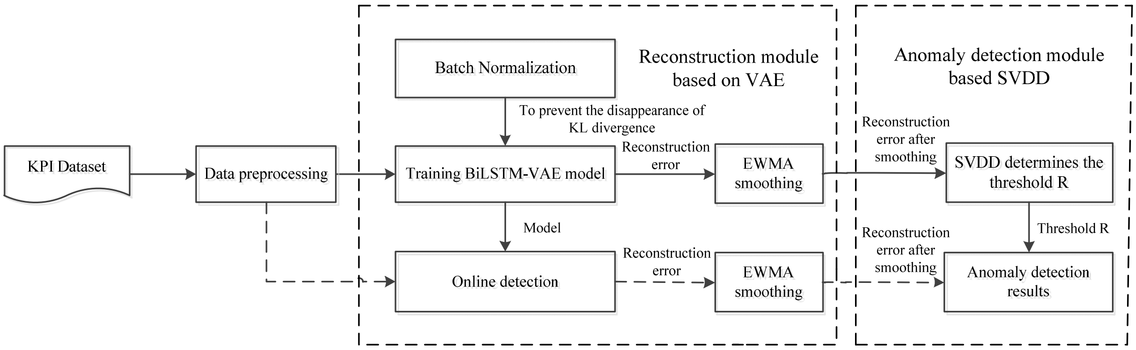

KPI Dataset
Data preprocessing
Batch Normalization
Reconstruction module based on VAE
To prevent the disappearance of KL divergence
Training BiLSTM-VAE model
Reconstruction error
EWMA smoothing
Model
Online detection
Reconstruction error
EWMA smoothing
Anomaly detection module based SVDD
Reconstruction error after smoothing
SVDD determines the threshold R
Threshold R
Reconstruction error after smoothing
Anomaly detection results

Figure 1. Method flow chart.

3.2. Data Preprocessing

3.2.1. Missing Value Processing

In the real scene, there may be a small number of missing reports or noise data may be deleted in the original KPI data, resulting in the loss of values in the data. Supplementing appropriate data is helpful for subsequent model training. When the number of missing values is small, the effects of nearest neighbor interpolation, linear interpolation, and cubic polynomial interpolation are similar [37]. There are fewer missing values in the KPI dataset used in this paper. After measuring speed and simplicity, the linear interpolation method was selected to deal with missing values. First, the slope was calculated according to the data before and after the missing value, and then the missing KPI data was supplemented according to the slope. Figure 2 shows the interpolation method. If the data x_k was lost, the slope would be calculated as follows:

$$b = \frac{x_{k+1} - x_{k-1}}{t_{k+1} - t_{k-1}}, \tag{1}$$

Next, the missing KPI data x_k was calculated based on the slope b:

$$x_k = x_{k-1} + b \times (t_k - t_{k-1}), \tag{2}$$

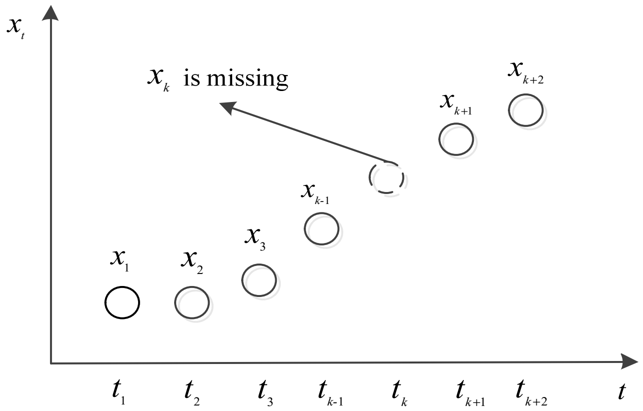


Figure 2. Schematic diagram of the interpolation method.

3.2.2. Data Standardization

In order to eliminate the dimensional influence between indicators, data standardization is needed. After data standardization, all indicators are in the same order of magnitude, which is suitable for comprehensive comparative evaluation. In addition, it can reduce the training time of the model and make the training process converge as soon as possible. This paper normalizes the KPI data, and the data was mapped to the range of 0–1. The normalization formula is as follows:

$$x^* = \frac{x - \min}{\max - \min}, \tag{3}$$

3.3. Reconstruction Module Based on VAE

3.3.1. BiLSTM-VAE Model

In physics, symmetry has a more profound meaning, which refers to invariance under certain transformations. In the VAE, the data is invariant in time and space after encoding and decoding operations, so it just conforms to the concept of symmetry.

The encoder of the VAE is used to learn the distribution of training data and generate the compressed value of training data, and the decoder reconstructs the compressed data. The basic idea is to use a deep neural network to model two complex probability density functions: posteriori probability distribution and conditional probability distribution. The neural network fitting $x \to z$ is called the inference network $q_{\varphi}(z \mid x)$, as shown in Formula (4). The neural network fitting $z \to x$ is called the generative network $p_{\theta}(x \mid z)$, as shown in Formula (5).

$$z \sim Enc(x) = q_{\varphi}(z \mid x), \tag{4}$$

$$x \sim Dec(z) = p_{\theta}(x \mid z), \tag{5}$$

KPI data belongs to time series. Using the memory function of LSTM [7], LSTM network units are introduced into the VAE network to replace traditional neural units in the inference network and generation network. The time dependence and correlation of input data can be learned, which is helpful to extract appropriate features in the hidden layer and reconstruct the input sequence. In this paper, BiLSTM [17] is used as the encoder and decoder of VAE. Compared with LSTM, the advantage of BiLSTM is not only to consider the past KPI data, but also to consider the future KPI data. Figure 3 shows the network structure of BiLSTM-VAE.

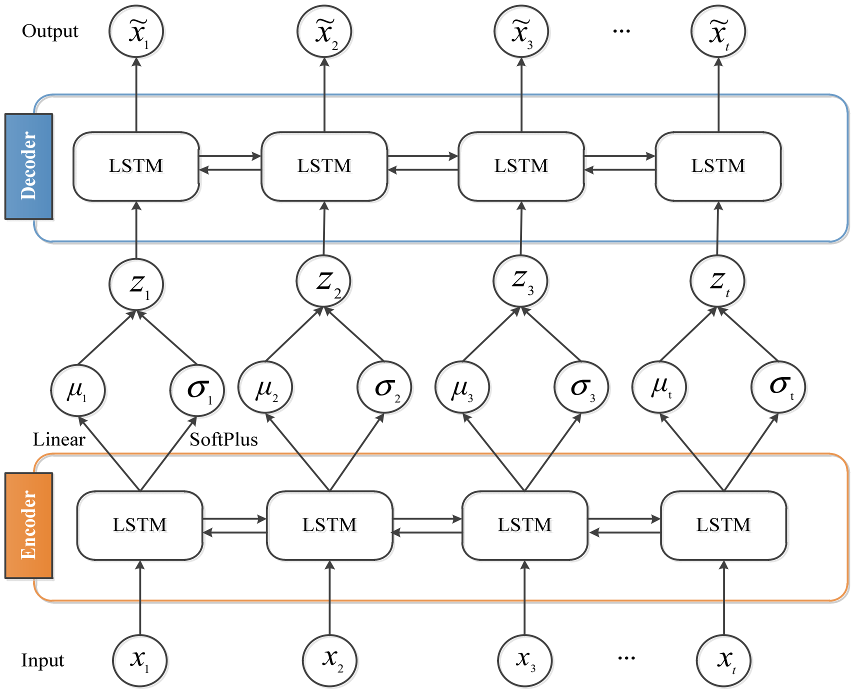


Figure 3. BiLSTM-VAE network structure.

Firstly, time series were divided into sub-sequences corresponding to input variables by the sliding window with a certain step size. Each input sample of the encoder was a vector of a specific size, expressed as $x = \{x_1, x_2, x_3, \cdots, x_t\}$. Then, the encoder encoded input variables into latent variables through the inference network. It was assumed that the true posterior of latent variables z obeys the standard Gaussian distribution (standard normal distribution), i.e., $p_\theta(z) = N(0, I)$. According to the description of reference [2], the standard normal distribution can simulate any distribution through a sufficiently complex function. It can be proved by the inverse transformation sampling theorem. $F(x)$ is a cumulative distribution function; U is a standard normal distribution variable between 0 and 1; and $F^{-1}(U)$ is a sample of the target distribution. Then, the following formula can be obtained:

$$\begin{aligned} & P(F^{-1}(U) \le x) \\ & = P(U \le F(x)) \ , \\ & = F(x) \end{aligned} \tag{6}$$

In short, we can use a normal distribution to obtain a complex distribution through the $D(z)$ function of the decoder to output $\widetilde{x}$. In this way, $\widetilde{x}$ and x have the same probability distribution and content. Therefore, this assumption was reasonable. Specifically, given a real sample x_t, we assumed that there is a distribution $p_\theta(z \mid x_t)$ exclusively belonging to x_t, and further assumed that this distribution is (independent and multivariate) normal distribution. If $p_\theta(z \mid x_t)$ belongs exclusively to x_t, it is reasonable to say that z sampled from this distribution should be restored to x_t. Since the distributions assumed above are normal distributions, it was necessary to obtain the corresponding variance and mean. Then, a z_t was sampled from this exclusive distribution and $\widetilde{x}$ was obtained through a decoder $\widetilde{x}_t = D(z_t)$.

