



Article

Labeling Expert: A New Multi-Network Anomaly Detection Architecture Based on LNN-RLSTM

Xiaoyu Tang , Sijia Xu and Hui Ye 

School of Science, Jiangsu University of Science and Technology, Zhenjiang 212000, China

* Correspondence: yehui@just.edu.cn

Abstract: In network edge computing scenarios, close monitoring of network data and anomaly detection is critical for Internet services. Although a variety of anomaly detectors have been proposed by many scholars, few of these take into account the anomalies of the data in business logic. Expert labeling of business logic exceptions is also very important for detection. Most exception detection algorithms focus on problems, such as numerical exceptions, missed exceptions and false exceptions, but they ignore the existence of business logic exceptions, which brings a whole new challenge to exception detection. Moreover, anomaly detection in the context of big data is limited to the need to manually adjust detector parameters and thresholds, which is constrained by the physiological limits of operators. In this paper, a neural network algorithm based on the combination of Labeling Neural Network and Relevant Long Short-Term Memory Neural Network is proposed. This is a semi-supervised exception detection algorithm that can be readily extended with business logic exception types. The self-learning performance of this multi-network is better adapted to the big data anomaly detection scenario, which further improves the efficiency and accuracy of network data anomaly detection and considers business scenario-based anomaly data detection. The results show that the algorithm achieves 96% detection accuracy and 97% recall rate, which are consistent with the business logic anomaly fragments marked by experts. Both theoretical analysis and simulation experiments verify its effectiveness.



Citation: Tang, X.; Xu, S.; Ye, H. Labeling Expert: A New Multi-Network Anomaly Detection Architecture Based on LNN-RLSTM. *Appl. Sci.* **2023**, *13*, 581. <https://doi.org/10.3390/app13010581>

Academic Editors: Yuehua Cheng, Qingxian Jia, Guang Jin and Yuqing Li

Received: 19 November 2022
Revised: 15 December 2022
Accepted: 17 December 2022
Published: 31 December 2022



Copyright: © 2022 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/>).

Keywords: anomaly detection; labeling neural network; relevant long short-term memory; compressing window; quartile method; KPI; deep learning

1. Introduction

Anomaly detection is an important task in wireless network data analysis and management, aiming to discover abnormal behavior or abnormal states in the data. Real-time anomaly detection of network data helps to improve the intelligent operation and maintenance of the network and achieve optimal allocation and on-demand scheduling of network resources. Accurate anomaly detection can trigger timely troubleshooting and help avoid revenue loss, thus, maintaining the company's reputation and brand. For this reason, large network companies have built their own anomaly detection services to monitor the health of their business, products and services.

Currently, most researchers focus on studying network attack anomalies and industrial data anomaly attacks. However, few researchers have conducted research on non-intrusive network data anomalies. This paper focuses on anomaly detection of user behavior indices in the process of network monitoring, which belongs to a different field from network attack anomaly detection, such as Dos attack detection. KPI means "Key Performance Indicator". The network KPIs discussed in this paper are, specifically: the average number of users, PDCP (Packet Data Convergence Protocol) traffic and the average number of activated users. In network KPI monitoring scenarios, anomaly detection tasks are calculated at the closest location to the data source to reduce end-to-end latency in mobile service delivery and improve the inherent capabilities of the wireless network [1]. When anomalies are detected, the monitoring

software will send the network alert data to the operator for timely incident-related decisions [2]. However, anomaly detection for KPI is a typical big data environment where anomaly detection is generally inefficient due to the constraints of operators' physiological limits. Moreover, KPIs are presented in the form of time-series data, which makes them different from detection in non-time-series data and images [3]. It should be emphasized that the detection method proposed in this paper is mainly used to identify routine data anomalies, and network attack anomalies are not in the scope of this paper.

In general, there are many challenges in designing services for time-series anomaly detection.

1.1. Challenge 1: Lack of Labeling

When providing anomaly detection services for a single business scenario, the anomaly detection system will be faced with handling millions of time series. At this point, it is not possible for operators to manually tag each time series, and if all of them are manually tagged, they will also be affected by the efficiency and will not be applicable to the big data operation and maintenance environment. In addition, an effective anomaly detection system not only needs to be able to accurately identify known types of anomalies, but also needs to have the ability to detect anomalies in unknown situations. Currently, although supervised anomaly detection methods based on deep learning have emerged, they are unable to effectively detect and cope with the ever-changing and complex forms of the current network environment because they are trained with training datasets of already-labeled anomaly types [4]. In summary, in the face of anomaly detection in the current network environment, traditional supervisory methods not only fail to obtain sufficient labels to ensure accuracy, but also fail to address the challenges of new types of anomalies.

1.2. Challenge 2: Lack of Professionalism

There are many anomalies in the network anomaly detection process that cannot be identified by purely mathematical algorithms (e.g., isolated forests), as shown in Figure 1b. This is because the predefined anomalies or anomaly cycles in anomaly detection are difficult to describe accurately by exact mathematical formulas [5]. Operators usually identify anomalies based on expertise with their own understanding of KPIs. Most of the current unsupervised anomaly detection (e.g., isolated forest, LOF, etc.) perform anomaly detection from a statistical perspective, ignoring the hidden features in the network data, which leads to unsupervised detection of anomalies only at a simple numerical level, ignoring the actual meaning of the anomalies themselves. At the same time, the unsupervised anomaly detection method must satisfy two conditions: (1) most data in the dataset must be normal data; (2) the anomalous data must be significantly different from the normal data. Due to these two constraints, unsupervised anomaly detection is not only ineffective in detecting anomalies for anomalous cycles with many outliers, but also causes misjudgment of normal values due to the bias caused by it.

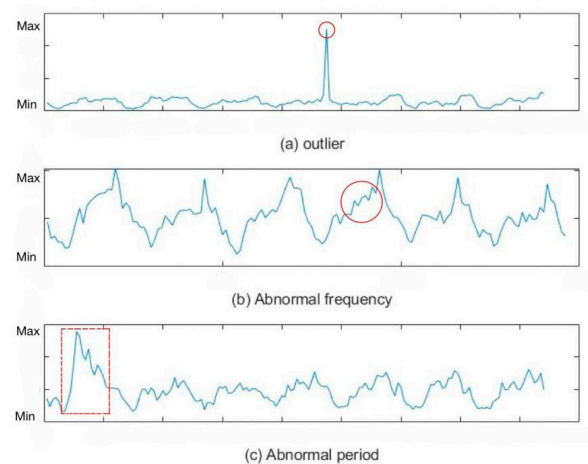


Figure 1. View of various exceptions about KPI.

1.3. Challenge 3: Lack of Efficiency

During network monitoring, the monitoring system must process millions or even billions of time series in real time. Especially for minute time series, the process of anomaly detection needs to be completed in a limited amount of time. Therefore, efficiency is one of the necessary prerequisites for online anomaly detection services. However, current anomaly detection methods are often computationally huge, and for supervised types of anomaly detection methods based on Bayes networks, neural networks for network intrusion detection require many samples [6]; support vector machine-based network detection has a high correct rate, but the time taken for anomaly detection is long and the real-time performance of network anomaly detection cannot be guaranteed [7]. These imply that supervised anomaly detection models, even though they can do well in terms of accuracy, have poor timeliness in online scenarios due to their large time complexity and space complexity.

Based on the three types of problems presented above, a novel hybrid anomaly detection network structure is designed, LNN-RLSTM, which is a semi-supervised anomaly detection method. The reason to design semi-supervised is to adapt to the real-time monitoring application scenario where new anomaly types are constantly emerging. To address the issue of business logic exceptions, this article divides the categories of detection encountered in the business domain into anomaly isolated points and exception cycles. This algorithm solves the problem of few initial labels by combining the compressed window with the quadrature method. In terms of lack of specialization, artificial labels are added in the second-generation training set of the network, and by introducing artificial label data, LNN-RLSTM further solves the anomaly detection problem at the business level. Finally, to solve the problem of detection training time, artificial labels pre-training on the LNN-RLSTM network are pre-trained, so that the frequency of manual intervention for anomaly detection is continuously reduced and converges to a tiny value, thus, reducing the amount of model post-tuning, which, in turn, reduces the detection time and improves the detection in real time.

The main contributions of this paper are as follows:

- (1) A semi-supervised dual (LNN and RLSTM) neural network for detecting an anomaly in the time-series data. LNN was devised to train a neural network that can identify and label outliers in a “sliding window” and label individual data points according to the data distribution in a local range, while RLSTM can combine test results and solve “detect local fragmentation problems”.
- (2) The concept of “compression window” is proposed to solve the problem of “anomaly clustering”. The time complexity and space complexity of this algorithm are effectively reduced to $O(\text{length of window})$.
- (3) An improved LSTM model by adding a “correlation gate” as RLSTM to obtain the final outcome, which is adaptable to anomaly detection in a long time span.
- (4) The experiments were conducted on two separate sets of data. Network traffic detection data are extracted from Kaggle and China Big Data Challenge datasets to verify the effectiveness and hardness of the algorithms. The effectiveness and sensitivity of the detector are verified in terms of accuracy and false-positive rate.
- (5) The algorithm incorporates the process of manual tag input, effectively solving the problem of lack of business logic in detection.

The rest of this article is organized as follows. Section 2 introduces the literature review and related work in anomaly detection. Section 3 describes drawbacks of the existing method and our research goals are proposed in the meantime. Section 4 describes the LNN-RLSTM method. Section 5 introduces the labeling neural network (LNN) in detail. Section 6 introduces relevant long short-term memory (RLSTM) in detail. Section 7 describes the data flow of LNN-RLSTM globally. Results and discussion are given in Section 8. Section 9 provides the conclusion and directions for future work.

2. Literature Review

Threats to validity: In this literature review, we include studies that (1) address algorithmic research in three areas of anomaly detection; (2) anomaly detection research for cyber threats, including intrusion detection, spam detection and malware detection; and (3) discuss performance evaluation for accuracy, recall, etc. We used various string combinations, such as “anomaly detection” and “machine learning and network intrusion” [8], “business data anomalies” and “network traffic”, to retrieve peer-reviewed articles in journals, conference proceedings, book chapters and reports. We targeted five databases, namely ACM Digital Library, IEEE Xplore, ScienceDirect, SpringerLink and Web of Science. Google Scholar was also used for forward and backward searches. We focused on the latest advances in the last decade. A total of 154 papers was retrieved, and titles and abstracts were screened to identify potential articles. The full texts of 56 studies were evaluated for relevance to the inclusion criteria. We excluded articles that discussed (1) cyber threats, such as image-mapping-based intrusion detection and spam detection, (2) threats to cyber-physical systems, and (3) threats to IoT devices, smart grids, and smart cities, by searching before and after. Finally, a total of 50 studies was selected for data extraction.

