future internet

MDPI

Article

Deep Anomaly Detection Based on Variational Deviation Network

Junwen Lu *, Jinhui Wang *, Xiaojun Wei, Keshou Wu and Guanfeng Liu

School of Computer and Information and Engineering, Xiamen University of Technology, Xiamen 361024, China; xjwei@xmut.edu.cn (X.W.); kswu@xmut.edu.cn (K.W.); guanfeng.liu@mq.edu.au (G.L.)
* Correspondence: jwlu@xmut.edu.cn (J.L.); 1922031151@stu.xmut.edu.cn (J.W.)

Abstract: There is relatively little research on deep learning for anomaly detection within the field of deep learning. Existing deep anomaly detection methods focus on the learning of feature reconstruction, but such methods mainly learn new feature representations, and the new features do not fully reflect the original features, leading to inaccurate anomaly scores; in addition, there is an end-to-end deep anomaly detection algorithm, but the method cannot accurately obtain a reference score that matches the data themselves. In addition, in most practical scenarios, the data are unlabeled, and there exist some datasets with labels, but the confidence and accuracy of the labels are very low, resulting in inaccurate results when put into the model, which makes them often designed for unsupervised learning, and thus in such algorithms, the prior knowledge of known anomalous data is often not used to optimize the anomaly scores. To address the two problems raised above, this paper proposes a new anomaly detection model that learns anomaly scores mainly through a variational deviation network (i.e., not by learning the reconstruction error of new features, distance metrics, or random generation, but by learning the normal distribution of normal data). In this model, we force the anomaly scores to deviate significantly from the normal data by a small amount of anomalous data and a reference score generated by variational self-encoding. The experimental results in multiple classes of data show that the new variational deviation network proposed in this paper has higher accuracy among the mainstream anomaly detection algorithms.

Keywords: anomaly detection; variational auto-encoder; deep learning; semi-supervised learning

Citation: Lu, J.; Wang, J.; Wei, X.; Wu, K.; Liu, G. Deep Anomaly Detection Based on Variational Deviation Network. *Future Internet* **2022**, *14*, 80. https://doi.org/10.3390/fi14030080

Academic Editor: Daniel Gutiérrez Reina

Received: 17 January 2022
Accepted: 24 February 2022
Published: 8 March 2022

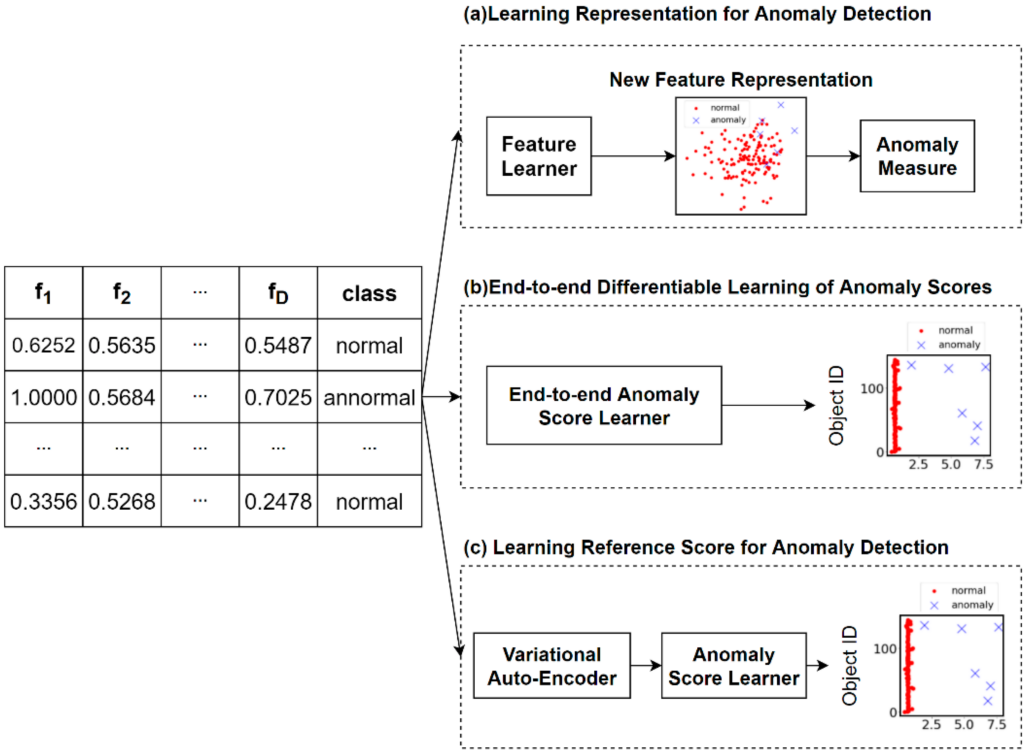
1. Introduction

Anomaly detection refers to the identification of "abnormal points" in data through data mining methods. Common application scenarios include the financial field: identifying "fraud cases" from financial data such as identifying credit card application fraud, false credit, etc.; network security: identify "intruders" from traffic data and identify new network intrusion patterns [1,2]; e-commerce field: identify "malicious buyers" from transaction data such as fleece parties and malicious screen-sweeping groups; early warning of ecological disasters: based on the forecast of wind speed, rainfall, temperature, and other indicators, judge the extreme weather that may occur in the future; industry: the defect detection of industrial products can be carried out by abnormal detection methods, instead of human eyes for measurement and judgment [3,4]. Many anomaly detection algorithms have been introduced for these fields, but the high latitude and non-linearity of the data features make it impossible to effectively detect abnormal data in this type of data [5,6].