The approximate posterior distribution $q_\varphi(z \mid x)$ also obeys the Gaussian distribution $N(\mu, \sigma I)$, where μ and σ are the mean and variance of the Gaussian distribution. It is not difficult to see that the function of the encoder is to generate the mean μ and variance σ through two networks. The encoder was parameterized through BiLSTM with an activation function to generate hidden state sequences in both directions, forward $\rightarrow$ and backward $\leftarrow$. The final encoder hidden states of both passes were concatenated with each other to produce the vector $h_t = [\overrightarrow{h}_t, \overleftarrow{h}_t]$. μ and σ were derived from the final encoder hidden state h_t using two fully connected layers with linear and Softplus activations, respectively.

Latent variables were obtained by reparameterization, that is, $z = \mu + \sigma \odot \varepsilon$. Among them, $\varepsilon \sim N(0, I)$ is an auxiliary noise variable, and $\odot$ represents the product at the element level. Evidently, noise will increase the difficulty of reconstruction. Resampling essentially adds "Gaussian noise" to the encoder result (the mean value), so that the decoder result can be robust to noise. Another encoder result (the variance) was used to dynamically adjust the intensity of noise. Intuitively, when the decoder is not well trained, it will appropriately reduce the noise to make the fitting easier (the reconstruction error becomes smaller). On the contrary, if the decoder is well trained, the noise will increase, making the fitting more difficult (the reconstruction error becomes larger).

Finally, the decoder decoded latent variables back to the original data space through the generation network $p_\theta(x \mid z)$, so as to obtain reconstructed data samples.

In VAE, the parameters of the network were optimized by maximizing the lower bound of evidence $ELBO_{vae}$, as shown in Formula (7):

$$ELBO_{vae} = E_{q_\varphi(z|x)}[\log\ p_\theta(x \mid z)] - D_{KL}(q_\varphi(z \mid x) \mid p_\theta(z)), \tag{7}$$

where the first term represents the reconstruction term, and $E_{q_\varphi(Z|X)}$ is the logarithmic likelihood estimate of the posterior probability of x. The second term represents the regularization term, which measures the gap between approximate posterior $q_\varphi(z \mid x)$ and true posterior $p_\theta(z)$ by *KL* divergence. The goal of optimization is to maximize the likelihood function of generated data and minimize the *KL* divergence between the approximate posterior distribution and the true posterior distribution. In short, on the one hand, the output was fitted to the input as much as possible. On the other hand, the noise was appropriately increased through the *KL* divergence to prevent over-fitting.

3.3.2. Batch Normalization Prevents the Disappearance of *KL* Divergence

When VAE is used with a powerful autoregressive decoder (LSTM), *KL* divergence often disappears. This is generally believed due to the strong autoregression of the decoder, that is, the generated network $p_\theta(x \mid z)$ is too strong. This will cause the model to abandon the use of the approximate posterior of encoder and directly use the latent variables of the model. At the same time, the *KL* divergence term will quickly decrease to 0, that is, prior and approximate posterior are equal. In addition, the reparameter operation will introduce noise during training. When it has high noise, latent variables are difficult to be used, so the VAE ignores latent variables and carries out the reconstruction independently. When the *KL* divergence is 0, the encoder outputs a constant vector. The use of VAE usually focuses on its ability to construct coding vectors unsupervised. Therefore, the problem of the *KL* divergence disappearance must be solved when applying VAE.

Based on the batch normalized-VAE (BN-VAE) method proposed in reference [18], this paper solves the problem of *KL* disappearance. This method has good results in language modeling, text classification, and dialog generation. This paper applies it to the field of anomaly detection for the first time. The core idea of BN-VAE is to apply BN to the mean vector output by the encoder, so as to ensure that the expected lower limit of *KL* divergence distribution is positive, as shown in Figure 4.

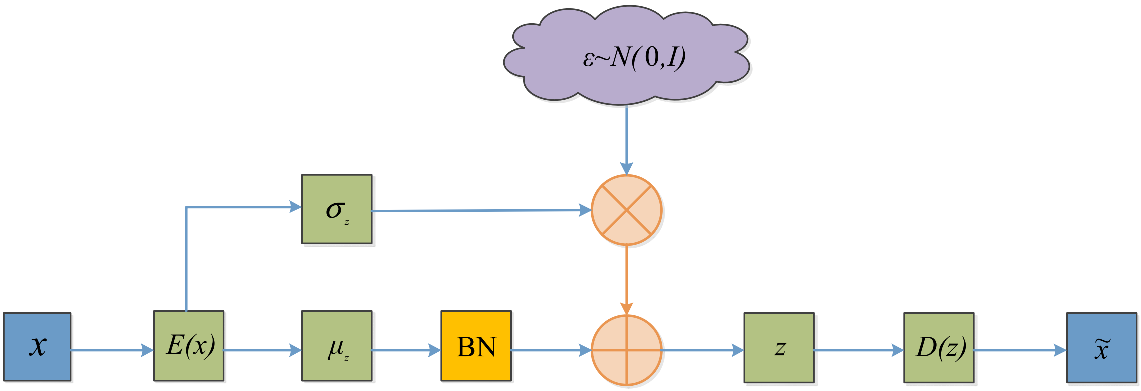


Figure 4. Adding BN to VAE prevents the disappearance of the *KL* divergence.

In order to explain how BN was associated with *KL* divergence, the *KL* divergence term formula is given first:

$$KL = \frac{1}{2}\sum_{i=1}^{d}\mu_i^2 + \sigma_i^2 - \log\sigma_i^2 - 1, \tag{8}$$

In the above formula, d is the dimension of latent variables. μ_i and σ_i are the mean and standard deviation of posterior distribution of the i-th dimension of latent variables, respectively. In the actual calculation, we often used batch training, so the above formula was further calculated during the training process:

$$\begin{aligned} KL &= \frac{1}{2b}\sum_{j=1}^{b}\sum_{i=1}^{d}(\mu_{i,j}^2 + \sigma_{i,j}^2 - \log\sigma_{i,j}^2 - 1) \\ &= \frac{1}{2}\sum_{i=1}^{d}(\frac{\sum_{j=1}^{b}\mu_{i,j}^2}{b} + \frac{\sum_{j=1}^{b}\sigma_{i,j}^2}{b} - \frac{\sum_{j=1}^{b}\log\sigma_{i,j}^2}{b} - 1) \end{aligned}, \tag{9}$$

In the above formula, b is the size of batch. When b is large enough, the *KL* term will approximate the average value of *KL* of the whole data set. Thus, we limited the distribution of *KL* in the data set by limiting the distribution of mean and variance. In this way, *KL* was equivalent to the distribution of posterior distribution parameters of latent variables. Therefore, the above formula can be expressed as follows:

$$\begin{aligned} E[KL] &= \frac{1}{2}\sum_{i=1}^{d}(Var[\mu_i] + E^2[\mu_i] + E^2[\sigma_i^2] - E[\log\sigma_i^2] - 1) \\ &\geq \frac{1}{2}\sum_{i=1}^{d}(Var[\mu_i] + E^2[\mu_i]) \end{aligned}, \tag{10}$$

In the above formula, $E^2[\sigma_i^2] - E[\log\sigma_i^2] - 1 \geq 0$ can be derived from $e^x \geq x + 1$, so the inequality holds. Through this transformation, it is not difficult to realize that batch normalization can be used to constrain the distribution of mean. The mean value in the posterior distribution was performed as follows:

$$\hat{\mu}_i = \gamma\frac{\mu_i - \mu_{\mathrm{B}i}}{\sigma_{\mathrm{B}i}} + \beta, \tag{11}$$

In the above formula, $\hat{\mu}_i$ is the mean value of μ_i transformed by the BN layer. $\mu_{\mathrm{B}i}$ and $\sigma_{\mathrm{B}i}$ represent the mean and standard deviation of μ_i. β and γ are parameters in batch normalization, which can control the variance and mean value of μ_i distribution, respectively. Finally, Formula (12) was obtained by replacing μ_i in KL formula:

$$\begin{aligned} E[KL] &\geq \frac{1}{2}\sum_{i=1}^{d}\left(Var[\mu_i] + E^2[\mu_i]\right) \\ &= \frac{d}{2}(\gamma^2 + \beta^2) \end{aligned}, \tag{12}$$

Therefore, as long as β and γ are well controlled (mainly γ is fixed to a certain constant), the KL divergence term can have a positive lower bound. In this way, KL divergence and BN were cleverly linked to avoid the disappearance of KL divergence.

3.3.3. EWMA Smoothing Reconstruction Errors

The difference sequence $d = \mid \tilde{x} - x \mid$ can be obtained by comparing reconstructed KPI sequence with the original sequence. However, the original difference sequence represents an instantaneous measure of the predictability of the current input. Nevertheless, in many practical applications, the underlying system is inherently unpredictable. In this case, predictable change usually means meaningless behavior. This is seen, for example, in the latency of HTTP requests for websites. Although the latency is usually low, it is not uncommon for random jumps to reach the peak corresponding to anomaly scores. In fact, abnormal observations usually occur continuously, and it is acceptable to trigger an alarm in a short time. Setting thresholds directly on original difference sequences will lead to many false positives. Therefore, this paper uses EWMA [19] to smooth the difference sequence to suppress the frequently occurring error peaks. System behavior is usually not perfectly predictable, and normal behavior can also cause sharp peaks in error values [38]. At time k, the smoothed sequence e_k was obtained according to the original difference sequence d_k. The calculation process is shown in Formula (13):

$$e_k = \alpha d_k + (1 - \alpha) e_{k-1}, \tag{13}$$

In the above formula, $\alpha (0 < \alpha < 1)$ is the weight coefficient of EWMA for the historical measurement value. The closer its value is to 1, the lower weight for the past measurement value. α determines the ability of EWMA to track sudden changes in actual data, namely timeliness. With the increase in α, the timeliness of EWMA is stronger; otherwise, it is weaker. EWMA also shows a certain ability to absorb instantaneous bursts. By controlling α, short-term fluctuations are eliminated and long-term development trends are retained, providing a smooth form of sequences.