There are many different machine learning techniques used to improve anomaly detection performance, which can be generally classified as supervised, unsupervised, semi-supervised and statistical-based models. Supervised methods require extremely high quality of data labels, such as K-nearest neighbors, neural networks and support vector machines. However, during the real-time monitoring of the network, new types of anomalies often appear frequently, which are not present in the training set, which causes anomaly detection failure. Unsupervised learning uses numerical analysis of data attributes alone for detection, for example, by filtering outliers through clustering methods. Since this detection method is limited to anomalies from a statistical perspective only, it often does not meet the detection requirements of the business environment. Semi-supervised learning methods are more efficient than supervised learning through semi-learning and can outperform unsupervised learning in the business detection perspective. Statistical models were the first algorithms to be used for anomaly detection. They are mostly based on the comparison of some statistical properties to determine the outliers.

To solve the three types of problems in Section 1, an accurate, efficient and versatile anomaly detection system needs to be developed. Traditional network anomaly detection has been concluded below. Some details of related work are summarized in Table 1.

Table 1. Anomaly detection-related work summary.

Categories	Technique	Year	Domain	Ref	Time Complexity	Recall	Accuracy
Supervised	SVM	1963	Cybersecurity	[9]	$O(n^2)^1$	-	-
Supervised	Naive Bayes	1960	Cybersecurity	[10]	$O(mn)^2$	-	-
Supervised	Random forest	2001	Cybersecurity	[11]	$O(O(Mm\log n))^3$	-	-
Supervised	ANN	2000	Cybersecurity	[12]	$O(emnk)^4$	-	-
Supervised	Decision Tree	1979	Cybersecurity	[13]	$O(mn^2)^5$	-	-
Supervised	SVM	2011	Email Spam	[14]	$O(n^2)^1$	95.00%	96.90%
Unsupervised	DBN	2016	Email Spam	[15]	$O((n + N)k)^7$	95.61%	95.86%
Supervised	Decision Tree	2016	Email Spam	[16]	$O(mn^2)^5$	94.00%	96.00%
Supervised	Decision Tree	2014	Email Spam	[17]	$O(mn^2)^5$	88.08%	92.08%
Supervised	SVM	2018	Spam Tweets	[18]	$O(n^2)^1$	93.14%	93.14%
Unsupervised	K-means	2007	Software	[19]	-	-	-
Unsupervised	ADA	2020	service disruption	[20]	-	93.00%	94.00%
Semi-supervised	FCM	2015	Network	[21]	-	-	-
Semi-supervised	CBLOF	2019	Financial Transaction	[22]	-	-	95.79%

2.1. Supervised Anomaly Detection

Supervised learning-based anomaly detection methods work mainly through machine learning methods to study and analyze the existing training set of completed labeled network data to find out the laws contained in it and how to apply this law for anomaly detection. Supervised methods require a label for each of the data points in the training dataset. They use this information to learn the differences between normal and outlier instances by either discriminative or generative algorithms. Theoretically, supervised abnormality detection is superior in overall accuracy due to a clear understanding of normality and abnormality. Neural network algorithms have proven to be robust and highly accurate in applications, such as the prediction of medical time-series data [23].

Methods, such as Linear regression model parameter estimation [20], design and implementation of a network anomaly detection system based on weighted plain Bayes [24] and research on deep neural network video anomaly target detection [25], use linear regression, Bayesian algorithm and neural network, respectively, and these methods can effectively ensure the accuracy of anomaly detection in different scenarios. However, the method is extremely dependent on the training set, and the accuracy of the model is easily affected if there are unlabeled anomalies. Moreover, for online network anomaly detection, it is simply impossible to perform enough labeling, so the supervised learning network anomaly detection method is difficult to adapt to online network anomaly detection.

Several practical problems greatly hinder its use. Firstly, in the context of big data, the labels of much of the data are unknown, which poses a challenge for supervised anomaly detection algorithms. Data labels are often unavailable or too costly to obtain, e.g., the labeling of network configurations is not completely clear unless the network is run and inspected. Secondly, data labels may be unbalanced. Often, in practical anomaly detection problems, the number of normal samples greatly exceeds the number of anomalous samples, which leads to prominent biases in the classification model and may degrade the performance of anomaly detection [26].

2.2. Unsupervised Anomaly Detection

The unsupervised learning-based anomaly detection method refers to the input of an unlabeled dataset to study and analyze the network traffic data via the machine learning (ML) method to find out the data structure and features in it and then perform anomaly detection. Unsupervised techniques are typically employed in a situation where no prior knowledge of the dataset is known [26].

Methods, such as variable selection and anomaly detection for automatic k-means clustering [19], an improved hierarchy-based clustering and anomaly detection algorithm and its application to data mining platforms [27] and research on parallel network traffic anomaly detection method based on Spark [28], solve the dependence of supervised learning on markers, but they only analyze from a statistical point of view and ignore the potential features among data, leading to bias due to ignoring the actual meaning of outliers; for example, they cannot solve the situation due to the aggregation of outliers. Therefore, how to address unsupervised learning to learn and link the actual meaning of network data is a key issue to improve online anomaly detection [29].

2.3. Semi-Supervised Anomaly Detection

Technique patterns that operate in semi-supervised mode have only labeled instances of normal classes in the training data. Since they do not require labels of anomaly classes, they are more widely applicable than supervised techniques than supervised ones [30]. For example, in spacecraft fault detection, anomalies imply accidents, which are not easy to model [31]. The typical approach used in such techniques is to build a model normal behavior for the class corresponding to normal behavior and use that model to identify anomalies in the test data [32].

It has been previously proposed to use only anomalous instances for training [33]. This technique is not commonly used, mainly because it is difficult to obtain training datasets

that cover all possible anomalous behaviors in the data [34]. Some algorithms are the fuzzy c-means and probabilistic c-means [35]. Other algorithms include Self-Organizing.

Maps (SOM), Expectation-Maximization, Find Out, CLAD and CBLOF [36] belong to this category. However, based on the present large data situation, anomalies are not difficult to obtain, so this provides a new opportunity for semi-supervised learning.

3. Problem and Goal

In big data anomaly detection, the KPI data tagged by business operators are called “underlying facts” [36]. The primary goal of anomaly detection is accuracy, followed by timeliness. The aim is identifying anomalies that are similar to the “underlying facts” in a short enough time to eliminate false alarms.

Recall $\left(\frac{\text{number of true anomalous points detected}}{\text{number of true anomalous points}}\right)$ and precision $\left(\frac{\text{number of true anomalous points detected}}{\text{number of anomalous points detected}}\right)$ are used to measure the accuracy of anomaly detection in this paper.

For business operators, *Precision* describes something more important than *false-positive rate* (FPR) [37], because most anomalous data points are rare. According to Liu et al. [38], operators not only understand the concept of *recall rate* and *precision* but can also express their tendency to detect anomalous data with accuracy using “*recall* $\geq x$ and *precision* $\geq y$ ”, which comes from the operator’s experience with other detectors.

In this paper, the different accuracy criteria of the business operators will determine the size of the hidden layers and training update frequency of the LNN-RLSTM neural network. The LNN-RLSTM neural network will modify the configuration parameters by the accuracy preference of the business operator. Of course, for the training of the network, the lower the accuracy, the lower the required detection time and complexity will be. For conventional detectors, it is relatively difficult for these detectors to obtain high recall and precision due to the relatively small amount of anomaly data.

For the problem of few samples, Qui [39] et al. proposed VAEGEN oversampling to augment anomalous samples for the problem of few anomalous samples, but it did not increase the data characteristics of the anomalous data, but only augmented the samples from the original anomalous data. Overall, the problem of less identification of anomalous data has not been well solved. In fact, the requirements of recall and precision are usually adjusted according to the actual needs. For example, busy operators are more concerned with accuracy because they do not want to be constantly distracted by many false alarms, while the labeling neural network (LNN) proposed in this paper would be good at filtering out false alarms and allowing operators to focus only on “individual anomalies”. On the other hand, if the operator’s task is to focus only on a single KPI or on a major KPI, he will care more about the value of recall.

In addition, traditional unsupervised machine learning algorithms (e.g., isolated forest, LOF, etc.) will fail when faced with the problem of “clustering of anomalies”, as shown in Figure 2, where the set of anomalies may form an aggregated cluster, making it difficult to distinguish from normal clusters. Based on this, the concept of “anomaly cycle detection” is introduced and tries to solve the anomaly clustering problem using LNN-RLSTM neural networks.

This paper focuses on identifying anomalous behaviors in the KPI time series of user-aware class networks. This is the first solution to the anomaly detection problem using a semi-supervised dual neural network algorithm and the first approach to label data at the business and data layers using labeled neural networks. However, the user-aware class of network KPIs discussed in this paper is somewhat seasonal, and other forms of KPI data types are not reflected in the practice of the algorithm in this paper, but this does not mean that our proposed LNN-RLSTM neural network model is not applicable to the unstable big data environment. On the contrary, it is also possible to make our model applicable to other data types by adjusting the window parameters and other methods, but we do not elaborate on them in this paper.

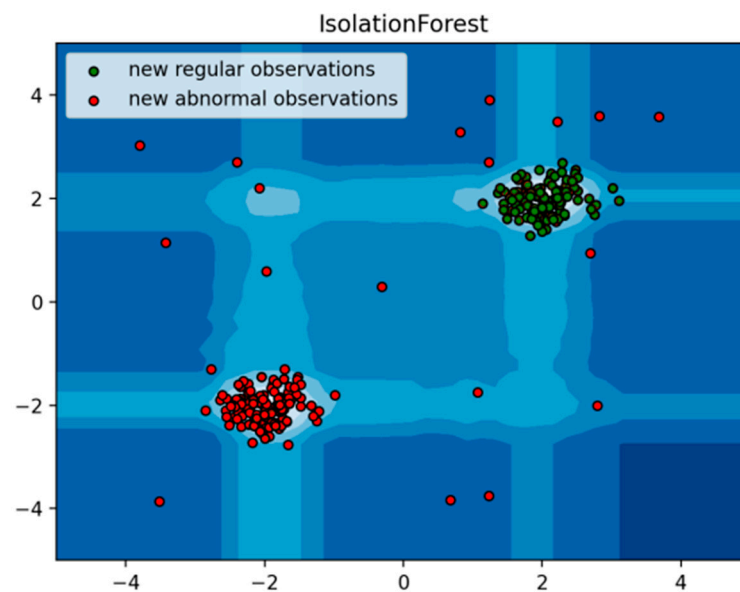


Figure 2. On the phenomenon of clustering of anomalies in isolated forest.

4. Network Traffic Anomaly Detection Scheme Based on LNN-RLSTM Neural Network

Our proposed multi-neural network fusion model is a combination of Labeling Neural Networks and Long Short-Term Memory (LSTM) networks. The essence of the labeling neural network is a BP neural network. The labeling neural network is proposed to train a neural network that can automatically identify and label outliers in a sliding window, which can automatically label individual data points according to the data distribution in a local range. The training set for the labeling neural network is derived from three sources.

Source 1: Anomalous isolated points after initial screening.

Source 2: Anomalous cycles after decompression.

Source 3: Anomalous isolated points or anomalous cycles selected manually.