In recent years, deep learning [7] has been able to learn and train complex relationships in high-latitude, non-linear data, but there are still two main problems in deep learning in anomaly detection. (1) Unbalanced categories, where anomalies are usually rare instances of data, while normal instances usually account for the vast majority of data. Therefore, it is difficult or even impossible to collect a large number of labeled abnormal instances. This makes it impossible to obtain large-scale markup data in most applications. (2) Heterogeneity of abnormal types: Abnormalities are irregular, and one type of abnormality may

show completely different abnormal characteristics from another type of abnormality. For example, in video surveillance, abnormal events such as robbery, traffic accidents, and theft are visually very different; this poses a huge challenge to the breadth of anomaly detection.

In response to the above two challenges, the existing deep learning-based anomaly detection algorithms [5,6,8–12] use unsupervised learning to complete the above challenges in two steps (i.e., the pipeline (a) in Figure 1). (1) Reconstruct features: learn new feature representations for all normal data such as the intermediate representation of the autoencoder [8,9,12]. The use of autoencoders for anomaly detection is based on the fact that normal instances can be better derived from the compressed feature space than abnormal instances. Refactor this assumption. In order to minimize the overall reconstruction error, the retained information must be as relevant as possible to the input instance (such as the normal instance). Generate the latent space in the confrontation network [10,11]. The distance-based method [5,6] is to calculate the distance between each point and the surrounding points to determine whether a point is abnormal. The method assumes that there are many neighboring points around the normal point, and the abnormal point is away from the surrounding points and the distances are relatively far. (2) Through the reconstruction error [8–12] or distance metric [5,6] obtained by learning in the first step, the abnormal anomaly score is expressed. However, the reconstruction error in this two-step method [8–12] may not be able to fully represent the original features, resulting in a decrease in the accuracy of the final anomaly detection result. Compared with the distance metric [5,6], the traditional anomaly detection method is integrated into the reconstructed feature to solve this problem. Although this method greatly improves the ability of the reconstructed feature to restore the original feature, its optimization point is mainly the feature learning, indicating that this may lead to low learning efficiency and the low quality of abnormal scores.



f_1	f_2	···	f_D	class
0.6252	0.5635	···	0.5487	normal
1.0000	0.5684	···	0.7025	annormal
···	···	···	···	···
0.3356	0.5268	···	0.2478	normal

Figure 1. (**a**) Learning features for subsequent anomaly measures vs. (**b**) direct learning of anomaly scores vs. (**c**) learning reference score for subsequent anomaly measures.

For the above-mentioned main class algorithms, it is mainly unsupervised learning [13], and a common problem in unsupervised learning is that it is difficult to control what features the model learns because unsupervised learning lacks prior knowledge to

guide optimization features. The solution to this problem is to have supervised learning [14] and semi-supervised learning [15], but in the field of anomaly detection, there are no large-scale labeled data, so here is mainly the description of semi-supervised learning, that is, the use of limited abnormal data to learn the abnormal score model (these types of data are often the representative abnormal data that was manually screened in the early stage, for example, in the field of financial fraud, the deceived customer reports the bank, and the bank puts an abnormal label on the customer's data), semi-supervised learning avoids the waste of data and resources, and at the same time, solves the problems of insufficient generalization ability of supervised learning models and inaccurate models of unsupervised learning.

In recent years, there is a type of end-to-end learning model of deviation networks in semi-supervised learning (i.e., the pipeline (b) in Figure 1) [16]. The method takes raw data as input and learns and outputs anomaly scores directly instead of feature representations. Specifically, given a training data object, the proposed framework first uses a neural anomaly score learner to assign anomaly scores to it, and then defines the average of the anomaly scores of some normal data objects based on prior probabilities as a reference score to guide subsequent anomaly score learning. Finally, the framework defines a loss function, called bias loss, to enforce statistically significant deviations of the anomaly scores from the normal data object scores in the upper tail. Although this type of method solves the above two main problems, the reference score of the normal data is obtained through the prior probability. Scores cannot explain normal data well.

In this article, in response to the shortcomings of the existing methods, we introduce a new anomaly detection model: variational auto-encoder (i.e., the pipeline (c) in Figure 1), which uses a data-driven method to perfectly learn the reference scores of each normal data, making it a more convincing force. Thus, a new anomaly detection framework was designed. First, we learned the normal distribution of normal data through variational self-encoding as a reference score, and then used another neural network to learn the anomaly scores. Finally, the framework defines a deviation loss function. The mandatory anomaly score has a significant deviation from the reference score of the normal data.

The network model is further instantiated as a method called variational deviation networks (V-DevNet). V-DevNet is a semi-supervised learning method that uses a small amount of abnormal data, which usually varies from a few to dozens of abnormal data, which only accounts for 0.006–1% of all data. These abnormal data are trained through neural networks, where the normal distribution generated by the variational auto-encoder is used as the reference score, and finally, the abnormal score is directly optimized through the deviation loss. The variational deviation network designed in this way not only makes the reference score more consistent with different data types, making it more convincing, but also has more extensiveness for multiple types of abnormal data. In addition, unlike the anomaly scores generated by other mainstream algorithms, it is more difficult to interpret, and the anomaly scores obtained through the variational deviation networks of the deviation loss are easier to interpret.

Compared with the (a) class method in Figure 1, our model does not directly learn the feature representation of the data, but continuously learns the normal distribution of each data through variational self-coding, which greatly reduces the problem of inaccurate detection caused by reconstruction error; compared with the (b) class method in Figure 1, our model learns the reference scores through variational self-encoding, so that it can learn the reference scores that match the data itself for different data types, which makes its detection accuracy higher, instead of directly obtaining the reference scores randomly in the (b) class method.

Therefore, this article has made the following contributions:

1. Introduced a new model: Variational auto-encoder. This model realizes the specification of reference scores by learning the normal distribution of each normal data and generates different reference scores for different data, making the reference scores more meaningful strong explanatory power; and

2. The framework instantiates a new deep anomaly detection method, namely the variational deviation networks (V-DevNet). V-DevNet optimizes the anomaly score by anomaly score neural network, variational self-encoding, and deviation loss, and the obtained anomaly score is optimized accurately and easily explained.