3.4. Anomaly Detection Module Based on SVDD

SVDD [20] is an algorithm that can describe the target data in a hypersphere, which can contain as many data points as possible. It can be described as: if only one class can be judged, then the smallest hypersphere needs to be found through SVDD to include all the data of this class. When the hypersphere is used to identify new data, if the data fall within the hypersphere, the data are considered to belong to this class. Otherwise, the data do not belong to this class.

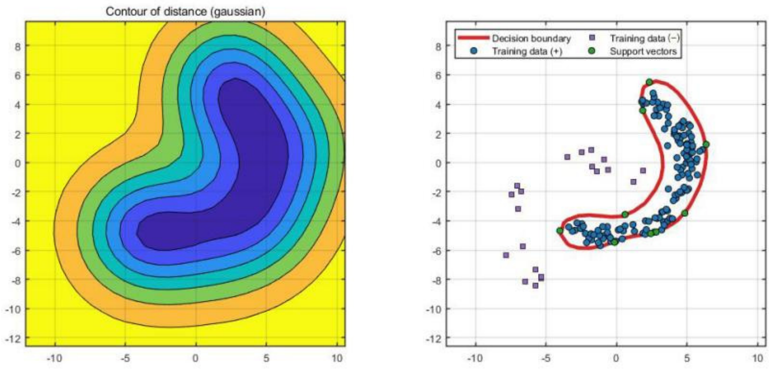
When training the SVDD classifier, this paper inputs the reconstruction error of normal data into the SVDD for training to determine the threshold. In the test phase, the reconstruction error of abnormal data was greater than that of normal data, so it exceeded the threshold to realize anomaly detection. However, different KPI curves correspond to different reconstruction error curves. If a fixed threshold is set based on human experience, a large number of false positives and false negatives will be caused. Therefore, this paper inputs the reconstruction error into SVDD for training to determine the threshold, which can adaptively set different thresholds for different KPIs.

The goal of the SVDD is to find support vectors and use them to construct a minimal closed hypersphere that contains all or most of the target training samples. In this paper, target training samples are the smoothed reconstruction error of normal KPI, expressed as $e = \{e_1, e_2, e_t, \cdots, e_n\}$, where n is the number of samples. The sample was distributed in a ball with center a and radius R, i.e., $\|e_t - a\|^2 \leq R^2$. By introducing the slack variable ξ_t, it was allowed that some samples were no longer in the ball, that is, $\|e_t - a\|^2 \leq R^2 + \xi_t$. The training objective was to minimize the value of radius R and slack variable ξ_t, so the objective function was expressed as Formula (14):

$$\begin{gathered} \min F(R, a, \xi_t) = R^2 + C\sum_{t=1}^{n} \xi_t \\ s.t.\left\{ \begin{array}{l} \|e_t - a\|^2 \leq R^2 + \xi_t, (t = 1, 2, \cdots, n) \\ \xi_t \geq 0 \end{array} \right\}, \end{gathered} \tag{14}$$

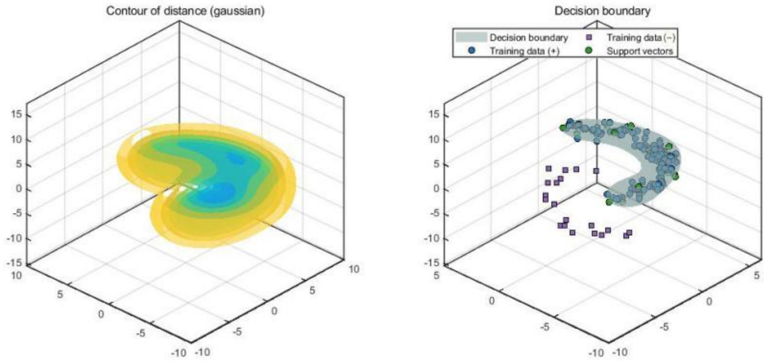
Among them, ξ_t is the slack variable, which is used to measure a small amount of abnormal data outside the hypersphere. C is the penalty coefficient used to control the volume of the hypersphere, and its value ranges from 0 to 1. The slack variable ξ_t prevented the model from being "destroyed" by individual extreme data points. In short, if most data points are in a small area and only a few abnormal data are far away from them, the model prefers to regard those few data points as anomalies. To avoid the model making excessive sacrifices to cater to few data points, the model tolerated some data points that did not meet the rigid constraints and gave them some elasticity. C adjusted the influence of the slack variable. Generally speaking, the slack space is given to those data points that need slack. If C is large, the loss caused by the slack variable in the loss function is large. Then, the slack variable will be reduced during training. In this way, the model does not tolerate those outliers and just wants to include them. On the contrary, if C is small, the model will give outliers greater elasticity, so that they can not be included.

In order to make the training process easier to understand, the hypersphere was visualized in two-dimensional and three-dimensional space respectively, as shown in Figures 5 and 6. The hypersphere corresponds to a curve in two-dimensional space and a sphere in three-dimensional space. Under normal circumstances, the data will not show spherical distribution, so the Gaussian kernel function method was used to improve the expression ability of model. Figures 5a and 6a show the contour distance visualization of hypersphere in two-dimensional and three-dimensional space, respectively. Figures 5b and 6b show the decision boundary visualization of hypersphere in two-dimensional and three-dimensional space, respectively.



(**a**) (**b**)

Figure 5. Visualization of the hypersphere in two-dimensional space: (**a**) Contour of distance; and (**b**) Decision boundary.



(**a**) (**b**)

Figure 6. Visualization of the hypersphere in three-dimensional space: (**a**) Contour of distance; and (**b**) Decision boundary.

The optimization problem was solved by the Lagrange multiplier method, and the following Lagrange function was obtained:

$$L(R, a, \xi_t) = R^2 + C\sum_{t=1}^{n} \xi_t - \sum_{t=1}^{n} \lambda_t \left[R^2 + \xi_t - \|e_t - a\|^2 \right] - \sum_{t=1}^{n} \beta_t \xi_t, \tag{15}$$

where λ_t and β_t are Lagrange multipliers. The distance from e_t to a is recorded as $g(e_t)$, and the calculation formula is as follows:

$$g(e_t) = \|e_t - a\| = \sqrt{(e_t, e_t) - 2\sum_{i=1}^{n} \lambda_i (e_i, e_t) + \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j (e_i, e_j)}, \tag{16}$$

The calculation formula of radius R is as follows:

$$R = \sqrt{(e_s, e_s) - 2\sum_{i=1}^{n} \lambda_i (e_i, e_s) + \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j (e_i, e_j)}, \tag{17}$$

Among them, e_s is a support vector on the sphere of the hypersphere. e_i and e_j are any two samples input to the SVDD. In addition, in order to make the samples linearly separable in the feature space, it was necessary to map samples from the original space to the high-dimensional feature space by using a kernel function. In this paper, a Gaussian kernel function is used to map samples from original space to appropriate feature space. The expression of Gaussian kernel function is:

$$K_{Gauss}(e_i, e_j) = \exp(-\|e_i - e_j\|^2 / s^2), \tag{18}$$

where s is the Gaussian kernel parameter.

In the anomaly detection stage, $g(e_{test}) > R$ indicates that the distance from e_{test} to a is greater than R, then x_{test} is the abnormal KPI data. $g(e_{test}) \leq R$ indicates that the distance from e_{test} to a is less than or equal to R, then x_{test} is the normal KPI data. The process of anomaly detection is shown in Figure 7.

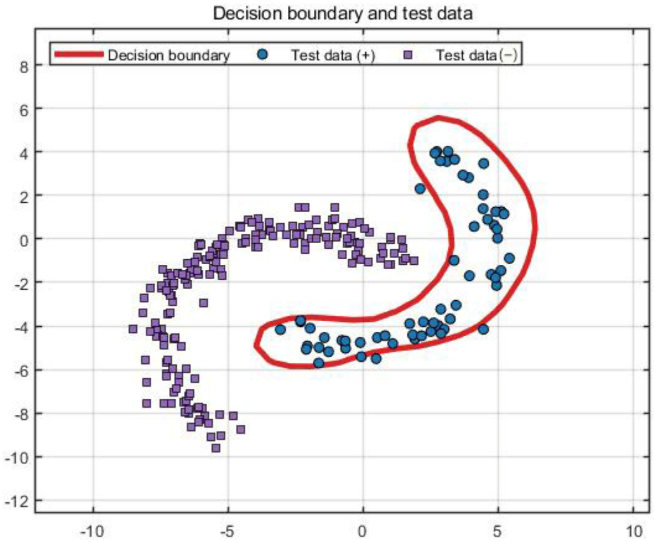


Figure 7. Anomaly detection based on SVDD.