Then, the labeling neural network is applied to label the brand-new time series by continuously iterating the sliding window and then substitute the labeled dataset into the RLSTM neural network for training, and finally obtain an optimal RLSTM network that can perform anomaly detection at the global level. Our proposed dual network model has the following advantages:

- (1) Scientific automatic machine labeling (mainly in the integration of business knowledge and data feature detection).
- (2) Greatly reduces the workload of manual labeling (only manual click anomaly cycle).
- (3) Support for big data anomaly detection and update, network update suitable for big data environment.

4.1. Automatic Labeling of Data

It should be emphasized that there are two sources of training sets for the identification neural network.

Source 1: Abnormal dataset automatically marked by machine.

Source 2: Manually selected anomalous datasets.

Although the initial labeling data only identify these anomalies from a mathematical statistical point of view, when these anomalies are substituted into the labeling neural network for training, the labeling neural network will automatically learn relevant data features other than numerical values. This means that, compared with traditional statistical models, the identification neural network will have the ability to detect anomalies in terms of data features. In addition, with the input of a limited number of manually selected “special case” anomalies, the labeling neural network will have the ability to accurately label each period. Eventually, the labeling neural network can not only identify anomalous

data features, but also learn the expertise of business operators from a business perspective, thus, achieving the characteristics of scientific identification.

Since Zhang [40], Yang [41] et al. developed a software that allows for manual tagging, and this paper will focus on the principle of automatic machine tagging, which divides into two initial tagging steps.

Step 1: Initial labeling of anomalous isolated points.

Step 2: Initial labeling of anomalous cycle points.

After filtering by the statistical anomaly labeling algorithm, the raw data will be automatically labeled. Then, these labels plus the manual input data will train the labeling neural network well. This labeling neural network will replace the statistical initial labeling method after reaching a certain training amount, i.e., the later data labeling will be done automatically by the labeling neural network.

4.2. Initial Labeling of Exception Isolation Points

Anomalous values have a negative impact on the accuracy of the analysis results of intelligent O&M. If the anomalous values are examined through the analysis of the trend components in the time-series model, they will be too large and complex for the operation and maintenance scenarios and the data will lose timeliness. Therefore, for a group of brand-new, unlabeled data, a traditional simple test method is needed for fast initial labeling of the data, which makes big data anomaly detection relevant. As a modification of the “standard deviation method”, the quartile method uses the median and standard interquartile distance to estimate the dataset instead of the mean and standard deviation in the “standard deviation method”, respectively. It has a high robustness and perfectly fits the analysis of a large amount of data in operation and maintenance scenarios. At the same time, it is also free from the constraints of normal distribution and has higher generalizability.

Through the preliminary observation of the data, it can be found that: the user perception class network KPI data have obvious seasonality and seasonal variation by days, and although these time-series data do not conform to the positive-terrestrial distribution from the whole, the positive-terrestrial distribution can be proved for each cycle, as shown in Figure 3, and the positive-terrestrial distribution characteristics of the user perception class KPI data within the cycle can be proved through the QQ chart observation. The quartile method can have good robustness in the face of traditional anomaly detection tasks. Figure 4 illustrates the anomaly detection process for data using the quartic method, which will be subsequently extended and used to identify anomalous cycles.

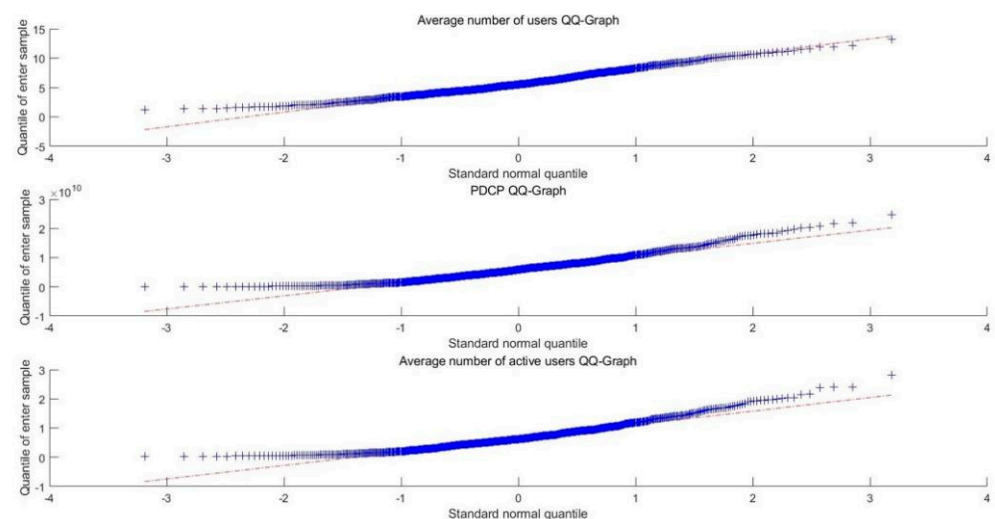


Figure 3. Regarding the QQ chart of various KPI data.

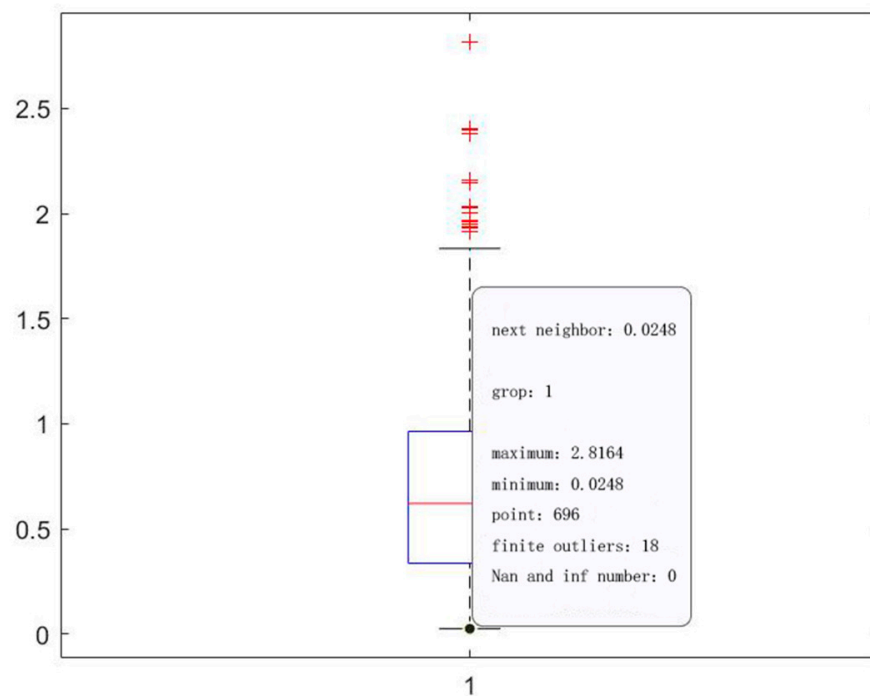


Figure 4. Quartile detection on various KPI data.

Therefore, a sliding window is applied, the data in one cycle at a time for quadrature detection are framed and the data with a preliminary hit are labeled.

4.3. Initial Labeling of Exception Cycles

First, it is necessary to emphasize that a completely new approach to anomaly cycle detection is necessary. Because in the face of a large number of anomalous data points, aggregation, the traditional clustering, LOF, and isolated forest algorithms [14,20,24] will not be able to identify the anomalous data points because the anomalous data clusters are very similar to the normal data clusters, both in terms of aggregation degree and density degree, as shown in Figure 5, the traditional unsupervised learning anomaly detection algorithm is invalid.

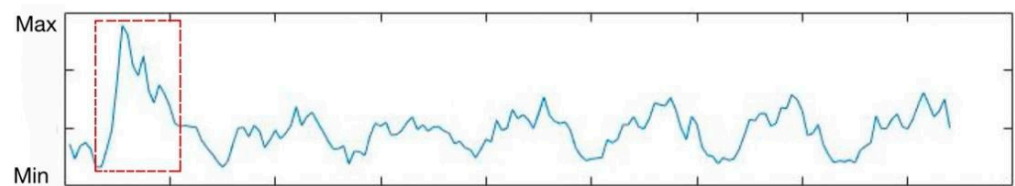


Figure 5. On the gathering of many anomalies.

The phenomenon of “anomaly clustering” occurs when the set of data points is abnormal in a certain period with respect to the whole KPI series. To effectively solve the problem of “anomaly clustering”, the concept of “compression window” is innovatively proposed. This method intends to reduce the anomaly detection problem to the anomaly detection problem by compressing the data through the compression window, as shown in Figure 6. The main steps are as follows.

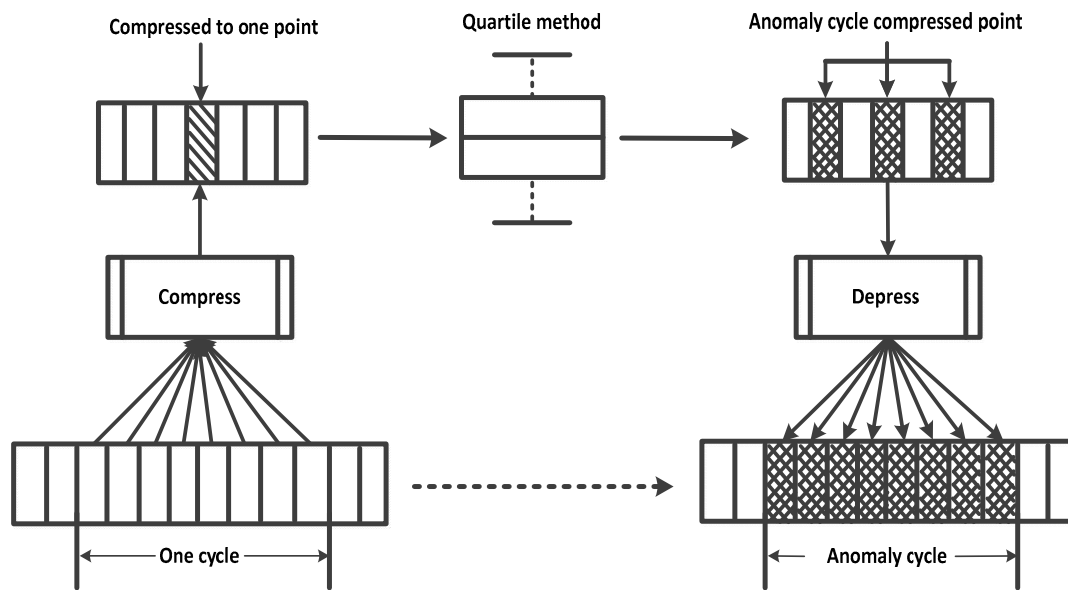


Figure 6. Demonstration of the compressed window method.

Step 1: Choose a suitable compression window size (in this paper, we choose a cycle length as the window size).

Step 2: Find a point in the compression window with the shortest sum of Euclidean distances to all other points and use this point as the compression point.

$$\text{Min}E = \sum_{x \in C_i} \|x - x_k\|^2, x_k \in C_i, k = 1, 2, \dots$$

C_i is the set of data points in the i -th cycle, x_k is the value of the k th data point in the cycle and x is the value of the current target data point.