Experiments on nine types of existing high-dimensional data show that:

In the AUC-ROC and AUC-PR curves, the experimental results of V-DevNet were significantly better than the current mainstream four types of abnormal detection methods. Among them, AUC-ROC was improved by 2–33% on average, and AUC-PR was improved by 6–332% on average.

2. Related Work

2.1. Traditional Anomaly Detection

In traditional anomaly detection algorithms, there are mainly distance-based and density-based unsupervised learning. The distance-based method is not suitable for high-dimensional data, and can only find abnormal points, not abnormal clusters, and calculate the nearest neighbors every time distance must traverse the entire dataset, which is not suitable for online applications of big data; the same density-based methods are also not suitable for high-dimensional data and online applications; in recent years, many integration methods have appeared [17], etc. [18,19] Although these methods have made great progress in selecting features from high-dimensional data, how to accurately extract the complex relationships and important features of high-level data is still a challenge.

The limitation of traditional anomaly detection work is that it is impossible to accurately train for high latitude data to obtain feature representations that match the anomaly data, thus greatly reducing the accuracy of anomaly detection.

2.2. Deep Anomaly Detection

At present, after the extensive application of deep learning, people have also begun to study the application of deep learning to various abnormal tasks, and have achieved great success such as autoencoders, which train encoders and decoders on normal data. During anomaly detection, if the vector from the decoder is very different from the original vector, it is considered abnormal data. There is also a GAN network, which is mainly composed of a generation network and a discrimination network. It is also a learning reconstruction error. If it is very different from the original vector, it is judged as abnormal data. This type of method solves, to a certain extent, the challenge that traditional methods (random projection [20]) cannot learn high latitude data at high latitudes. This type of algorithm can learn the complex relationships between each feature at high latitudes as well as important features. However, the accuracy of subsequent individual anomaly detection work is not good [5,6], so in recent research, in order to solve this problem, the REPEN [5] algorithm (algorithm for learning representations of ultrahigh-dimensional data for random distance-based outlier) and the Deep-SVDD [6] algorithm (algorithm for deep support vector data description) and DevNet [16] algorithm (algorithm for deep anomaly detection with deviation networks) have been proposed successively. The REPEN algorithm combines feature learning representation with distance-based methods, while Deep-SVDD mainly learns the single-class feature representation of SVDD (support vector data description) [21]; this type of algorithm mainly learns the feature representation of high-dimensional data where sometimes it cannot fully represent the characteristics of high-dimensional data and is unexplainable. The DevNet algorithm implements an end-to-end abnormal scoring network, eliminating the need for intermediate feature learning, but the reference score of normal data in this method is randomly generated for a special dataset, so a valid reference score cannot be obtained, which affects the accuracy. The variational deviation network proposed in this paper mainly learns the normal distribution of each normal data, and calculates the normal distribution of the overall normal data. The deviation loss function is then used to guide the optimization of the abnormal score network.

Although the current mainstream deep anomaly detection solves the feature training under high-latitude data, their limitations lie in their uninterpretability and the fact that high-latitude feature learning ignores its detailed features to reduce accuracy; although the DevNet algorithm solves the uninterpretability of the model to a certain extent, its limitation lies in the fact that the parameters are randomly generated and cannot fit the reference scores of the normal data of the data itself.

2.3. Limited Abnormal Data

In the above-mentioned deep anomaly detection algorithm, it is mainly unsupervised learning, so the limitation is that they do not utilize too few anomaly data as a guide to optimize the whole anomaly score network, while REPEN [5] uses a small amount of abnormal labeled data to further learn anomaly features, which is consistent compared to unsupervised learning [22], where its accuracy is improved by 30%; DevNet also uses a small number of abnormal data to directly learn abnormal scores, and its accuracy was also significantly improved by 33% [23]. The algorithm learns the correlation between different labels, which provides a theoretical basis for the detection of different abnormal types of limited abnormal data. Therefore, the algorithm proposed in this paper inherits the above advantages [24] and uses a small amount of abnormal data to produce significant deviations from the reference scores of normal data to guide its optimization, thereby further improving the accuracy of detection.

3. Reference Score Learning

3.1. Problem Statement

The goal of this article is how to accurately learn the reference scores of the corresponding data in a data-driven way, instead of using the current random generation method to randomly generate 5000 reference scores through a normal distribution and find their average reference scores as a guide to optimizing the abnormal score network. Specifically, it learns the normal distribution of each normal data in a data-driven manner through variational self-encoding, and then averages the normal distribution of each normal data to guide the optimization of the abnormal score network.

3.2. Variational Auto-Encoder

In order to solve this problem, we introduced a new model—variational auto-encoder [25,26]. Although the model is also trained to train encoders and decoders, the essence of the encoder in this model is used to calculate the mean and variance in the normal distribution, so two encoders are generated for the mean and variance, which is essentially based on the conventional autoencoder. The result of the encoder (corresponding to the network for calculating the mean in the variational auto-encoder) adds "Gaussian noise" to make the resulting decoder robust to noise, and the additional KL loss function in the variational auto-encoder (the purpose is to make the mean value 0 and the variance 1). The above is equivalent to a regular term for the encoder, and it is hoped that everything from the encoder has a zero mean value. The function of the other encoder (corresponding to the network that calculates the variance) is to dynamically adjust the intensity of the noise, as shown in Figure 2.