4. Experimental Procedure

The experimental environment of this paper was Windows 10 (64-bit) operating system. The hardware configuration was Intel (R) Core (TM) i7-8700CPU@3.20 GHz 16 G RBM and 237 G solid state drive. The development language was python3.6, the development framework was Keras, and the back-end engine was TensorFlow.

4.1. Dataset

The KPI dataset used in this paper was published by the AIOps challenge competition (http://iops.ai/competition_detail/?competition_id=5&flag=1 (accessed on 10 September 2021)), which provides the KPI desensitization time series with anomaly labels. The data were collected from the real operation and maintenance environment of top Internet companies, such as Sogou, Tencent, eBay, Baidu, and Alibaba, and the sampling interval was 1 min. We randomly selected two KPIs to verify the proposed method in this paper. As shown in Table 1, the ratio of normal samples to abnormal samples in the data set was obviously very uneven, and abnormal samples account for less than 10% of the total number of samples. Figure 8 shows the visualization effect of two KPIs. It can be observed that KPIs showed a certain degree of periodicity and trend.

Table 1. Dataset details.

Dataset	KPI 1	KPI 2
Total points	128,562	129,035
Anomaly points	10,550/8.21%	7666/5.94%
Missing points	3233/0.02%	2755/0.02%
Duration	91 days	91 days
Sample Frequency	1412.77	1417.97

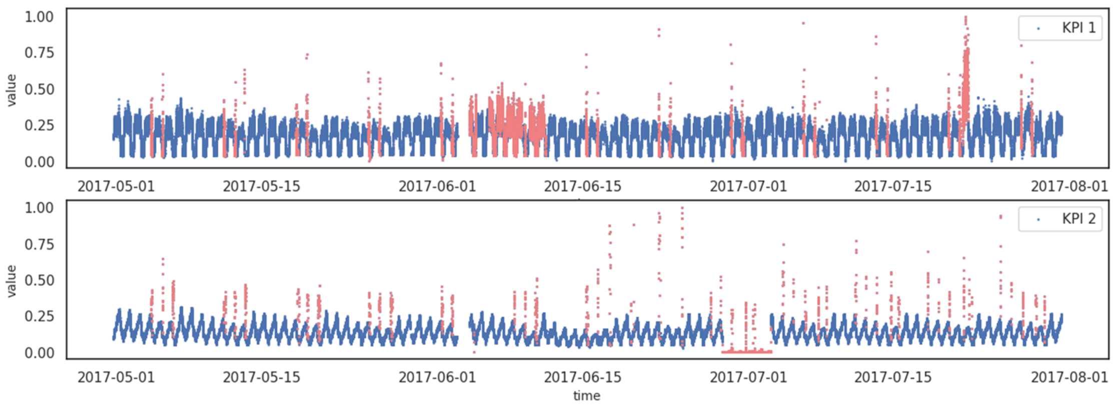
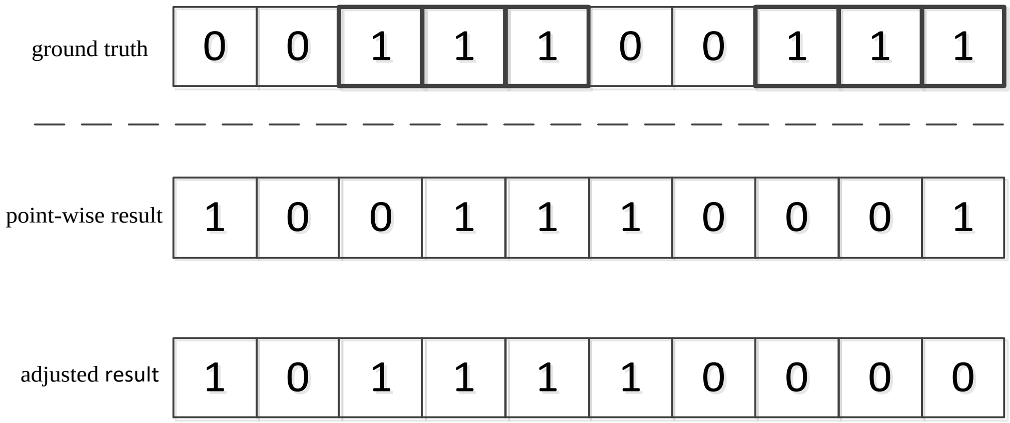


Figure 8. Two KPIs from real production environments.

4.2. Evaluation Metrics

In the anomaly detection module, this paper determines the threshold through SVDD. When the reconstruction error was greater than the threshold, the point was judged as an anomaly. Operation and maintenance personnel usually only care about whether the anomaly detection algorithm can detect a continuous anomaly interval, rather than detecting each anomaly point in the anomaly interval. Therefore, the evaluation of this paper adopts the strategy described in the literature [5]. If the anomaly detection algorithm made a judgment fast enough (before the maximum allowable delay) after the beginning of anomalies, it was considered to have successfully detected the whole anomaly segment. The alarm delay was the time difference between first anomaly point and first detection point in the anomaly segment. If the anomaly detection algorithm did not issue any alarm before the maximum allowable delay, even if the anomaly detection algorithm detected the anomaly, we considered that the algorithm failed to successfully detect the anomaly segment. Figure 9 shows the anomaly detection results with an alarm delay of 1 min (1 grid). The first line represents the real labeled data, including 10 consecutive time points and 2 anomaly intervals. The second line represents the output results of the anomaly detection method. The third line represents the anomaly detection results corrected according to the alarm delay. For the first anomaly interval, if the anomaly detection method found an anomaly within the longest delay alarm, it was considered that the whole anomaly interval was successfully detected. For the second anomaly interval, it was considered that the anomaly interval was not successfully detected, because the detection result exceeded the alarm delay.



ground truth	0	0	1	1	1	0	0	1	1	1
point-wise result	1	0	0	1	1	1	0	0	0	1
adjusted result	1	0	1	1	1	1	0	0	0	0

Figure 9. Description of anomaly detection policies.

Therefore, the anomaly detection of time series can be regarded as a classification problem. In this paper, *Precision*, *Recall*, and *F*1*-score* are used to evaluate the performance of detection. *Precision* represents the proportion of correct prediction being positive in relation to the total prediction being positive. *Recall* represents the proportion of correct prediction being positive in relation to the total actual being positive. *F*1*-score* is the weighted harmonic average of *Precision* and *Recall*. The calculation formula is as follows:

$$Precision = \frac{TP}{TP + FP}, \tag{19}$$

$$Recall = \frac{TP}{TP + FN}, \tag{20}$$

$$F1\text{-}score = \frac{2 \times Precision \times Recall}{Precision + Recall}, \tag{21}$$

where TP is the number of anomaly points correctly detected; FP is the number of normal points incorrectly identified as anomaly points; and FN is the number of anomaly points incorrectly identified as normal points.

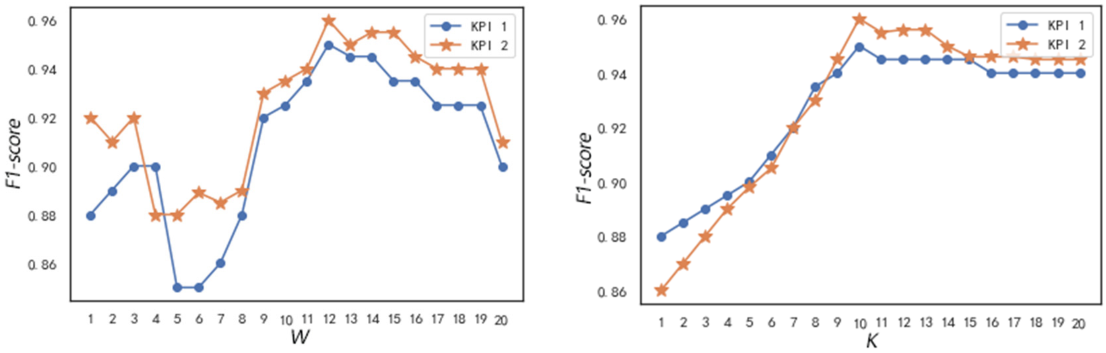
4.3. Experimental Parameter Setting

In the experiment, each KPI time series was divided into training set and test set by 8:2. After the many repeated experiments that were conducted in the context of this paper, the final hyperparameters are shown in Table 2 on the premise of balancing the time-consuming and detection effect.

Table 2. Main hyperparameter settings.

Hyperparameter Name	Hyperparameter Value
Batch size	256
Number of iterations	100
Optimizer	Adam
Learning rate	0.0005
LSTM unit size	128
Latent variable dimension	10
Sliding window length	12
Alarm delay	7
Penalty coefficient of SVDD	0.25
Gaussian kernel parameter of SVDD	9

In the process of adjusting the parameters, we found that the sliding window length W and latent variable dimension K had a great influence on the results of the anomaly detection. Too short sliding windows could not obtain the relationship between adjacent points. Too long sliding windows relied too much on historical information and lacked sensitivity to current values. Latent variables represent all the important information needed to contain the original data point. The representation ability of potential space varies with the dimension of latent variables. Therefore, this paper tests the best *F*1*-score* of algorithm under different sliding window lengths and latent variable dimensions, as shown in Figure 10. It can be seen that, when the sliding window length W = 12 and latent variable dimension K = 10, the *F*1*-score* reaches the optimal value. In addition, when SVDD determined the threshold, the selection of the penalty coefficient and kernel parameters also had an important impact on the effect of anomaly detection. In this paper, the accuracy of anomaly detection is used as the fitness function, and the penalty coefficient C = 0.25 and the Gaussian kernel parameter s = 9 are obtained by the particle swarm optimization algorithm [39].