Step 3: The KPI value of the compressed point is used as the compressed value of the data in the window for dimensionality reduction.

Step 4: Detect and mark the anomalies of the compressed new time series.

Step 5: Decompress the detected data and the decompressed data of each cycle will carry the label value of the parent compression point.

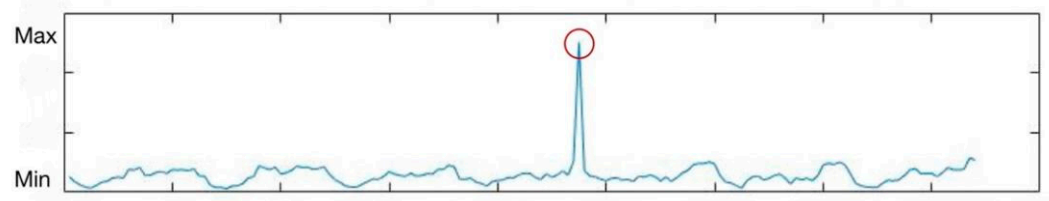
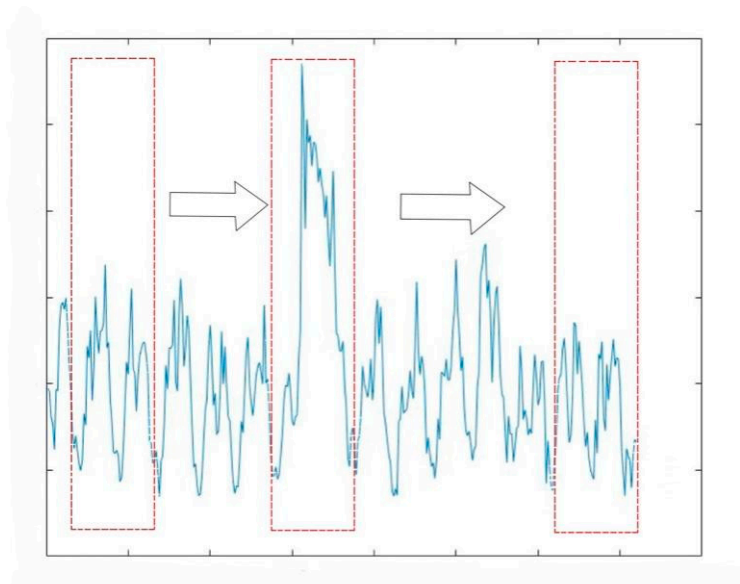
In this Algorithm 1, the rules for selecting the compression points are based on many experiments. In the case of Figure 7, the value of the compressed points will not exceed the threshold of the compressed sequence because the anomaly of the sequence is caused by very few anomalous isolated points, and most of the data points are not biased, and the very few anomalous isolated points can only pull the compressed points towards their own horizontal coordinates in this algorithm, without affecting the size of their indicators (if their indicators are normal). In the case of anomalies, such as Figure 8, the value of the compression points will significantly exceed the threshold of the compression sequence because many points within the compression window are “biased”, causing the value of the compression points to increase. In summary, this paper uses the compression window to label the data points in the anomaly cycle to complete the task of detecting the anomaly cycle.

Algorithm 1: Compress Window Anomaly Period Test (create LNN training set)**Input:** Training KPI dataset A without labels**Output:** label_A is Set A after quartile labeling.

```

1: 0 means outlier, 1 means normal point
2: ceiling < -Upper bound of box graph
3: floor < -bound of box graph
4: for each element of A do
5:   if A[i] < floor || A[i] > ceiling then
6:     label_A[i] = 0
7:   end if
8: end for
9: j < -1
10: for I = 1: N: length(A) do
11:   for each elements a of A [i: i + N - 1] do
12:     compress[j] <- a point whose distance from all elements in the A[i: i + N - 1] is the smallest
13:     j < -j + 1
14:   end for
15: end for
16: for each element of A do
17:   if compress[i] < floor || compress[i] > ceiling then
18:     label_A[(i - 1) × 24 + 1:24 × i] = 0
19:   end if
20: end for

```

**Figure 7.** Situation of very few anomalous isolated points.**Figure 8.** About the abnormal data handling of compressed windows.**5. Labeling Neural Network Structure**

Labeling neural network structure is a multilayer forward type of neural network with error back-propagation learning algorithm, and its structure is shown in Figure 9. In this neural network model, it contains an input layer, an output layer and an intermediate layer in between the input and output layers.

The main idea of labeling neural networks is to divide the learning process into two stages.

The first stage is a forward-propagation process that gives the input information, which is processed layer by layer through the input layer by the implicit layer and calculates the actual output value of each cell.

The second stage is the error back-propagation process, where the difference between the actual output and the desired output (i.e., the error) is recursively calculated layer by layer if the desired output value is not obtained in the output layer to adjust the weights according to this error, so that the error decreases along the gradient direction.

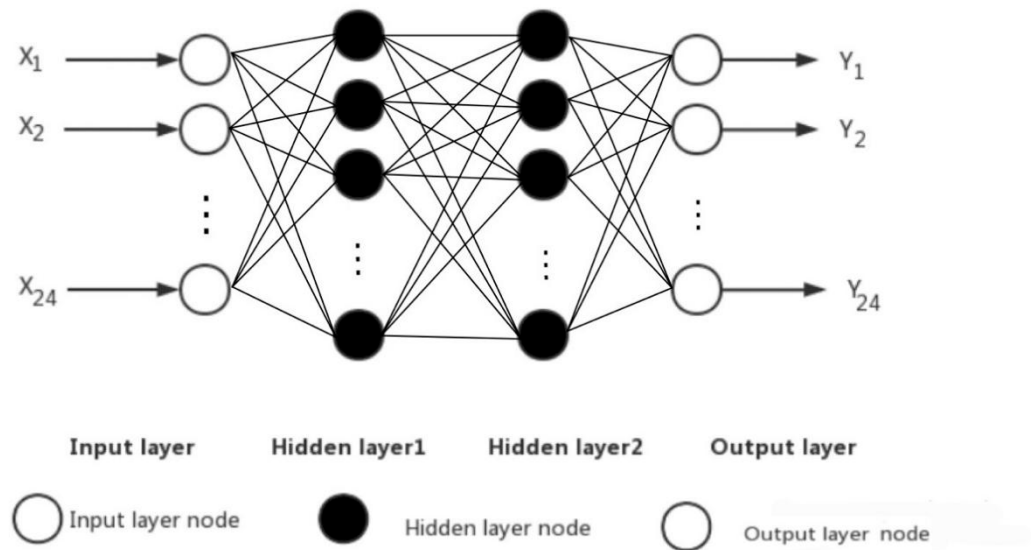


Figure 9. Structural diagram of neural network.

As the process of pattern forward propagation and error back propagation is repeated alternately, the actual output of the network gradually approaches the respective desired output, and the correctness of the network’s response to the input pattern increases. Once the connection weights for each layer question are determined by learning, the neural network is ready to work.

The key to the training process of labeling a neural network is the selection of input and output samples, the initialization of network connection weights and the data of network results.

Preparation of training sample set: Firstly, input samples and output samples are selected, then the data samples must be normalized, and finally, the training set and test set must be set, specifying the number of samples to be trained and the organization of samples to be selected.

Preparation of initial weights: The starting point of the network training travel surface depends on the initialization of the connection weights, so choosing a suitable initialization method is the key to reducing the training time. Usually, the weights are randomly taken as a very small non-zero number.

The design of the neural network structure: mainly the design of the number of hidden layers and the number of connected nodes. Since the labeling network with a single hidden layer can approximate a continuous function in any closed interval, the three-layer network structure can achieve the mapping from X-dimensional space to Y-dimensional space. The number of connected nodes is determined by the number of training samples in the network and the complexity of the implicit law of the samples.

Input layer: the input pattern vector has n inputs $x_i, (i = 1, 2, \dots, n)$;

Hidden layer (middle layer): there are n_1 neurons with input of $x_i, (i = 1, 2, \dots, n)$ and output of $h_j, (j = 1, 2, \dots, n_1)$, weight of hidden layer of w_{ij} and threshold of θ_j ;

$$h_j = f\left(\sum_{i=1}^{n1} w_{ij}h_j - \theta_j\right) \quad j = 1, 2, \dots, n1$$

Output layer: there are m neurons, the output is $y_l, (l = 1, 2, \dots, m)$, the weight of this layer is w_{jl} and the threshold is θ_l .

$$y_l = f \left(\sum_{j=1}^{n1} w_{jl}h_j - \theta_l \right) \quad l = 1, 2, \dots, m$$

Learning sample set:

$$\{(X^p, t^p) \parallel p = 1, \dots, T\}$$

Here: $X^P = (-1, x_1^P, x_2^P, \dots, x_n^P)$ is the input of the P -th sample; $t^P = (-1, t_1^P, t_2^P, \dots, t_m^P)$ is the standard output of X^P ; $y^P = (-1, y_1^P, y_2^P, \dots, y_m^P)$ is the actual output of X^P .

Untrained network, general $t^P - y^P \neq 0$ and the learning algorithm are used to adjust the network weight W so that $(t^P - y^P) \rightarrow 0$. The specific steps of labeling neural network algorithms (Algorithm 2) are as follows.

Algorithm 2: Labeling neural network

Input: training KPI dataset A with labels, params for Labeling neural network

Output: trained model labeling neural network

- 1: set parameters
 - 2: epochs < -1000 training times
 - 3: learning rate < -0.01
 - 4: goal < -0.000001 minimum error of training target
 - 5: **for** element is groped into N **do**
 - 6: count Windows < -A[i:i + N - 1] and labels[i:i + N - 1]
 - 7: **end for**
 - 8: count Windows < -0
 - 9: **for** each count Windows **do**
 - 10: $\omega, \beta < -\text{count Windows}$
 - 11: count Windows < -countWindows + 1
 - 12: **end for**
 - 13: build labeling neural network by ω and β
-

This paper selects KPI data with seasonal distribution. First, 24 data points are taken as the input layer to the bp neural network to learn and then continuously adjust the model according to the anomalies derived from the quartiles to improve the accuracy of the model. However, just bringing in the anomalies detected by the quadrature method does not effectively cover all anomalies, such as anomalous cycles. In this regard, the compression window method mentioned above is used to label all points in the anomaly cycle as anomalies, and then label these anomalies as anomalies and bring them into the LNN neural network for learning to further improve its accuracy.

Labeling neural network has an extremely strong nonlinear mapping capability. When there is a certain mapping relationship between the input information and the output information, the labeling neural network can preserve this relationship without knowing the corresponding equation about this relationship in advance. By building a training network with sufficient input samples, a spatial nonlinear mapping from the x dimension to the Y dimension can be achieved. It can be found that the learned labeling neural network not only accurately learns the anomalies judged by the quadrature method, but also identifies the outliers not judged by the quadrature through its learning ability for potential rules, which indicates that the labeling neural network can effectively solve the novel anomalies appearing in the network environment.