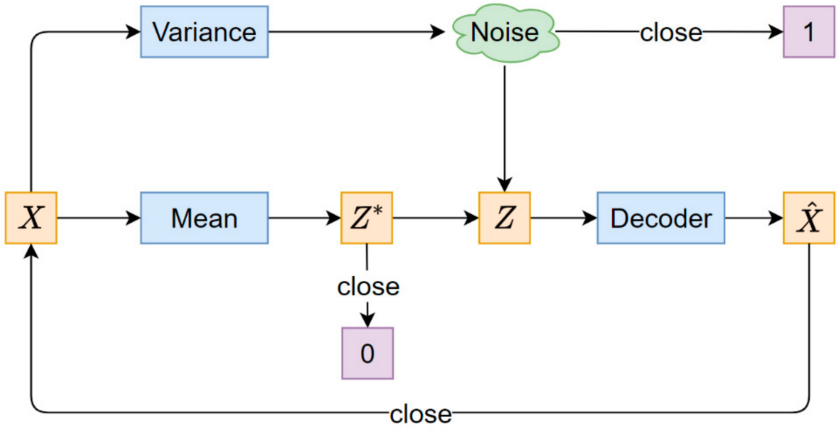


Figure 2. The entire training process of variational auto-encoder. X is trained by two encoders to obtain the mean and variance, where the encoder that obtains the mean converges to 0 through Gaussian noise to make the decoder robust to noise; the other encoder that obtains the variance, which makes the variance obtained from training converge to 1 to dynamically adjust the intensity of noise, thus further optimizing the decoder; the $\hat{X}$ obtained through the decoder is continuously optimized to approach X through the loss function.

4. Methods

In this paper, a variational auto-encoder was added to the deviation framework [16], which is instantiated as a variational deviation network. This method calculates the reference score and optimizes the deviation loss of the abnormal score network through a data-driven method. Finally, anomaly scores were obtained through anomaly score network training.

4.1. Overall Framework

Based on the combination of the variational encoder model, we introduced a new framework, which is mainly composed of three parts: anomaly score network, variational auto-encoder, and deviation loss function. These three parts were used to train the anomaly detection model. Our main contribution is that when generating reference scores, we adopted a data-driven approach-variational auto-encoder to make it more explanatory and superior to other mainstream models in terms of abnormal scores and data efficiency. As shown in Figure 3, we list the overall architecture of the variational auto-encoder model, which is mainly composed of three parts:

1. Anomaly score network (ϕ_x): Its main function is to generate an anomaly score for each input x.
2. Variational auto-encoder: In order to generate reference scores in a data-driven manner, we introduced a variational auto-encoder, which generates reference scores μ_R and σ_R through two encoders, and the reference scores generated by this method are more explanatory and can better fit the data itself and learn its reference scores.
3. Finally, we define the deviation loss function, and use ϕ_x, μ_R, and σ_R to guide the optimization of parameters. Through the deviation loss function, we can make the abnormal data, the reference score of the normal score μ_R has a significant deviation, and the normal data are closer to μ_R.

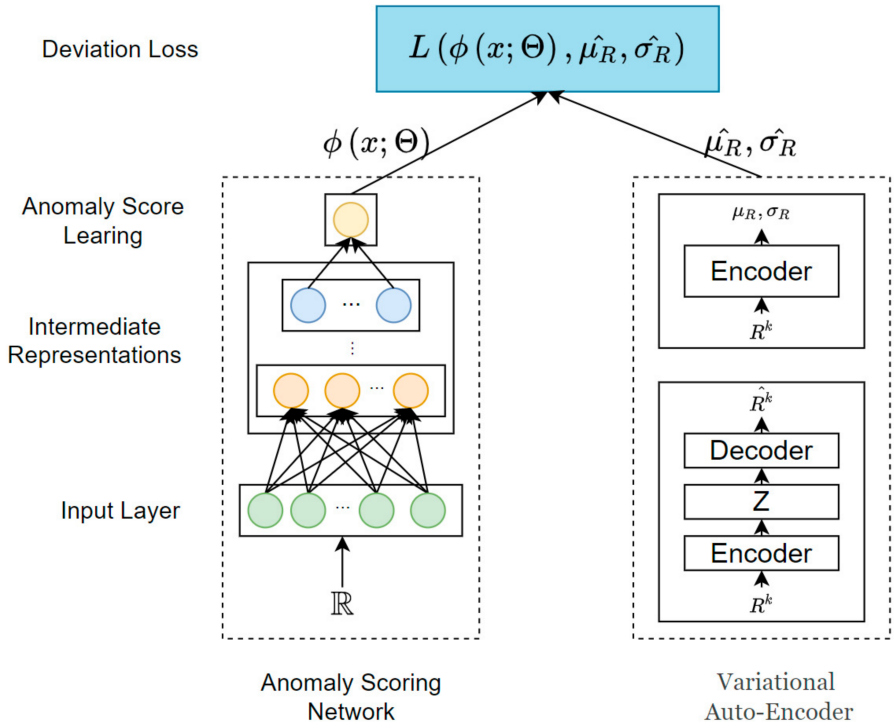


Figure 3. The overall framework of the variational deviation network. $\phi(x;\Theta)$ is an anomaly score learner with the parameters Θ; $\hat{\mu_R}$, $\hat{\sigma_R}$ are the reference scores of the training anomaly scores, which are the average mean and average variance of the normal data, where the normal data R^k are trained by the encoder in the variational self-coding to obtain the mean and variance.

4.2. Anomaly Score Network

The anomaly score network is mainly composed of three parts: the input layer, intermediate representations layer, and anomaly scorer (anomaly score learning). The anomaly score network is expressed as $\phi(x;\Theta) : \chi \mapsto \mathbb{R}$, where the intermediate presentation layer is expressed as $\eta(x;\Theta_t) : \chi \mapsto Q$, and the anomaly scorer is expressed as $\psi(x_t;\Theta_s) : Q \mapsto \mathbb{R}$, where $\Theta = \{\Theta_t, \Theta_s\}$.