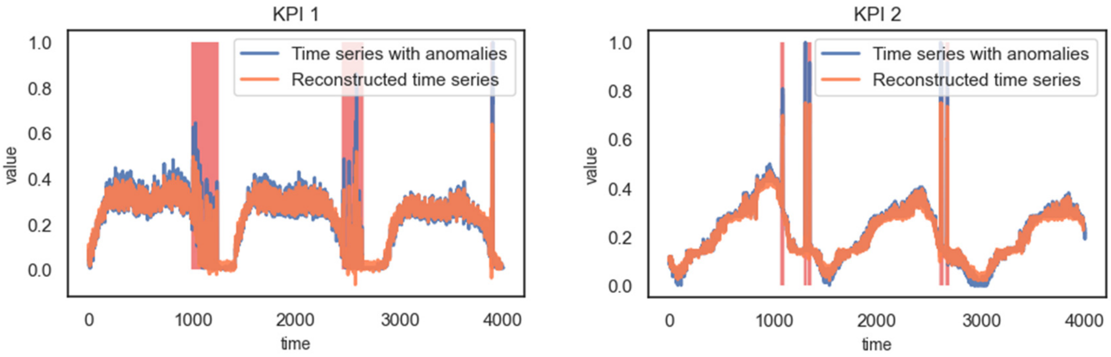


(**a**) (**b**)

Figure 10. The best *F1-score* under different sliding window lengths and latent variable dimensions: (**a**) Different sliding window lengths; and (**b**) Different latent variable dimensions.

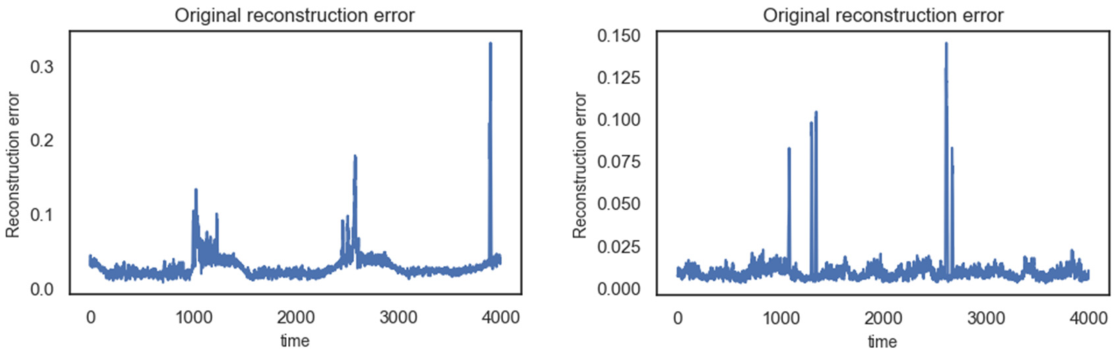
4.4. Experimental Results of Anomaly Detection

In this paper, normal data are used to train BiLSTM-VAE model and the distribution of normal data is learned. During the test phase, the model did not reconstruct the anomalous data well, because of the different distributions of the anomalous data from the normal data. To visually observe this, we plotted the reconstruction effect of two KPIs on a partial test set. As shown in Figure 11, normal samples of two KPIs can be reconstructed well, while abnormal samples cannot be reconstructed well, resulting in higher reconstruction errors. Figure 12 shows original the reconstruction errors of two KPIs. It can be seen that reconstruction errors of normal points are closer to 0, while abnormal points will lead to the error peak. However, setting a fixed threshold directly on the original reconstruction error threshold will not only lead to a large number of false positives and false negatives, but also to the need to adjust the threshold manually. In addition, it is unrealistic to set a unified threshold for different KPIs, which may lead to poor adaptability. Therefore, this paper uses EWMA to smooth the reconstruction error and SVDD to adaptively determine the threshold. Figure 13 shows smoothed reconstruction errors of two KPIs, and the red dotted line is the threshold determined by SVDD. It can be seen from the figure that the threshold of KPI 1 is 0.04, and the threshold of KPI 2 is 0.018. The errors of normal points are lower than the threshold, and the errors of abnormal points are higher than the threshold. In summary, the method in this paper can accurately detect anomalies and adaptively determine the optimal threshold for each KPI.



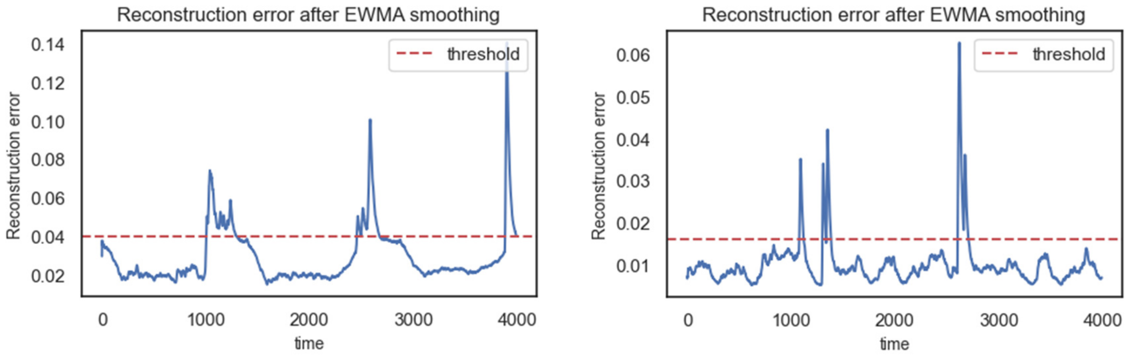
(**a**) (**b**)

Figure 11. The reconstruction effect of two KPIs: (**a**) KPI 1; and (**b**) KPI 2.



(**a**) (**b**)

Figure 12. Original reconstruction errors of two KPIs: (**a**) KPI 1; and (**b**) KPI 2.



(**a**) (**b**)

Figure 13. Smoothed reconstruction errors of two KPIs: (**a**) KPI 1; and (**b**) KPI 2.

4.5. Comparative Experiment and Analysis

In order to verify the effectiveness of this method, we selected three methods as baseline to test the detection effect of each method on the KPI data set. These methods included the VAE-based anomaly detection method Donut [5] and LSTM-VAE [8]. In fact, VAE was part of our approach. In order to make a fair comparison, the hyperparameters of Donut and LSTM-VAE were the same as the method in this paper. In addition, we also compared the supervised learning method Opprentice [26] as the most competitive method of non-deep learning:

- Opprentice [26] is an ensemble supervised algorithm that uses random forest classifiers. Its principal concept is to use more than ten different types of detectors to extract hundreds of abnormal features. Then, using the manually labeled data and anomaly features, the anomaly detection problem can be transformed into a supervised classification problem in machine learning. The extracted features are used as the input of machine learning algorithm. The points on the KPI curve are divided into normal points and abnormal points through a classification algorithm, so as to realize anomaly detection.
- Donut [5] is an unsupervised anomaly detection algorithm based on VAE. Through the improved variational lower bound and Markov chain Monte Carlo interpolation technology, the algorithm can be used without labels. Donut applies a sliding window on the KPI to obtain the sub-sequence, and tries to identify the normal pattern. Then, anomalies are determined by reconstruction probability. In fact, it selects a threshold for each KPI.
- LSTM-VAE [8] combines LSTM and VAE to make it more suitable for time series modeling. Specifically, it replaces the feedforward neural network in VAE with LSTM. LSTM-VAE fuses sequences and reconstructs their expected distribution by

introducing a schedule based variational a priori. In the anomaly detection phase, it uses an anomaly score based on reconstruction probability and a state-based threshold.

Table 3 shows the best *Precision*, *Recall*, and *F*1-*score* of various anomaly detection methods on two KPIs. Opprentice based on machine learning performed worse than VAE based on deep learning, and it needed labels for training, which was difficult to achieve in the actual scene. Donut made a series of improvements based on VAE so that it can train without labels. Since Donut treats time series as sliding windows and does not process time information, Donut's performance was poor when anomaly detection relies on time information. LSTM-VAE can extract the time correlation of sequence better than Donut, so a better detection effect was obtained. The method in this paper is VAE-SVDD—VAE uses BiLSTM as encoder and decoder; batch normalization avoids the disappearance of the *KL* divergence; EWMA smooths the original reconstruction error; and SVDD adaptively determines the threshold. Through the above improvements, VAE-SVDD had higher *Precision*, *Recall*, and *F*1-*score* compared to other baseline methods.

Table 3. Experimental results of various anomaly detection methods.