6. The RLSTM Model

The multi-neural network model is a fusion of supervised and unsupervised learning algorithms. The keys of this model are labeling neural network and LSTM-based RLSTM neural network. Therefore, the detailed structure of the labeling neural network will be introduced, LSTM neural network, and the LNN-RLSTM model outlined.

Long short-term memory (LSTM) is a special form of recurrent neural network first proposed by Hoch Reiter [4] in 1997. In the gradient back-propagation phase, traditional recurrent neural networks suffer from gradient disappearance and gradient explosion

when the gradient signal is multiplied by many times. LSTM neural networks overcome the long-term dependence and solve the problem of too small an impact factor due to the large time span by cell variables. These features of LSTM neural networks are just suitable for user-aware classes that possess time-series properties and the anomaly detection of KPI data in the network.

To simulate the RLSTM, a graph of the operational structure of a single neuron is constructed in Figure 10. $C^{(t-1)}$ is the input to the memory unit, $C^{(t)}$ is the memory unit state, $\tilde{C}^{(t)}$ is the updated memory unit state, $O^{(t)}$ is the value of the output gate at moment t , W_f, W_c, W_u, W_r, W_o is the weight matrix and b_f, b_c, b_u, b_r, b_o is the deviation vector.

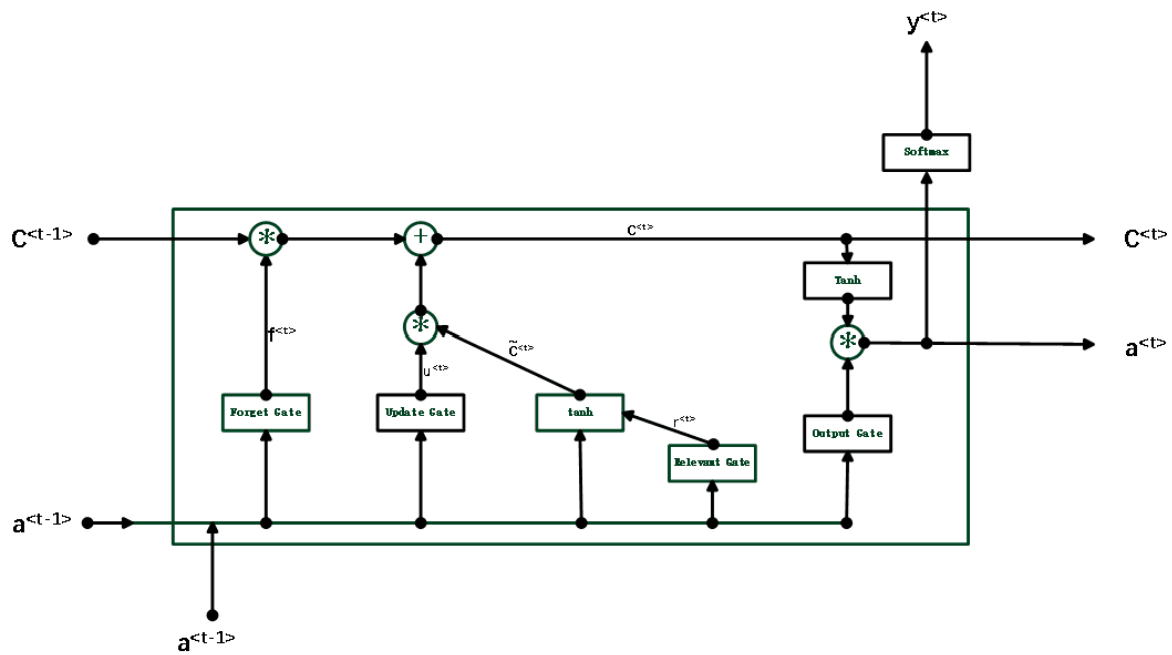


Figure 10. Structural diagram of RLSTM.

The key to LSTM networks is to add or remove information using memory unit states, which are regulated by structures called “update gates” and “forget gates”. In fact, the traditional LSTM neural network is improved by adding a “correlation gate”, which adds the correlation of the before and after time series as a weighting factor in the calculation of $\tilde{C}^{(t)}$. The gates act like filters, allowing optional information to pass through. They consist of a sigmoid neural layer with output values between 0 and 1. The formulas inside these gates are not indicated in Figure 1, but they will be given in detail later in the procedure. The point-state multiplication operations \otimes and \ominus are used to denote the algebraic operations referred to in the RLSTM internals. From Figure 10, an RLSTM has four gates to control the information, which are: update gate, forget gate, correlation gate and output gate. Through the RLSTM (Algorithm 3) neural network, the time-series information will be transmitted, filtered, combined and finally output a whole message to the next memory storage unit. The detailed steps of transmission are shown below.

Step 1: Calculate update gate to store new useful information

$$U_t = \sigma(W_u [a^{(t-1)}, x^{(t)}] + b_u)$$

Step 2: Calculate forgetting gate to forget the redundant old information

$$f_t = \sigma(W_f [a^{(t-1)}, x^{(t)}] + b_f)$$

Step 3: Calculate the correlation gate to determine the correlation before and after the time series

$$r_t = \sigma\left(W_r \left[a^{(t-1)}, x^{(t)} \right] + b_r\right)$$

Step 4: Calculate the state of the memory unit to be updated

$$\tilde{C}^{(t)} = \tanh\left(W_c \left[r_t^* a^{(t-1)}, x^{(t)} \right] + b_c\right)$$

Step 5: Calculate and update the state of the unit

$$C^{(t)} = U_t^* \tilde{C}^{(t)} + f_t^* C^{(t-1)}$$

Step 6: Compute output gate

$$O_t = \sigma\left(W_o \left[a^{(t-1)}, X^{(t)} \right] + b_o\right)$$

Step 7: Output the next memory unit state

$$a^{(t)} = O_t^* \tanh C^{(t)}$$

Algorithm 3: RLSTM neural network

Input: training KPI dataset A with labels, params for RLSTM neural network

Output: trained model RLSTM neural network

```

1: for each element of set A do
2:   count Windows < -A[i: i + N - 1] and labels[i: i + N - 1]
3: end for
4: set parameters
5:   Max Epochs < -1000 the maximum number of rounds to train
6:   Gradient Threshold < -1 the positive threshold for the gradient
7:   Initial Learn Rate < -0.005 the initial learning rate of the training
8:   Learn Rate Schedule < -piecewise ways to reduce the overall learning rate
9:   Learn Rate Drop Period < -125 learning rate reduction factor
10:  Learn Rate Drop Factor < -0.2 the number of epochs
11: count Windows < -0
12: for sliding read data by N do
13:  ω < -count Windows
14:  errors < -∑labels
15:  count Windows < -countWindows + 1
16: end for
17: build LSTM neural network by minimum errors

```

7. Combination of LNN and RLSTM Based on Pre-Labeled Data

In this section, a neural network model with the combination of LNN and RLSTM will be elaborated, which has the ability to identify (1) anomalies, (2) anomalous cycles, (3) anomalous frequencies and other anomalous changes (based on operator experience) that occur in a range of business scenarios.

Figure 11 shows the data-flow diagram for training the anomaly detection neural network in this paper. The network service data mainly come from the terminal layer, which consists of various devices and mainly completes the operation of collecting raw data and reporting them. Taking the mobile communication industry as an example, these data will be captured by the nearby base stations. The base station computing layer realizes the response of basic services through reasonable deployment and deployment of storage capacity. The reported data from the base stations will be permanently stored by the cloud computing center. In the cloud computing center, our model is enabled to be used. Through the pre-processing of the data, the data take on the characteristics of a time series, when the network signal data are not tagged. During the initial input period of the model, to train the LNN, the data are initially labeled by using the quadrature method combined with compressed windows for outlier and anomaly cycle detection, respectively, for the network data. At this point, the data labels contain numerically anomalous outliers and anomalous cycles, and then manual anomalous network data are input (“individual examples” that

the salesperson considers anomalous). The purpose of the preceding step is to solve the problem of missing LNN training data in the case of missing data labels. The labeled data will be substituted into the LNN, a sliding window and BP-based neural network model, for training, which will have the ability to accurately label the data on each given cycle. After training reaches a certain number, the LNN will replace all previous data pre-processing operations and directly label the initial data. The LNN will identify anomalies from the data perspective and business perspective, but due to the existence of the sliding window, the LNN labeling step is “fragmented”, so the labeled data are substituted into the RSTM. The RLSTM is a recurrent neural network model with continuous training, which can solve the “fragmentation” phenomenon of LNN labeling. Through training, RLSTM will inherit the abnormal data recognition ability of LNN and further recognize abnormalities from the perspective of time-series continuity, so a neural network model that can accurately recognize data abnormalities is constructed. The final anomalous data detection results are shown in Figure 12.

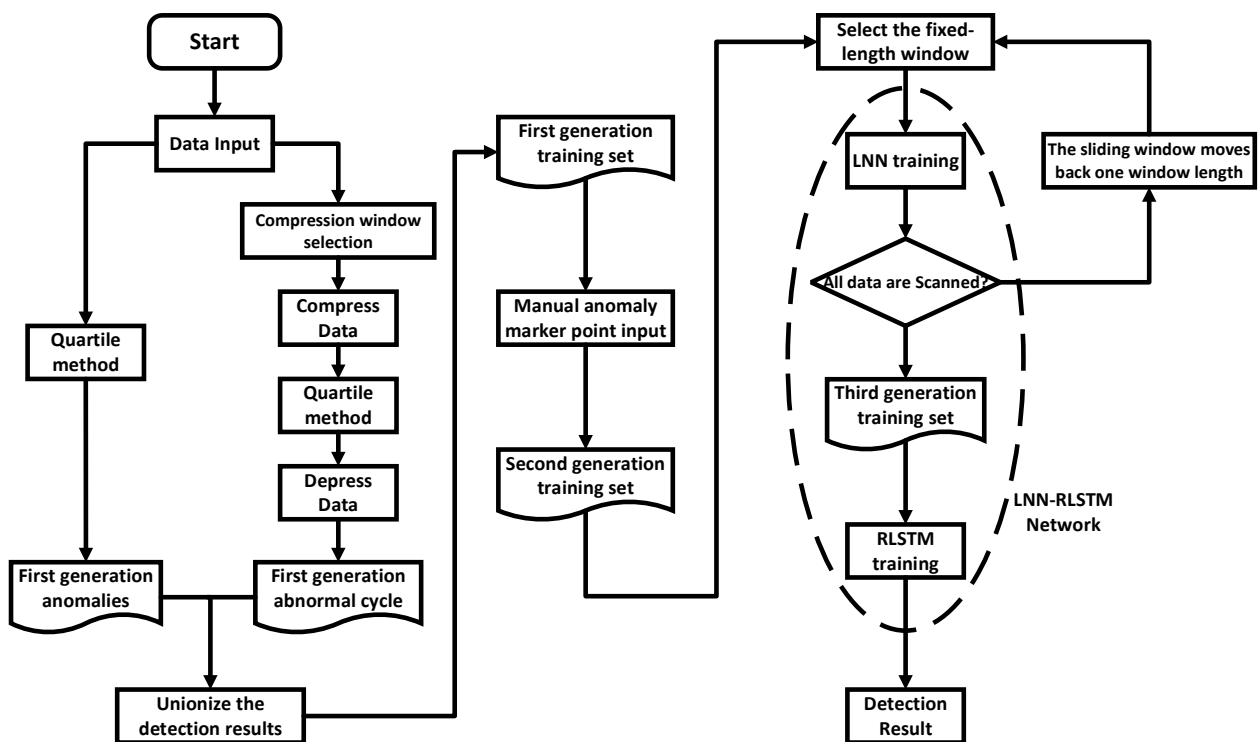


Figure 11. Flow chart of anomaly detection.