The intermediate representations layer is a feature learning network with h hidden layers, in which the weight $\Theta_t = \{W_1, W_2, W_3, \ldots, W_h\}$ is specifically expressed as:

$$q = \eta(x;\Theta_t) \tag{1}$$

Among them, we can choose different hidden networks according to different types of data such as convolutional networks, recurrent networks, and multilayer perceptron networks.

Anomaly scorer, which uses a single neural unit to obtain anomalous scores through the feature representation output by the intermediate representation layer, is specifically expressed as:

$$\psi(x_t;\Theta_s) = \sum_{i=1}^{M} w_i^s q_i + w_{M+1}^s \tag{2}$$

where $q \in Q$, $\Theta_s = \{w^s\}$, w_{M+1}^s are the deviation parameters.

Therefore, the overall abnormal score network can be boiled down to:

$$\phi(x;\Theta) = \psi(\eta(x;\Theta_t);\Theta_s) \tag{3}$$

4.3. Variational Auto-Encoder

After the abnormal score network is built, it is necessary to define a reference score μ_R for normal data to guide the optimization of the abnormal score network. The generation of reference scores is mainly generated by two methods: data-driven and prior probability.

The main reason why we chose the data-driven method is its good interpretability. When dealing with different data, corresponding reference scores are generated to optimize the accuracy and data efficiency of abnormal scores.

4.3.1. Variational Auto-Encoder and Autoencoder

Variational self-encoding [27] is the same as autoencoder [28,29] in that both consist of an encoder and decoder, with the difference that the variational autoencoder returns a distribution in the hidden space by making the encoder rather than a single point. The autoencoder mainly uses neural networks as the encoder and decoder and uses iterative optimization to learn the best encoding–decoding scheme (as shown in Figure 4). Thus, in each iteration, we provide some data to the autoencoder structure, compare the encoded and then decoded output with the initial data, and update the weights of the network by back-propagating the error. Thus, intuitively, the entire autoencoder structure (encoder + decoder) constructs data bottlenecks, thus ensuring that only the major part of the information can pass through the bottleneck and be reconstructed. From our general framework, the considered encoder is defined by the encoder network structure, the considered decoder family is defined by the decoder network structure, and the reconstruction error reduction is performed by gradient descent of the encoder and decoder parameters. In contrast, variational self-encoding is performed by describing the probability distribution of each potential attribute through the encoder and adding a regular term to the returned distribution in the loss function to address the problem of irregularity in the hidden space to ensure a better organization of the hidden space. For example, in the field of image generation, an autoencoder model is trained. The autoencoder learns the relevant representations of pictures through neural network compression such as skin color, face shape, mouth, etc. in the field of portraits. The disadvantage of the autoencoder is that it obtains a single value through training, and uses it to describe the latent attributes of the image, thereby generating each attribute of the image, which is often not universal, and it cannot express the features well in the complex feature representation. However, the variational auto-encoder solves this challenging point. We describe the underlying attributes in probabilistic terms instead of training a single value. In this way, we now represent each potential attribute of a given input as a probability distribution. When decoding from a latent state, we randomly sample from each latent state distribution and generate a vector as input to the decoder model.

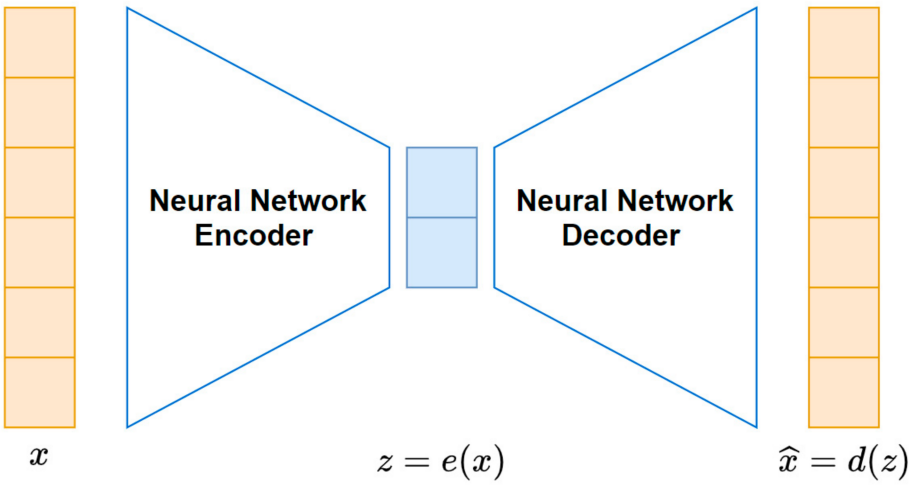


Figure 4. Autoencoder schematic. x is trained by the encoder to obtain the low-dimensional hidden space z. After reducing z to $\hat{x}$, which is similar to x by the decoder, we continuously optimized $\hat{x}$ by the loss function to keep $\hat{x}$ close to x.

4.3.2. Principle of Variational Auto-Encoder

The variational auto-encoder generates a latent space representation through the encoder, and then generates a $\hat{x}$ similar to the normal data x through the decoder, where we only know the original normal data x, and we want to infer the characteristics of z. The distribution, that is, calculating $p(z|x)$, is expressed as follows:

$$p(z|x) = \frac{p(x|z)p(z)}{p(x)} \tag{4}$$

However, calculating $p(x)$ is quite difficult, so let us approximate $p(z|x)$ with another distribution $q(z|x)$, we defined it to have a scalable distribution. If we can define the parameters of $q(z|x)$ so that it is very similar to $p(z|x)$, we can use it to approximate the complex distribution, so here we used the KL loss function. The KL dispersion degree is the difference between two probability distributions, so if we want to ensure that $q(z|x)$ is similar to $p(z|x)$, we can minimize the KL divergence between the two distributions.

$$minKL(q(z|x)||p(z|x)) \tag{5}$$

In the image generation model, we can use q to infer potentially hidden variables (latent states) that can be used to generate observations. We can further construct this model into a neural network structure, where the encoder model learns the mapping from x to z, and two neural networks are constructed in the encoder $\mu_k = f_1(x_k), log\sigma^2 = f_2(x_k)$ to calculate the mean and variance in the normal distribution, and the decoder model learns the mapping from z to x, as shown in Figure 5.