Method	**KPI 1**			**KPI 2**		
	Precision	***Recall***	***F1-Score***	***Precision***	***Recall***	***F1-Score***
Opprentice	0.72	0.66	0.69	0.78	0.70	0.74
Donut	0.83	0.76	0.79	0.86	0.83	0.84
LSTM-VAE	0.91	0.84	0.87	0.90	0.85	0.87
VAE-SVDD	0.95	0.96	0.95	0.97	0.96	0.96

VAE-SVDD uses VAE to reconstruct KPI data and uses SVDD to train the reconstruction error again. It is necessary to compare the complexity of VAE-SVDD with other detection methods. This paper records the anomaly detection time of various anomaly detection methods. Table 4 shows the average duration of 5 anomaly detections performed by each method on the test set (containing 26,358 data). The Opprentice method is based on a random forest, and it can improve efficiency through parallelization, so the detection time is short. Donut based on VAE is more complex than the Opprentice method, resulting in slightly longer detection time. LSTM-VAE integrates LSTM and VAE, which evidently leads to a longer detection time. In VAE-SVDD, the encoder and decoder of VAE were designed as BiLSTM, and the threshold was determined by SVDD. However, these optimization mechanisms also increased the detection time. Although the detection time of VAE-SVDD was slightly longer than other methods, the detection was more accurate. Therefore, it is still acceptable in the actual situation.

Table 4. Average detection time.

Evaluation Index	**Opprentice**	**Donut**	**LSTM-VAE**	**VAE-SVDD**
Detection time (s)	34.5	46.3	53.8	65.2

4.6. Effects of Different Components

4.6.1. Time Correlation

In this paper, BiLSTM network is used as the encoder and decoder of VAE, which can better capture the time correlation of sequence data. We compared the distribution of latent variables between VAE and BiLSTM-VAE to prove that time correlation has a positive effect. In order to verify this time correlation, part of the test set containing anomalies was selected. At the same time, the hours of timestamp were extracted as labels, that is, there were 24 types of labels.

In order to facilitate visualization, we used a principal component analysis (PCA) [40] and t-distributed Stochastic Neighbor Embedding (t-SNE) [41] to reduce the dimension of latent variables to 2. Latent variables can be regarded as the characteristic representation of data. PCA replaces original 10 features with a smaller number of 2 features. New features are the linear combination of old features. These linear combinations maximize the sample variance and try to make new features irrelevant to each other. The mapping from old features to new features captures the inherent variability in the data. In addition, it has the advantage of being quick and easy to implement. However, the nonlinear correlation between samples may be lost after linear dimension reduction using PCA. In contrast, t-SNE is a nonlinear dimension reduction method. It converts the similarity of data points into joint probability and optimizes the *KL* error between low-dimensional data and high-dimensional data. t-SNE dimension reduction can not only maintain the difference of data, but also maintain the local structure of data. However, the results of t-SNE have a certain degree of randomness, rather than the consistency of PCA results. Therefore, we combined the two methods to visualize latent variables after dimension reduction. It can be more reasonable to prove that our method better captures the potential pattern of data, that is, time correlation.

Figure 14 shows the two-dimensional visualization effect of VAE latent variables on KPI 1. Figure 15 shows the two-dimensional visualization effect of BilSTM-VAE latent variables on KPI 1. Figure 16 shows the two-dimensional visualization effect of VAE latent variables on KPI 2. Figure 17 shows the two-dimensional visualization effect of BilSTM-VAE latent variables on KPI 2. All subgraphs (a) are the visualization effect of the PCA on latent variables after dimension reduction. All subgraphs (b) are the visualization effect of t-SNE on latent variables after dimension reduction. Evidently, the latent variables distribution of BiLSTM-VAE is more regular than VAE. It was proved that BiLSTM-VAE captures the time correlation of sequences better than VAE. It can be observed from the figure that the latent variables of the time-aligned sequence are roughly in the same area, and the anomaly moment will show a large deviation (such as 8 o'clock). The literature [5] explains this effect for the first time, which is called time gradient.

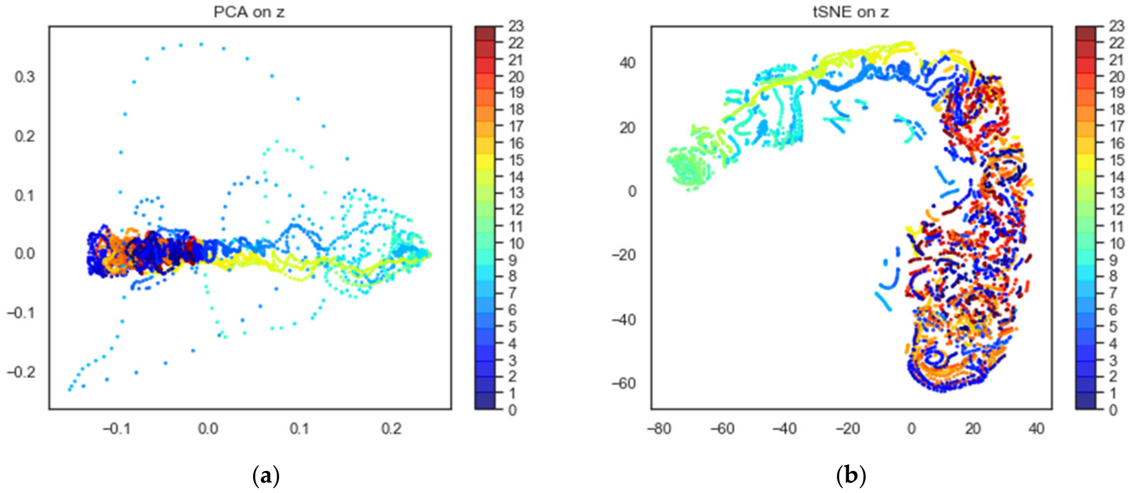
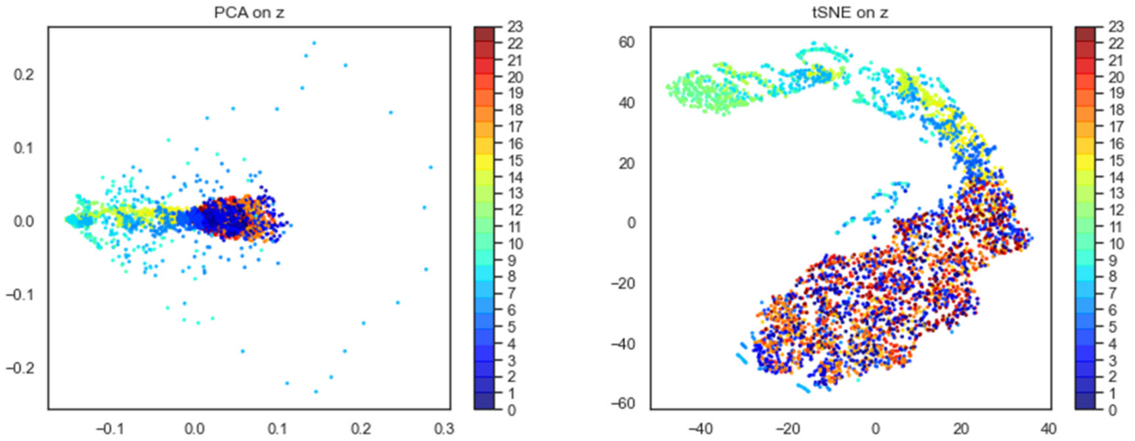
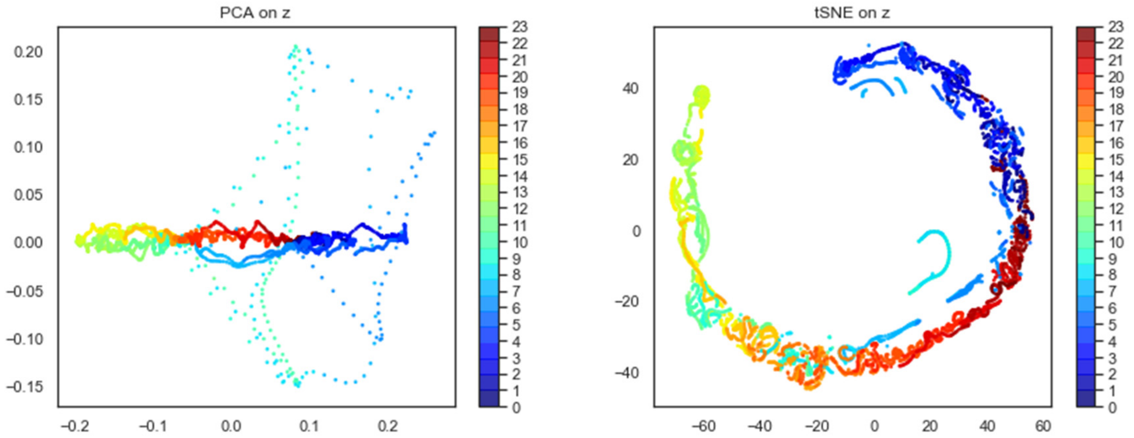


Figure 14. Two-dimensional visualization of the VAE latent variables for KPI 1: (**a**) PCA on z; and (**b**) t-SNE on z.