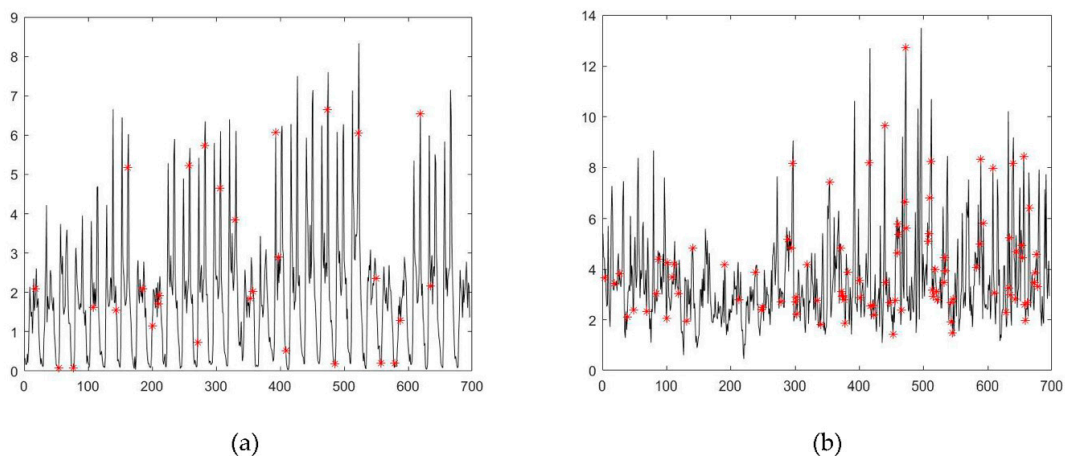
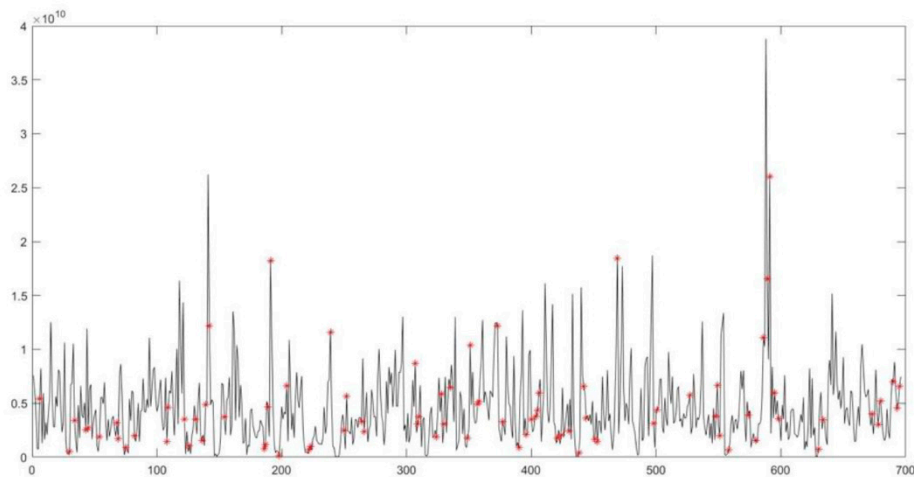


Figure 12. Cont.



(c)

Figure 12. Anomaly detection of three types of KPI data. (a) Anomaly detection of average # of users. (b) Anomaly detection of average # of activated users. (c) Anomaly detection of cell PDCP traffic.

8. Evaluation

8.1. Dataset

The dataset used in this experiment comes from 67 KPI indicators, corresponding to 58 cells covered by five base stations for a total of 29 days from 28 August 2021 at 00:00 to 25 September at 23:00 of the National University Big Data Contest, and they are manually labeled accordingly. We select three KPIs with different characteristics and they have the same 1 h interval between observations. Table 2 shows the detail information of the dataset we used.

Table 2. Statistics of dataset.

KPI	Average # of Users	Average # of Activated Users	Cell PDCP Traffic
Total points	3480	3480	3480
Anomaly point	156/4.48%	100/2.87%	178/4.91%
Total windows	145	145	145
Abnormal windows	2/1.38%	1/0.69%	2/1.38%
Average	16.2898	1.7046	1.7133×10^{10}

8.2. Model Parameter Setting

To compare the objectivity of the results, the hidden layer of the LNN network was set to two layers, 24 units in the first layer and 12 units in the second layer. The size of each sliding window is set to 24 and a training target minimum error of 0.00001 is chosen. The Adam optimizer with an initial learning rate of 0.005 is chosen for RLSTM neural network training. As one of the important components in the model proposed in this paper, the LNN-RLSTM has two important parameters: Initial Learn Rate and Learn Rate Drop Factor. If Initial Learn Rate is too low, the training will take a long time, but if the learning rate is too high, the training may fall into suboptimal results. Learn Rate Drop Factor is applied to the learning rate each time a certain number of epochs pass, reducing the impact of overly old historical data on future data. Experimentally, Initial Learn Rate is chosen to be 0.005 and Learn Rate Drop Factor is chosen to be 0.2, and the model accuracy is high.

8.3. Performance Evaluation

There are different metrics and measures to evaluate an ML model. Each learning task has a focus on various measures [42]. A good anomaly detector must have a high detection rate and a low false-alarm rate. Therefore, in this paper, the relevant measures from the confusion matrix are used to measure the model efficiency. The performance evaluation of anomaly detection algorithms based on time-series data uses recall, precision and accuracy in addition to the best F-Score. The percentage of positive samples predicted by using the model in the overall sample is reflected by the precision, recall reflects the proportion of true-positive samples among those predicted to be positive, F1-score combines precision and recall and accuracy is the number of correctly identified samples in the test sample as a percentage of the total number of samples tested in the database.

- True Positive (TP): if an anomaly is classified by model as an anomaly, result is accepted as TP.
- False Positive (FP): if a normal instance is classified by model as an anomaly, result is accepted as FP.
- True Negative (TN): if an anomaly is classified by model as normal instance, result is accepted as TN.
- False Negative (FN): if a normal instance is classified by model as normal instance, result is accepted as FN.

$$\begin{aligned} \text{precision} &= \frac{TP}{TP+TN}; \\ \text{recall} &= \frac{TP}{TP+FN}; \end{aligned}$$

$$\begin{aligned} \text{F1-score} &= \frac{2TP}{2TP+FP+FN}; \\ \text{accuracy} &= \frac{TP+TN}{TP+TN+FP+FN} \end{aligned}$$

As shown in Figure 13, we compare the performance based on the four metrics under different models with three KPIs. The experimental results show that the LNN-RLSTM model outperforms the other three models. In the three datasets, the anomaly feature of the average number of activated users is more prominent, so the F-Score is higher. The anomaly frequency of the average number of users in the cell is larger than that of the anomaly data, which can explain the phenomenon of “anomaly aggregation” in the cell PDCP data (4.48% of the anomalies and 1.38% of the anomaly windows), but its F-Score is the smallest.

In fact, due to the complexity and diversity of data anomalies, it is still challenging to detect the anomaly frequency with very high accuracy unless the anomalies of data features are extremely obvious (obvious peaks or troughs in values).

8.4. Impact of LNN-RLSTM Technology

Compared with other models, LNN-RLSTM is improved by three points:

- (1) Numerical visualization of anomalous data using quartiles.
- (2) Identification of anomalous data and anomalous cycles using LNN neural network training data.
- (3) Anomalous frequencies can be identified using RLSTM neural network training data.

We used these three techniques to optimize our model and experiments. Adding these three improvements step by step gives us the LNN-RLSTM model.

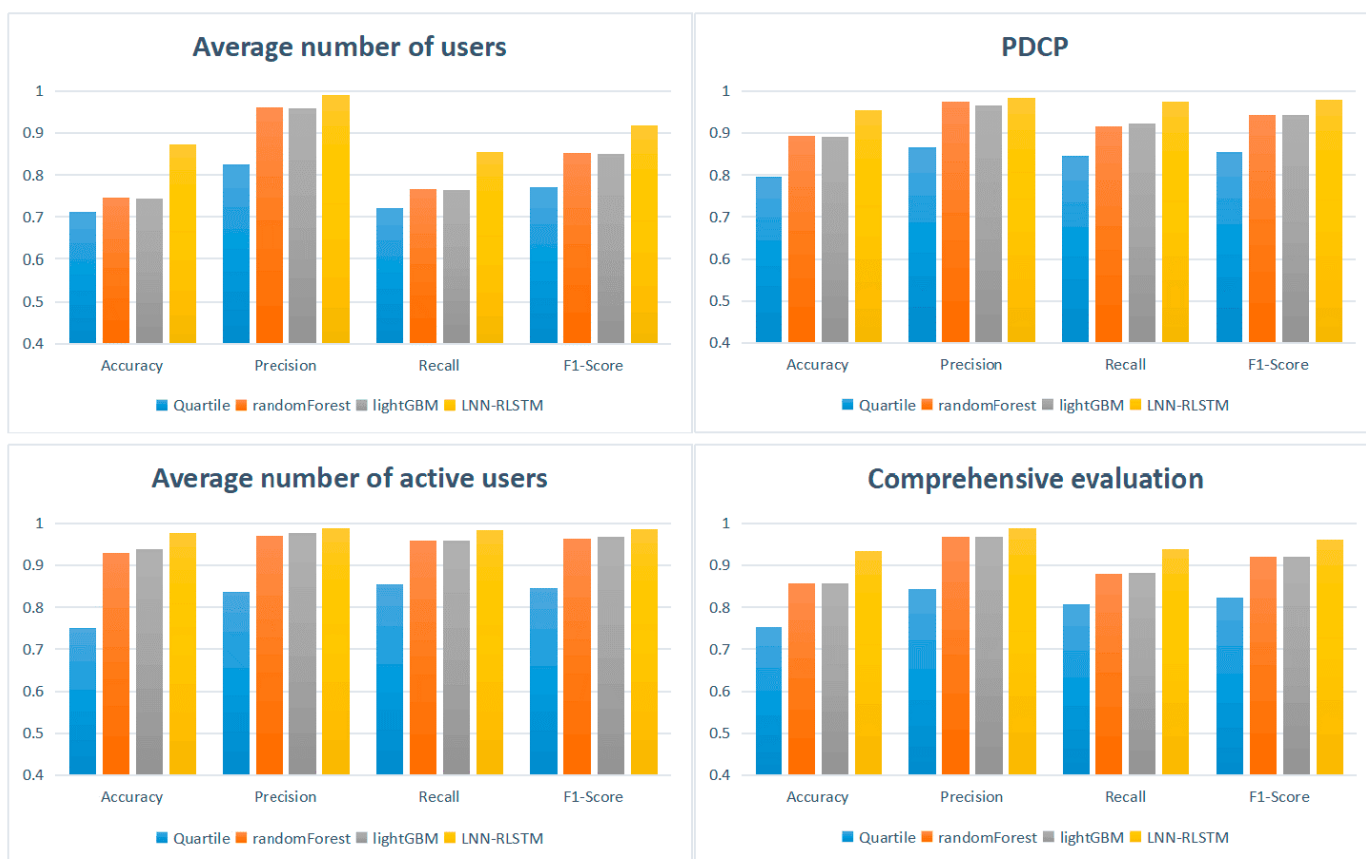


Figure 13. Performance based on four metrics under different models with three KPIs.