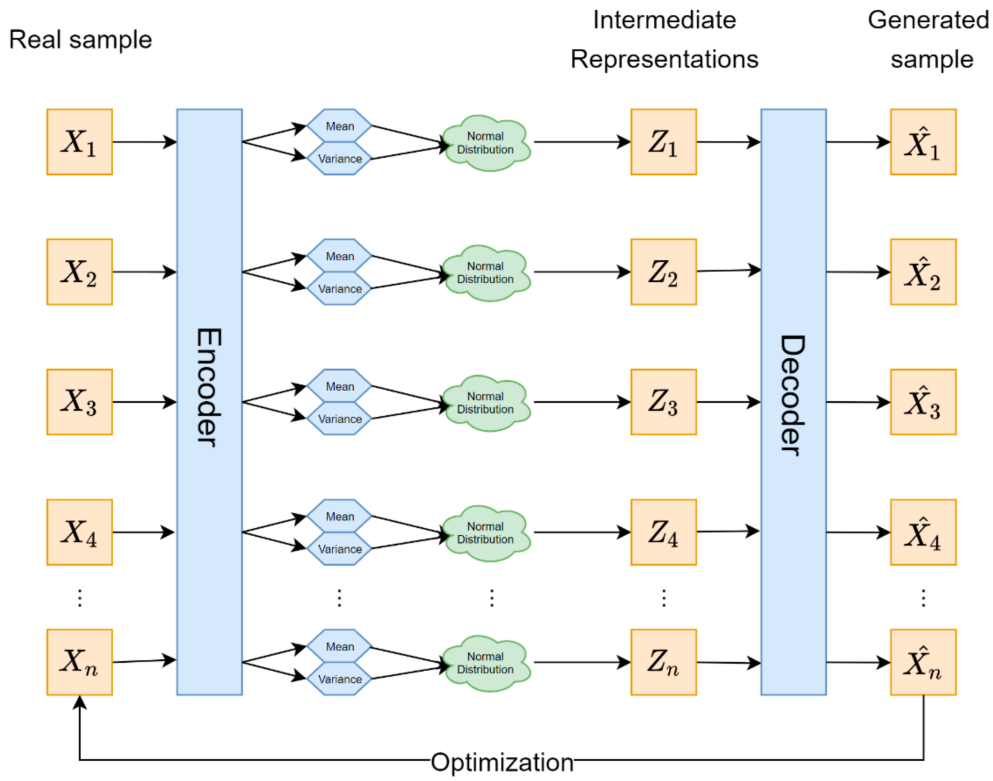


Figure 5. Schematic diagram of the basic principle of a variational auto-encoder. X is trained by two encoders separately to obtain the mean and variance, which are kept close to the normal distribution by an additional loss function to obtain a probability distribution Z conforming to X. Z is reduced to $\hat{X}$, similar to X by a decoder and continuously optimized by reconstructing the error loss function.

In order to prevent the noise from being zero, we stipulated that $p(z|x)$ were all aligned with the standard normal distribution while ensuring that the model had the ability to generate, that is, it could sample from $N(0,1)$ to generate an image.

The loss function includes two items. The first penalizes the reconstruction error, and the second encourages us to learn that the distribution $q(z|x)$ is similar to the real prior distribution $p(z)$. We assumed the unit Gaussian distribution, where each is the latent space of dimension j.

$$\mathcal{L}(x,\hat{x}) + \sum_{j} KL(q_j(z|x)||p(z)) \tag{6}$$

In order for $p(z|x)$ to be in line with the standard normal distribution, the loss function here is specifically expressed as:

$$\mathcal{L}(x,\hat{x}) + \sum_{j} KL(q_j(z|x)||N(0,1)) \tag{7}$$

4.3.3. Generate Reference Score

We used normal data $X = \{x_1, x_2, x_3, \ldots, x_n\}$ as the input of the variational auto-encoder, and the encoder generates the reference score of each data μ_R, σ_R, the potential space z for each probability distribution of the data is then generated by the decoder to generate $\hat{x}$, similar to x in the latent space z, and finally, the probability distribution of the latent space z is further optimized through the loss function (7) reference scores μ_R, σ_R. The reference scores μ_R and σ_R, generated through the training of the variational auto-encoder, were close to the normal distribution. Fortunately, according to [30], the normal distribution is very suitable for abnormal scores in many datasets, so we used data to drive the method of the variational auto-encoder, where the generated reference scores not only had good explanatory properties, but also generated different reference scores that were close to the normal distribution according to different data, which were more representative.

4.4. Deviation Loss Function

According to the reference score of the normal data generated by the variational auto-encoder, the deviation function to optimize the abnormal score is defined as:

$$dev(x) = \frac{\phi(x;\Theta) - \mu_R}{\sigma_R} \tag{8}$$

Among them, μ_R and σ_R are the reference scores that are close to the normal distribution generated by the variational auto-encoder through the encoder, and then the deviation function is used as the loss function to guide the optimization [31]:

$$L(\phi(x;\Theta), \mu_R, \sigma_R) = (1-y)|dev(x)| + ymax(0, a - dev(x)) \tag{9}$$

Among them, we define $y = 1$ as abnormal data, and $y = 0$ as normal data; according to the loss function in (9), when $y = 0$, the latter term of the loss function will be 0, and the former term will be $dev(x)$, optimizing the abnormal score so that the normal data are closer to the reference score μ_R, in which the abnormal score generated by the abnormal score network is deviated from μ_R. When $y = 1$, the first term of the loss function is 0. At this time, the loss function $max(0, a - dev(x))$ encourages the abnormal data to have a significant deviation from the reference score μ_R, so that the abnormal data deviate to the greatest extent reference data μ_R. Here, we used a = 5, which made a significant deviation.