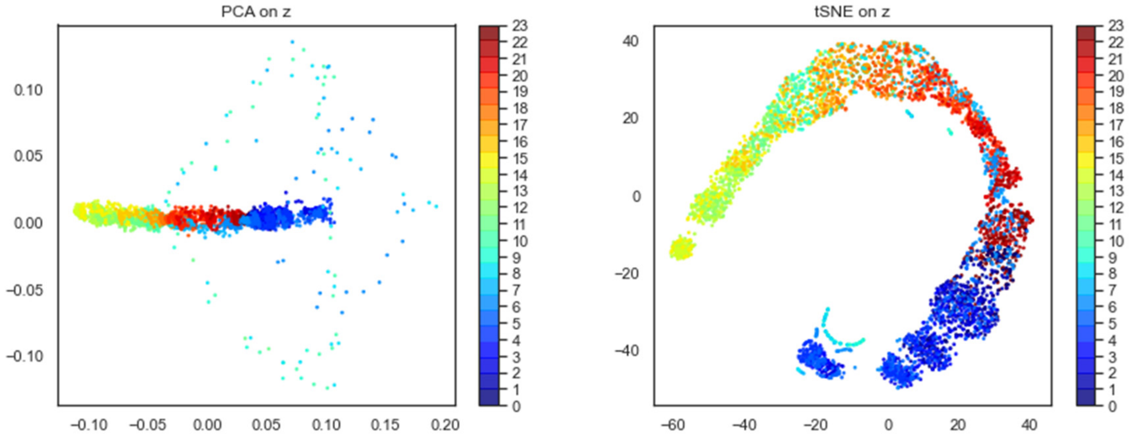
(**a**) (**b**)

Figure 15. Two-dimensional visualization of the BiLSTM-VAE latent variables for KPI 1: (**a**) PCA on z; and (**b**) t-SNE on z.



(**a**) (**b**)

Figure 16. Two-dimensional visualization of the VAE latent variables for KPI 2: (**a**) PCA on z; and (**b**) t-SNE on z.

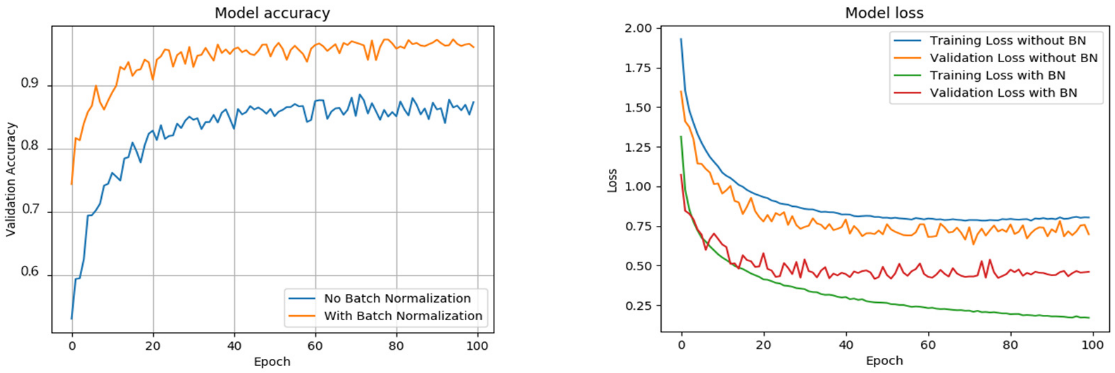


(**a**) (**b**)

Figure 17. Two-dimensional visualization of the BiLSTM-VAE latent variables for KPI 2: (**a**) PCA on z; and (**b**) t-SNE on z.

4.6.2. Batch Normalization

We used batch normalization to prevent the disappearance of the *KL* divergence during model training. In order to highlight the effect of batch normalization, we visualized the loss and accuracy of the model during training. As shown in Figure 18a, the accuracy of the model with BN (97%) is significantly higher than that of the model without BN (87%). From the perspective of training speed, the model with BN is already very close to the final convergence at the 22nd iteration. The model without BN is close to the final convergence at the 42nd time, indicating that the model with BN is faster. As shown in Figure 18b, the model with BN is lower than the model without BN, regardless of the loss of training set or test set. In conclusion, BN can accelerate the training speed of the model and even play a positive role in improving accuracy and reducing loss.



(**a**) (**b**)

Figure 18. Comparative effects of batch normalization are introduced during training: (**a**) Model accuracy; and (**b**) Model loss.

4.6.3. EWMA Smoothing

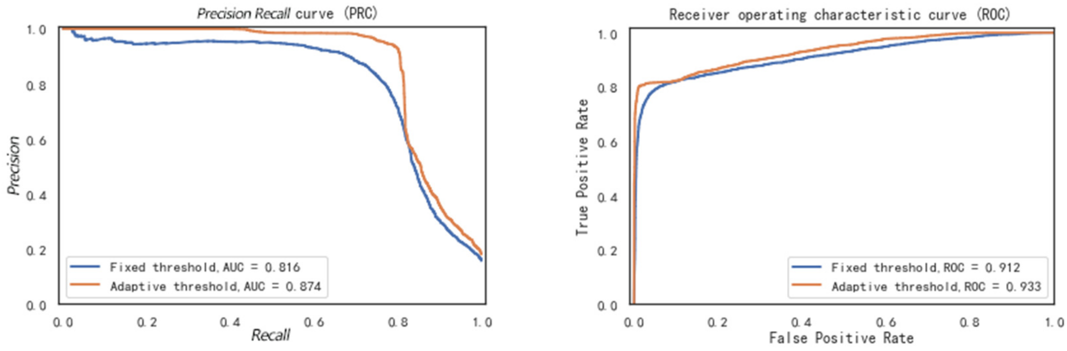
We used EWMA to smooth the error sequence, which reduces false positives and false negatives during the detection process. To prove that the smoothing operation can improve the effectiveness of anomaly detection, we compared the detection effects of no smoothing and EWMA smoothing on two KPIs, as shown in Table 5. As can be seen from the table, *Precision*, *Recall*, and *F*1-*score* all improved greatly after EWMA smoothing.

Table 5. EWMA smoothing effect.

Dataset	Method	*Precision*	*Recall*	*F*1-*Score*
KPI 1	No smoothing	0.88	0.85	0.86
	EWMA smoothing	0.95	0.96	0.95
KPI 2	No smoothing	0.91	0.88	0.89
	EWMA smoothing	0.97	0.96	0.96

4.6.4. Adaptive Threshold

We used SVDD to determine the threshold of anomaly detection, so that different thresholds can be set adaptively for different KPIs. Figure 19 shows the PRC curve and ROC curve of the two threshold methods, proving that the adaptive threshold has a better effect than the fixed threshold. The larger the area under the curve is, the better the model effect is. Compared with the traditional fixed threshold, the adaptive threshold not only has a higher precision rate and recall rate, but also produces a higher true positive rate (TPR) under the same false positive rate (FPR).



(**a**) (**b**)

Figure 19. Comparison of the adaptive threshold and fixed threshold: (**a**) PRC; and (**b**) ROC.

5. Conclusions

In this paper, a novel KPI anomaly detection method is proposed by combining VAE and SVDD. In this method, firstly, the encoder and decoder of VAE were designed as BiLSTM to capture the time dependence of data. Then, batch normalization was used on the output of the encoder to prevent the *KL* divergence from disappearing. In addition, EWMA was used to smooth reconstruction errors to eliminate accidental error peaks. Finally, smoothed reconstruction error sequences were put into the SVDD for training to determine the threshold of anomaly detection adaptively. In the experiment, the appropriate sliding window length and latent variable dimension were selected. The visualization effect of latent variables showed that time-aligned sequences are in the same region of latent variables space, and the model can better capture the time correlation of sequences. Batch normalization can speed up training and reduce loss. The reconstruction error after smoothing can reduce false positives and false negatives in the detection to some extent. Compared with the fixed threshold, the adaptive threshold has more flexibility and a better effect. The comparison result with current advanced baseline methods shows that the method in this paper has a better detection effect. Moreover, although the method in this paper is applied to KPI's univariate time series, it is also applicable to multivariate time series. The adaptive threshold can be applied not only to reconstruction errors, but also effectively to prediction errors.

In the future, we will continue our work focusing on the following two aspects:

(1) The linear interpolation method is too simple. When there are many missing values, some errors may be caused. Next, we will explore interpolation methods that can handle both linear and nonlinear data, such as modeling interpolation.

(2) The duration of anomaly detection is important. Next, we can improve the VAE-SVDD model structure and adjust parameters to obtain better performance.

Author Contributions: Y.Z., conceptualization, methodology, investigation, validation, formal analysis, writing—original draft, and writing—review and editing; X.Z., supervision, validation, formal analysis, and writing—review and editing; Z.C., supervision, validation, formal analysis, and writing—review and editing; Z.S., supervision, validation, formal analysis, and writing—review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partly supported by the National Key R & D Program of China (Grant No. 2018YFB1601502) and the LiaoNing Revitalization Talents Program (Grant No. XLYC1902071).

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon request.

Acknowledgments: The authors would like to thank all anonymous reviewers and editors for their helpful suggestions for the improvement of this paper.

Conflicts of Interest: The authors declare that they have no known competing financial interest or personal relationship that could appear to influence the work reported in this paper.