8.5. LNN-RLSTM Labels and Real Labels

The quartile method is suitable for excluding only the most prominent feature anomalies and this task is very meaningful for LNN-RLSTM, which has good noise immunity by training to describe the real data feature distribution. During the training process, although the proportion of anomalous data is small, each anomaly may significantly affect the parameters of the model, thus, improving the sensitivity of the model to anomalies. As shown in Figure 14, the time to train the model with the traditional machine learning algorithm with the addition of novel anomalies at a later stage is illustrated. Although the quartile method has better robustness in anomaly detection, the quartile method is not sensitive to novel anomalies added at a later stage because it is an unsupervised learning algorithm and does not learn from the later dataset. Although the training time of Random Forest and LightGBM is shorter than LNN-RLSTM in the model pre-training stage, the model needs to be trained once again as a whole every time a new anomaly type is added manually, thus, not improving in time and even increasing the time complexity. The LNN, on the other hand, can recognize the newly added data anomalies in one cycle length unit and, thus, the LNN training is much faster. In the early stage of model pre-training, the time consumption is relatively high because LNN-RLSTM needs to train a dual network. However, when the model is trained, for variable real-time monitoring environments, further updates to the model can be achieved by simply varying the relevant parameters of the LNN, because the operator’s marking law is also checked in one cycle.

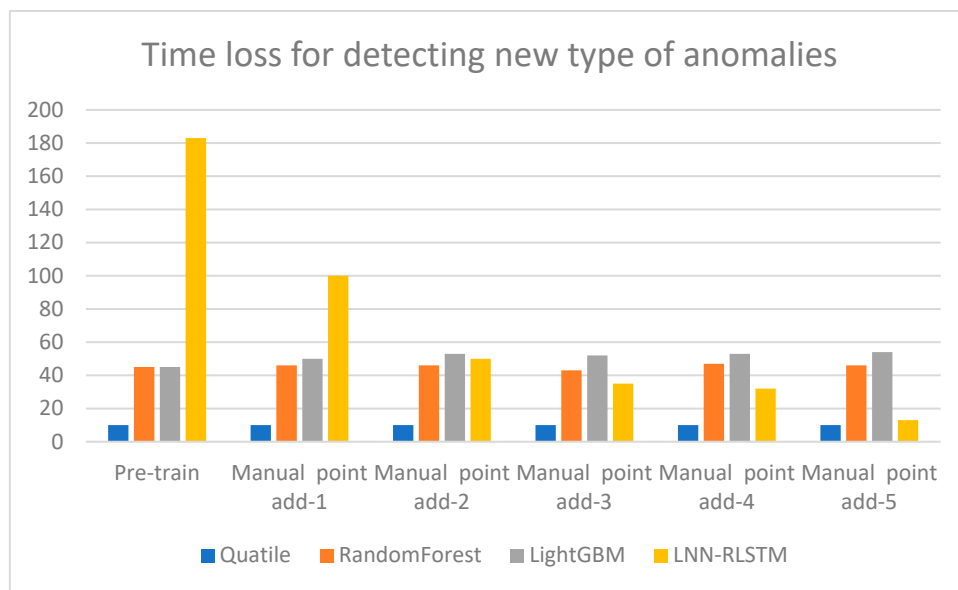


Figure 14. Time loss for training new type of anomalies added manually.

The time complexity of LNN and RLSTM is low, as shown in Table 3. The time complexity of LNN is mainly related to the number of layers in its stack. Since the concept of sliding window is introduced in this paper to help LNNs perform local detection, the number of layers of LNN stacking does not need to be excessive, about 4–7 layers are sufficient. The semi-supervised learning approach of this algorithm is further optimized to improve the accuracy of the algorithm by introducing artificial labeling. The fuzzy weight of the sequence memory in the previous cycle is realized by adding the concept of correlation gate to RLSTM. The parameters in RLSTM can be simplified to two matrices, U and V, which are mapped to input and output, the dimension of U is hidden number * Input and the dimension of V is hidden number * hidden number, so the time complexity is lower than other deep learning algorithms.

Table 3. Time complexity comparison.

Categories	Technique	Year	Domain	Ref	Time Complexity
Supervised	Naive Bayes	1960	Cybersecurity	[42]	$O(mn)^2$
Supervised	Random forest	2001	Cybersecurity	[43]	$O(O(Mm \log n))^3$
Supervised	ANN	2000	Cybersecurity	[44]	$O(emnk)^4$
Supervised	SVM	2011	Email Spam	[45]	$O(n^2)^1$
Unsupervised	DBN	2016	Email Spam	[46]	$O((n + N)k)^7$
Supervised	Decision Tree	2016	Email Spam	[47]	$O(mn^2)^5$
Supervised	SVM	2018	Spam Tweets	[48]	$O(n^2)^1$
Unsupervised	K-means	2007	Software	[49]	$O(I \times n \times k \times m)$
Semi-supervised	LNN	2022	Operational exception	-	$O(n \times m \times k \times v)$
Semi-supervised	RLSTM	2022	Operational exception	-	$O(4(nm + n^2 + n))$

8.6. Real Exceptions for Data Labels

The quartile method is only suitable for testing the most obvious anomalous data, while Random Forest, SVM [42] and LightGBM cannot effectively check the anomalous frequencies [49]. As shown in Table 1, for the average number of users in the cell, the quartile method has the worst detection effect on the dataset and performs better on other datasets, which, to some extent, can also explain why LNN-RLSTM has the smallest F-Score for the average number of users dataset in the previous experiments. Further, we can

see the limitations of the quartile method, so it can only be used as a tool for auxiliary LNN-RLSTM training.

9. Conclusions and Future Work

The LNN-RLSTM model not only has strong anomaly data detection capability, but also constructs different kinds of anomaly detection network models from network operation and maintenance business perspective and operator bias perspective, respectively. Compared with traditional classification methods, such as quadrature method, random forest and lightGBM-genetic algorithm [43–47], its performance is better in terms of Accuracy, Precision, Recall, F1-Score and other indexes. Especially in the case of complex business scenarios, the LNN-RLSTM network architecture can efficiently learn the business knowledge of business operators and can quickly and automatically iterate training to learn new abnormal data types in a timely manner in the face of big data environments. The self-learning habit of this network architecture makes it possible to detect anomalies in the field of anomaly detection, not only at the numerical level but also at the operational level. Of course, in future research, we will still study the factors influencing the formation of data-compression windows to avoid false anomalies from overlearning and further investigate the impact of multi-neural network models in anomaly detection performance.

However, although the proposed architecture in this paper improves in time complexity and business detection capability, it does not consider the impact brought by adversarial attacks [48], and this algorithm model only considers network business data anomalies caused by non-human factors [49], which is not comprehensive for network security detection. In the future, we will focus our research on anomaly detection by adversarial attacks, which is a method where an adversary intentionally adds an adversarial perturbation to the input file to disable the detector. Such anomaly detection is considered necessary in white-box attack prevention [50], where the attacker generates adversarial examples to fool the model based on its architecture, parameters and input features. Our future work will be devoted to the problem of adversarial attacks to improve the generalizability and robustness of the model.

Author Contributions: Conceptualization, X.T. and H.Y.; methodology, X.T.; software, X.T. and S.X.; validation, X.T., S.X. and H.Y.; formal analysis, X.T. and S.X.; investigation, X.T. and S.X.; resources, H.Y.; data curation, S.X.; writing X.T.; writing—review and editing, X.T. and S.X.; visualization, S.X.; supervision, X.T. and H.Y.; project administration, H.Y.; funding acquisition, H.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Natural Science Foundation of China (61773012), the National natural sciences fund youth fund project (61807017) and the research start-up fund of Jiangsu University of science and technology (1052932101).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The dataset for this study is from the 2021 Chinese Big data competition.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yang, Y.-L.; Bi, Z.-Z. Deep learning-based anomaly detection for network traffic. *Comput. Sci.* **2021**, *48*, 540–546.
2. Ren, H.; Xu, B.; Wang, Y.; Yi, C.; Huang, C.; Kou, X.; Xing, T.; Yang, M.; Tong, J.; Zhang, Q. Time-series anomaly detection service at Microsoft. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 3009–3017.
3. He, S.; Li, Z.; Wang, J.; Xiong, N.N. Intelligent detection for key performance indicators in industrial-based cyber-physical systems. *IEEE Trans. Ind. Inform.* **2020**, *17*, 5799–5809. [[CrossRef](#)]
4. Sommer, R.; Paxson, V. Outside the Closed World: On Using Machine Learning for Network Intrusion Detection. In Proceedings of the 2010 IEEE Symposium on Security and Privacy, Berkeley/Oakland, CA, USA, 16–19 May 2010; pp. 305–316.