In practical applications, there are only a few abnormal label data in the data. For normal data, we can solve this problem by specifying the data without abnormal labels as normal data. The data in the variational auto-encoder also uses this type of data and we generated reference scores by taking their average. The experimental results showed that this type of method is well realized when used. Normal data in anomaly detection applications may have very rare abnormal data, which are very limited for the optimization

of anomaly scores. In the experimental section, our variational bias network and other deep methods adopted this type of method to train the data where the effect was obvious.

4.5. Variational Deviation Network Algorithm

The variance deviation network algorithm consists of two main algorithms, where Algorithm 1 is mainly an algorithm for variance self-coding [18], which mainly generates reference scores, and Algorithm 2 composes an optimized anomaly scoring network by using the reference scores obtained from Algorithm 1.

Algorithm 1 realizes the whole process of generating the reference score. Because the reference score needs to be implemented in a data-driven manner, additional training of the variational auto-encoder is required here to ensure the accuracy and interpretability of the reference score. The training data $\mathbb{R}$ consist of R^k and R^n. R^k belongs to normal data, and R^n belongs to abnormal data. In Algorithm 1, the main input is normal data R^k. In algorithm steps 2–7, the weight of the encoder in the variational auto-encoder is mainly trained to obtain an encoder that can generate μ_R, σ_R. The fourth step is to optimize the loss function of the weight, and the eighth step is the input of the normal data into the trained encoder, the ninth step outputs the μ_R, σ_R of each normal data, and the tenth step finds the average $\overline{\mu}_R, \overline{\sigma}_R$. The main reference score calculated in the whole article was $\overline{\mu}_R, \overline{\sigma}_R$, μ_R, and σ_R; appearing elsewhere are the average reference scores.

Algorithm 1 Training the variational deviation network.

Input : $\chi \in R^k$-input normal data
Output : $\phi : \chi \mapsto \mu_R$-output reference score

1. Randomly initialize the weight $\Theta^{\mu}, \Theta^{\sigma}$
2. **for** i = 1 **to** n_epochs **do**
3. **for** j = 1 **to** n_batches **do**
4. loss $\leftarrow \mathcal{L}(x, \hat{x}) + \sum_j KL(q_j(z|x)||N(0,1))$
5. Optimize the weights $\Theta^{\mu}, \Theta^{\sigma}$ to perform gradient descent
6. **end for**
7. **end for**
8. χ input net of $\Theta^{\mu}, \Theta^{\sigma}$
9. output μ_R, σ_R
10. **return** $\overline{\mu}_R, \overline{\sigma}_R$ (Average)

Algorithm 2 is the whole process of realizing the entire variational deviation network.

Algorithm 2 mainly inputs $\mathbb{R}$ data. Steps 2–7 mainly train the weights of the abnormal score network. The fourth step is to optimize the loss function of the weights, which adds the reference score μ_R, σ_R values obtained in the first algorithm, and the final output of the algorithm is the trained variational deviation network model.

Algorithm 2 Training the variational deviation network.

Input : $\chi \in \mathbb{R}$-input training data
Output : $\phi : \chi \mapsto \mathbb{R}$-output the trained model

1. Randomly initialize the weight Θ
2. **for** i = 1 **to** n_epochs **do**
3. **for** j = 1 **to** n_batches **do**
4. $loss \leftarrow \frac{1}{b} \sum_{x \in \mathcal{B}} L(\phi(x; \Theta), \mu_R, \sigma_R)$
5. Optimize the weight Θ to perform gradient descent
6. **end for**
7. **end for**
8. **return** ϕ

5. Results

5.1. Dataset

In the experimental part, we used nine representative datasets (see Tables 1 and 2 for details) including five datasets with obvious abnormalities, four datasets with unobvious marked abnormalities, and training datasets in the experimental section including various fields such as the identification of credit card application fraud, false credit, etc.; network security: find out "intruders" from traffic data and identify new network intrusion patterns; and medical field: patients with abnormal thyroid function.

Table 1. AUC-ROC performance of V-DevNet and five competitive methods. The bolded data in the table indicates that the accuracy of the algorithm is higher compared to other algorithms.

Datasets	**AUC-ROC Performance**					
	V-DevNet	DevNet	REPEN	DSVDD	FSNet	iForest
donors	**1.000 ± 0.000**	1.000 ± 0.000	0.975 ± 0.005	0.995 ± 0.005	0.997 ± 0.002	0.874 ± 0.015
census	**0.856 ± 0.017**	0.828 ± 0.008	0.794 ± 0.005	0.835 ± 0.014	0.732 ± 0.020	0.624 ± 0.020
fraud	**0.988 ± 0.003**	0.980 ± 0.001	0.972 ± 0.003	0.977 ± 0.001	0.734 ± 0.046	0.953 ± 0.002
celeba	**0.967 ± 0.009**	0.951 ± 0.001	0.894 ± 0.005	0.944 ± 0.003	0.808 ± 0.027	0.698 ± 0.020
backdoor	**0.975 ± 0.023**	0.969 ± 0.004	0.878 ± 0.007	0.952 ± 0.018	0.928 ± 0.019	0.752 ± 0.021
URL	**0.985 ± 0.016**	0.977 ± 0.004	0.842 ± 0.006	0.908 ± 0.027	0.786 ± 0.047	0.720 ± 0.032
campaign	**0.914 ± 0.015**	0.807 ± 0.006	0.723 ± 0.006	0.748 ± 0.019	0.623 ± 0.024	0.731 ± 0.015
news20	0.948 ± 0.002	**0.950 ± 0.007**	0.885 ± 0.003	0.887 ± 0.000	0.578 ± 0.050	0.328 ± 0.016
thyroid	**0.831 ± 0.005**	0.783 ± 0.003	0.580 ± 0.016	0.749 ± 0.011	0.564 ± 0.017	0.688 ± 0.020

Table 2. AUC-PR performance of V-DevNet and five competitive methods. The bolded data in the table indicates that the accuracy of the algorithm is higher compared to other algorithms.