References

1. Pei, D.; Zhang, S.; Pei, C. Intelligent operation and maintenance based on machine learning. *Commun. CCF* **2017**, *13*, 68–72.
2. Kingma, D.P.; Welling, M. Auto-encoding variational bayes. In Proceedings of the 2nd International Conference on Learning Representations (ICLR), Banff, AB, Canada, 14–16 April 2014.
3. Goodfellow, I.J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative Adversarial Nets. In Proceedings of the 28th Conference on Neural Information Processing Systems (NIPS), Montreal, QC, Canada, 8–13 December 2014; pp. 2672–2680.
4. Huang, Y.; Li, Y.; Liu, Y.; Jing, R.; Li, M. A Multiple Comprehensive Analysis of scATAC-seq Based on Auto-Encoder and Matrix Decomposition. *Symmetry* **2021**, *13*, 1467. [CrossRef]
5. Xu, H.; Feng, Y.; Chen, J.; Wang, Z.; Qiao, H.; Chen, W. Unsupervised Anomaly Detection via Variational Auto-Encoder for Seasonal KPIs in Web Applications. In Proceedings of the 27th World Wide Web (WWW) Conference, Lyon, France, 23–27 April 2018; pp. 187–196.
6. Chen, W.; Xu, H.; Li, Z.; Pei, D.; Chen, J.; Qiao, H. Unsupervised Anomaly Detection for Intricate KPIs via Adversarial Training of VAE. In Proceedings of the IEEE Conference on Computer Communications (IEEE INFOCOM), Paris, France, 29 April–2 May 2019; pp. 1891–1899.
7. Hochreiter, S.; Jürgen, S. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
8. Daehyung, P.; Hoshi, Y.; Kemp, C.C. A Multimodal Anomaly Detector for Robot-Assisted Feeding Using an LSTM-based Variational Autoencoder. *IEEE Robot. Autom. Lett.* **2017**, *3*, 1544–1551.
9. Niu, Z.; Yu, K.; Wu, X. LSTM-Based VAE-GAN for Time-Series Anomaly Detection. *Sensors* **2020**, *20*, 3738. [CrossRef]
10. Bowman, S.R.; Vilnis, L.; Vinyals, O.; Dai, A.M.; Bengio, S. Generating Sentences from a Continuous Space. In Proceedings of the 20th Conference on Computational Natural Language Learning (CoNLL), Berlin, Germany, 11–12 August 2016; pp. 10–21.
11. Kingma, D.P.; Salimans, T.; Jozefowicz, R.; Chen, X.; Sutskever, I.; Welling, M. Improving Variational Inference with Inverse Autoregressive Flow. In Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS), Barcelona, Spain, 5–10 December 2016.
12. Xu, J.; Durrett, G. Spherical Latent Spaces for Stable Variational Autoencoders. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 4503–4513.
13. Yang, Z.; Hu, Z.; Salakhutdinov, R.; Berg-Kirkpatrick, T. Improved Variational Autoencoders for Text Modeling using Dilated Convolutions. In Proceedings of the 34th International Conference on Machine Learning (ICML), Sydney, Australia, 6–11 August 2017; pp. 3881–3890.
14. An, J.; Cho, S. Variational autoencoder based anomaly detection using reconstruction probability. *Spec. Lect. IE* **2015**, *2*, 1–18.
15. Suh, S.; Chae, D.H.; Kang, H.G.; Choi, S. Echo-state conditional variational autoencoder for anomaly detection. In Proceedings of the International Joint Conference on Neural Networks (IJCNN), Vancouver, BC, Canada, 24–29 July 2016; pp. 1015–1022.
16. Zong, B.; Song, Q.; Min, M.R.; Cheng, W.; Lumezanu, C.; Cho, D.; Chen, H. Deep autoencoding Gaussian mixture model for unsupervised anomaly detection. In Proceedings of the 6th International Conference on Learning Representations (ICLR), Vancouver, BC, Canada, 30 April–3 May 2018.
17. Graves, A.; Schmidhuber, J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Netw.* **2005**, *18*, 602–610. [CrossRef]
18. Zhu, Q.L.; Bi, W.; Liu, X.J.; Ma, X.Y.; Li, X.L.; Wu, D.P. A Batch Normalized Inference Network Keeps the KL Vanishing Away. In Proceedings of the 58th Annual Meeting of the Association-for-Computational-Linguistics (ACL), Seattle, WA, USA, 5–10 July 2020; pp. 2636–2649.
19. Hunter, J. The Exponentially Weighted Moving Average. *J. Qual. Technol.* **1986**, *18*, 19–25. [CrossRef]
20. Tax, D.M.; Duin, R.P. Support Vector Data Description. *Mach. Learn.* **2004**, *54*, 45–66. [CrossRef]
21. Pukelsheim, F. The three sigma rule. *Am. Stat.* **1994**, *48*, 88–91.
22. Yaacob, A.H.; Tan, I.K.; Chien, S.F.; Tan, H.K. ARIMA Based Network Anomaly Detection. In Proceedings of the Second International Conference on Communication Software and Networks (ICCSN), Singapore, 26–28 February 2010; pp. 205–209.
23. Yu, Q.; Jibin, L.; Jiang, L.R. An improved ARIMA-based traffic anomaly detection algorithm for wireless sensor networks. *Int. J. Distrib. Sens. Netw.* **2016**, *12*, 9653230. [CrossRef]
24. Kalekar, P. Time series forecasting using Holt-Winters exponential smoothing. *Kanwal Rekhi Sch. Inf. Technol.* **2004**, *4329008*, 1–13.
25. Laptev, N.; Amizadeh, S.; Flint, I. Generic and Scalable Framework for Automated Time-series Anomaly Detection. In Proceedings of the 21st ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), Sydney, Australia, 10–13 August 2015; pp. 1939–1947.
26. Liu, D.P.; Zhao, Y.J.; Xu, H.W.; Sun, Y.Q.; Pei, D.; Luo, J.; Jing, X.W.; Feng, M. Opprentice: Towards Practical and Automatic Anomaly Detection Through Machine Learning. In Proceedings of the ACM Internet Measurement Conference(IMC), Tokyo, Japan, 28–30 October 2015; pp. 211–224.
27. Erfani, S.M.; Rajasegarar, S.; Karunasekera, S.; Leckie, C. High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning. *Pattern Recognit.* **2016**, *58*, 121–134. [CrossRef]
28. Wazid, M.; Das, A.K. An Efficient Hybrid Anomaly Detection Scheme Using K-Means Clustering for Wireless Sensor Networks. *Wirel. Pers. Commun.* **2016**, *90*, 1971–2000. [CrossRef]

29. Laxhammar, R.; Falkman, G.; Sviestins, E. Anomaly detection in sea traffic—A comparison of the Gaussian Mixture Model and the Kernel Density Estimator. In Proceedings of the 12th International Conference on Information Fusion(FUSION), Seattle, WA, USA, 6–9 July 2009; pp. 756–763.
30. Wang, X.H.; Du, Y.; Lin, S.J.; Cui, P.; Shen, Y.T.; Yang, Y.P. adVAE: A self-adversarial variational autoencoder with Gaussian anomaly prior knowledge for anomaly detection. *Knowl.-Based Syst.* **2020**, *190*, 105187. [CrossRef]
31. Luo, P.; Wang, B.H.; Li, T.Y.; Tian, J.W. ADS-B anomaly data detection model based on VAE-SVDD. *Comput. Secur.* **2021**, *104*, 102213. [CrossRef]
32. Qiu, J.; Du, Q.F.; Qian, C.S. KPI-TSAD: A Time-Series Anomaly Detector for KPI Monitoring in Cloud Applications. *Symmetry* **2019**, *11*, 1350. [CrossRef]
33. Chen, R.Q.; Shi, G.H.; Zhao, W.L.; Liang, C.H. A Joint Model for IT Operation Series Prediction and Anomaly Detection. *Neurocomputing* **2021**, *448*, 130–139. [CrossRef]
34. Wang, J.Y.; Jing, Y.H.; Qi, Q.; Feng, T.T.; Liao, J.X. ALSR: An adaptive label screening and relearning approach for interval-oriented anomaly detection. *Expert Syst. Appl.* **2019**, *136*, 94–104. [CrossRef]
35. Li, J.; Di, S.; Shen, Y.; Chen, L. FluxEV: A Fast and Effective Unsupervised Framework for Time-Series Anomaly Detection. In Proceedings of the Fourteenth International Conference on Web Search and Data Mining (WSDM), Virtual Event, Israel, 8–12 March 2021; pp. 824–832.
36. Siffer, A.; Fouque, P.A.; Termier, A.; Largouet, C. Anomaly Detection in Streams with Extreme Value Theory. In Proceedings of the 23rd International Conference on Knowledge Discovery and Data Mining (KDD), Halifax, NS, Canada, 13–17 August 2017; pp. 1067–1075.
37. Yang, D. Influences of different interpolation methods on GPS time series. *Gnss World China* **2019**, *44*, 66–69.
38. Hundman, K.; Constantinou, V.; Laporte, C.; Colwell, I.; Soderstrom, T. Detecting Spacecraft Anomalies Using LSTMs and Nonparametric Dynamic Thresholding. In Proceedings of the 24th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), London, UK, 19–23 August 2018; pp. 387–395.
39. Robinson, J.; Rahmat-Samii, Y. Particle swarm optimization in electromagnetics. *IEEE Trans. Antennas Propag.* **2004**, *52*, 397–407. [CrossRef]
40. Bro, R.; Smilde, A.K. Principal Component Analysis. *J. Mark. Res.* **2014**, *6*, 2812–2831. [CrossRef]
41. Laurens, V.D.M.; Hinton, G. Visualizing Data using t-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.