5. Liu, D.; Zhao, Y.; Xu, H.; Sun, Y.; Pei, D.; Luo, J.; Jing, X.; Feng, M. Opprentice: Towards Practical and Automatic Anomaly Detection Through Machine Learning. In Proceedings of the 2015 Internet Measurement Conference, Tokyo, Japan, 28–30 October 2015; pp. 211–224.
6. Sedjelmaci, H.; Senouci, S.M.; Ansari, N. Intrusion Detection and Ejection Framework Against Lethal Attacks in UAV-Aided Networks: A Bayesian Game-Theoretic Methodology. *IEEE Trans. Intell. Transp. Syst.* **2016**, *18*, 1143–1153. [\[CrossRef\]](#)
7. Heba, F.E.; Darwish, A.; Hassanien, A.E.; Abraham, A. Principle Components Analysis and Support Vector Machine Based Intrusion Detection System. In Proceedings of the 2010 10th International Conference on Intelligent Systems Design and Applications, Cairo, Egypt, 29 November–1 December 2010; pp. 363–367.
8. Shaukat, K.; Luo, S.; Varadharajan, V.; Hameed, I.A.; Chen, S.; Liu, D.; Li, J. Performance comparison and current challenges of using machine learning techniques in cybersecurity. *Energies* **2020**, *13*, 2509. [\[CrossRef\]](#)
9. Jain, A.K.; Mao, J.; Mohiuddin, K.M. Artificial neural networks: A tutorial. *Computer* **1996**, *29*, 31–44. [\[CrossRef\]](#)
10. Ross, Q. *C4. 5: Programs for Machine Learning—San Mateoca*; Morgan Kaufmann: Burlington, MA, USA, 1993.
11. Awad, W.A.; Elseuofi, S.M. Machine learning methods for spam e-mail classification. *Int. J. Comput. Sci. Inf. Technol.* **2011**, *3*, 173–184. [\[CrossRef\]](#)
12. Tyagi, A. *Content Based Spam Classification-A Deep Learning Approach*; University of Calgary: Calgary, AB, Canada, 2016.
13. Khan, Z.; Qamar, U. Text Mining Approach to Detect Spam in Emails. In Proceedings of the International Conference on Innovations in Intelligent Systems and Computing Technologies (ICIISCT2016), Manila, Philippines, 24–26 February 2016; p. 45.
14. Saab, S.A.; Mitri, N.; Awad, M. Ham or Spam? A Comparative Study for Some Content-Based Classification Algorithms for Email Filtering. In Proceedings of the MELECON 2014–2014 17th IEEE Mediterranean Electrotechnical Conference, Beirut, Lebanon, 13–16 April 2014; pp. 339–343.
15. Jain, G.; Agarwal, B. Spam detection on social media using semantic convolutional neural network. *Int. J. Knowl. Discov. Bioinform.* **2018**, *8*, 12–26. [\[CrossRef\]](#)
16. Liu, F.T.; Ting, K.M.; Zhou, Z.H. Isolation Forest. In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 15–19 December 2008; pp. 413–422.
17. Wang, C.; Zhao, Z.; Gong, L.; Zhu, L.; Liu, Z.; Cheng, X. A distributed anomaly detection system for in-vehicle network using HTM. *IEEE Access* **2018**, *6*, 9091–9098. [\[CrossRef\]](#)
18. Yuan, Y.; Adhatarao, S.S.; Lin, M.; Liu, Z.; Fu, X. Ada: Adaptive deep log anomaly detector. In Proceedings of the IEEE INFOCOM 2020—IEEE Conference on Computer Communications, Toronto, ON, Canada, 6–9 July 2020; pp. 2449–2458.
19. Yoon, K.A.; Kwon, O.S.; Bae, D.H. An Approach to Outlier Detection of Software Measurement Data Using the k-Means Clustering Method. In Proceedings of the First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007), Madrid, Spain, 20–21 September 2007; pp. 443–445.
20. Stein, D.W.; Beaven, S.G.; Hoff, L.E.; Winter, E.M.; Schaum, A.P.; Stocker, D.A. Anomaly detection from hyperspectral imagery. *IEEE Signal Processing Mag.* **2002**, *19*, 58–69. [\[CrossRef\]](#)
21. Shaukat, K.; Luo, S.; Varadharajan, V.; Hameed, I.A.; Xu, M. A survey on machine learning techniques for cyber security in the last decade. *IEEE Access* **2020**, *8*, 222310–222354. [\[CrossRef\]](#)
22. Shaukat, K.; Luo, S.; Varadharajan, V. A novel method for improving the robustness of deep learning-based malware detectors against adversarial attacks. *Eng. Appl. Artif. Intell.* **2022**, *116*, 105461. [\[CrossRef\]](#)
23. Fu, X.; Fortino, G.; Pace, P.; Aloï, G.; Li, W. Environment-fusion multipath routing protocol for wireless sensor networks. *Inf. Fusion* **2020**, *53*, 4–19. [\[CrossRef\]](#)
24. Naseer, S.; Saleem, Y.; Khalid, S.; Bashir, M.K.; Han, J.; Iqbal, M.M.; Han, K. Enhanced network anomaly detection based on deep neural networks. *IEEE Access* **2018**, *6*, 48231–48246. [\[CrossRef\]](#)
25. Persian, C.B. Time-Invariant 3D Human Action Recognition with Positive and Negative Movement Memory and Convolutional Neural Network Three-stream Very Deep Neural Network for Video Action Recognition Unsupervised Hyperspectral Target Detection Using Spectral Residual of Deep Autoencoder Networks Towards Information Theoretic Measurement of Fidelity and Diversity in Handwriting Synthesis. In Proceedings of the 2019 4th International Conference on Pattern Recognition and Image Analysis (IPRIA), Tehran, Iran, 6–7 March 2019.
26. Shaukat, K.; Alam, T.M.; Luo, S.; Shabbir, S.; Hameed, I.A.; Li, J.; Abbas, S.K.; Javed, U. A review of Time-Series Anomaly Detection Techniques: A Step to Future Perspectives. In *Future of Information and Communication Conference*; Springer: Cham, Switzerland, 2021; pp. 865–877.
27. Ahmed, U.; Srivastava, G.; Djenouri, Y.; Chun-Wei Lin, J. Deviation Point Curriculum Learning for Trajectory Outlier Detection in Cooperative Intelligent Transport Systems. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 16514–16523. [\[CrossRef\]](#)
28. Tao, X.; Peng, Y.; Zhao, F.; Wang, S.; Liu, Z. An Improved Parallel Network Traffic Anomaly Detection Method Based on Bagging and GRU. In *Proceedings of the International Conference on Wireless Algorithms, Systems, and Applications, Qingdao, China, 13–15 September 2020*; Springer: Cham, Switzerland, 2020; pp. 420–431.
29. Ali, M.S.; Tabassum, H.; Hossain, E. Dynamic user clustering and power allocation for uplink and downlink non-orthogonal multiple access (NOMA) systems. *IEEE Access* **2016**, *4*, 6325–6343. [\[CrossRef\]](#)
30. Fujimaki, R.; Yairi, T.; Machida, K. An Approach to Spacecraft Anomaly Detection Problem Using Kernel Feature Space. In Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, Anchorage, AK, USA, 4–8 August 2005; pp. 401–410.

31. Toshniwal, A.; Mahesh, K.; Ayashree, R. Overview of Anomaly Detection Techniques in Machine Learning. In Proceedings of the 2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 7–9 October 2020; pp. 808–815.
32. Kamran, S.; Alam, T.M.; Hameed, I.A.; Khan, W.A.; Abbas, N.; Luo, S. A Review on Security Challenges in Internet of Things (IoT). In Proceedings of the 2021 26th International Conference on Automation and Computing (ICAC), Portsmouth, UK, 2–4 September 2021; pp. 1–6.
33. Kacprzyk, W.; Owsinski, J.W.; Viattchenin, D.A.; Shyrai, S. A New Heuristic Algorithm of Possibilistic Clustering Based on Intuitionistic Fuzzy Relations. In *Novel Developments in Uncertainty Representation and Processing*; Springer: Cham, Switzerland, 2016; pp. 199–214.
34. Heryadi, Y. The Effect of Several Kernel Functions to Financial Transaction Anomaly Detection Performance using One-Class SVM. In Proceedings of the 2019 International Congress on Applied Information Technology (AIT), Yogyakarta, Indonesia, 4–6 November 2019; pp. 1–7.
35. Burges, C. A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Discov.* **1998**, *2*, 121–167. [[CrossRef](#)]
36. Frank, E.; Hall, M.A. *Data Mining: Practical Machine Learning Tools and Techniques*; Morgan Kaufmann: Burlington, MA, USA, 2011.
37. Umair, J.; Shaukat, K.; Hameed, I.A.; Iqbal, F.; Alam, T.M.; Luo, S. A review of content-based and context-based recommendation systems. *Int. J. Emerg. Technol. Learn.* **2021**, *16*, 274–306.
38. Zhou, X.; Hu, Y.; Liang, W.; Ma, J.; Jin, Q. Variational LSTM enhanced anomaly detection for industrial big data. *IEEE Trans. Ind. Inform.* **2020**, *17*, 3469–3477. [[CrossRef](#)]
39. Junir, M.; Siddiqui, S.A.; Dengel, A.; Ahmed, S. DeepAnT: A deep learning approach for unsupervised anomaly detection in time series. *IEEE Access* **2018**, *7*, 1991–2005.
40. He, H.; Garcia, E.A. Learning from imbalanced data. *IEEE Trans. Knowl. Data Eng.* **2009**, *21*, 1263–1284.
41. Shaukat, K.; Rubab, A.; Shehzadi, I.; Iqbal, R. A socio-technological analysis of cyber crime and cyber security in Pakistan. *Transylv. Rev.* **2017**, *1*, 84.
42. Hassan, M.U.; Shahzaib, M.; Shaukat, K.; Hussain, S.N.; Mubashir, M.; Karim, S.; Shabir, M.A. DEAR-2: An Energy-Aware Routing Protocol with Guaranteed Delivery in Wireless Ad-Hoc Networks. In *Recent Trends and Advances in Wireless and IoT-Enabled Networks*; Springer: Cham, Switzerland, 2019; pp. 215–224.
43. Zhao, N.; Zhu, J.; Liu, R.; Liu, D.; Zhang, M.; Pei, D. Label-Less: A Semi-Automatic Labelling Tool for kpi Anomalies. In Proceedings of the IEEE INFOCOM 2019-IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; pp. 1882–1890.
44. Nassif, A.B.; Talib, M.A.; Nasir, Q.; Mohamad Dakalbab, F. Machine learning for anomaly detection: A systematic review. *IEEE Access* **2021**, *9*, 78658–78700. [[CrossRef](#)]
45. Agrawal, R.; Srikant, R. Mining Sequential Patterns. In Proceedings of the Eleventh International Conference on Data Engineering, Taipei, Taiwan, 6–10 March 1995; pp. 3–14.
46. Qiu, J.; Du, Q.; Qian, C. Kpi-tsad: A time-series anomaly detector for kpi monitoring in cloud applications. *Symmetry* **2019**, *11*, 1350. [[CrossRef](#)]
47. Zhang, S.; Zhao, C.; Sui, Y.; Su, Y.; Sun, Y.; Zhang, Y.; Pei, D.; Wang, Y. Robust KPI Anomaly Detection for Large-Scale Software Services with Partial Labels. In Proceedings of the 2021 IEEE 32nd International Symposium on Software Reliability Engineering (ISSRE), Wuhan, China, 25–28 October 2021; pp. 103–114.
48. Yan, S.; Tang, B.; Luo, J.; Fu, X.; Zhang, X. Unsupervised Anomaly Detection with Variational Auto-Encoder and Local Outliers Factor for KPIs. In Proceedings of the 2021 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom), New York City, NY, USA, 30 September–3 October 2021; pp. 476–483.
49. Zhang, Y.; Meratnia, N.; Havinga, P. Outlier detection techniques for wireless sensor networks: A survey. *IEEE Commun. Surv. Tutor.* **2010**, *12*, 159–170. [[CrossRef](#)]
50. Alipour, H.; Al-Nashif, Y.B.; Satam, P.; Hariri, S. Wireless anomaly detection based on IEEE 802.11 behavior analysis. *IEEE Trans. Inf. Forensics Secur.* **2015**, *10*, 2158–2170. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.