Data Sets	**AUC-PR Performance**					
	V-DevNet	DevNet	REPEN	DSVDD	FSNet	iForest
donors	**1.000 ± 0.000**	1.000 ± 0.000	0.508 ± 0.048	0.846 ± 0.114	0.994 ± 0.002	0.221 ± 0.025
census	**0.368 ± 0.015**	0.321 ± 0.004	0.164 ± 0.003	0.291 ± 0.008	0.193 ± 0.019	0.076 ± 0.004
fraud	**0.713 ± 0.004**	0.690 ± 0.002	0.674 ± 0.004	0.688 ± 0.004	0.043 ± 0.021	0.254 ± 0.043
celeba	**0.326 ± 0.006**	0.279 ± 0.009	0.161 ± 0.006	0.261 ± 0.008	0.085 ± 0.012	0.065 ± 0.006
backdoor	**0.886 ± 0.002**	0.883 ± 0.008	0.116 ± 0.003	0.856 ± 0.016	0.573 ± 0.167	0.051 ± 0.005
URL	**0.695 ± 0.015**	0.681 ± 0.022	0.103 ± 0.003	0.475 ± 0.040	0.149 ± 0.076	0.066 ± 0.012
campaign	**0.426 ± 0.009**	0.381 ± 0.008	0.330 ± 0.009	0.349 ± 0.023	0.193 ± 0.012	0.328 ± 0.022
news20	0.650 ± 0.003	**0.653 ± 0.009**	0.222 ± 0.004	0.253 ± 0.001	0.082 ± 0.010	0.035 ± 0.002
thyroid	**0.386 ± 0.003**	0.274 ± 0.011	0.093 ± 0.005	0.241 ± 0.009	0.116 ± 0.014	0.166 ± 0.017

5.2. Comparison Algorithm

The variational deviation algorithm will be compared with the deviation algorithm [16], REPEN algorithm [5], Deep-SVDD algorithm [6], FSNet algorithm (algorithm for prototypical networks for few-shot learning) [32], and iForest (algorithm for isolation-based anomaly detection) [17], which were the five algorithms used for verification, because these five algorithms are currently the most advanced methods in the field of anomaly detection. The deviation algorithm is an end-to-end anomaly detection algorithm; the REPEN algorithm is a semi-supervised deep learning anomaly detection algorithm that solves a small amount of abnormal data; and the Deep-SVDD algorithm is a feature representation of learning normal data. The FSNet algorithm is an anomaly detection algorithm that solves small sample learning detection. The iForest algorithm is a typical anomaly detection algorithm in unsupervised learning. These algorithms are implemented in the python language. Among them, V-DevNet, DevNet, Deep- SVDD, FSNet, and REPEN are implemented using keras [33], and the iForest algorithm is called in scikit-learn. In the training, the first four algorithms all used the gradient optimization algorithm of [34] for training.

5.3. Experiment Settings

Since our experimental data were multidimensional and disordered, the new model we designed here mainly consisted of a multilayer perceptron (MLP) network structure. Among the variational self-coding and anomaly score networks, we designed two architectures, one with only one hidden layer and 20 neural units, and another architecture with three hidden layers, in order to train more dimensional and complex data. The first hidden layer contains 1000 neural units, the second hidden layer contains 250 neural units, and the third hidden layer contains 20 units; here, we also applied the relu function because of its efficient computation and gradient propagation. Table 1 shows the results of architecture 1, which was trained under the structure of one hidden layer. In the above comparison algorithm, they were trained using 50 epochs, with 20 min-batches in each epoch.

In a real-world anomaly detection environment, there are a large number of unlabeled data objects and a very small set of anomalous data; to fit this scenario, the anomalous and normal objects of each dataset were first divided into two subsets, where 80% of the data were used as the training set and 20% as the test set. To conform to the realistic scenario, we randomly added/removed anomalous data in each training dataset so that the anomalous data accounted for 2% of the training data, and the resulting data constitutes the unlabeled training dataset U. We further randomly selected 30 anomalies from the anomaly class as the a priori knowledge of the anomalies of interest (i.e., the labeled anomaly set K, which accounts for only 0.005–1% of all training data objects), thus constituting a realistic anomaly detection environment.

5.4. Performance Evaluation Method

In this experiment, we mainly used two important evaluation indicators, AUC-ROC and AUC-PR. Before understanding these two types of indicators, we need to understand the relevant terms. Please refer to Table 3 for details.

Table 3. Confusion matrix. When the data label is positive and the forecast is also positive, it is called the true positive. When the data label is negative and the forecast is positive, it is called the false positive. When the data label is positive and the forecast is negative, it is called the false negative. When the data label is negative and the forecast is also negative, it is called the true negative.

	Actual Positive	**Actual Negative**
Predicted positive	True positive, TP	False positive, FP
Predicted negative	False negative, FN	True negative, TN

In the AUC-ROC evaluation index, we need to use two reference values, one of which is the true positive ratio:

$$TPR = \frac{TP}{TP + FN} \tag{10}$$

This represents the ratio between the positive examples in the prediction and all the positive examples in the dataset, which can represent the recall rate of positive examples; one of which is the false positive ratio:

$$FPR = \frac{FP}{FP + PN} \tag{11}$$

This represents the ratio between the negative example in the prediction and all the negative examples in the dataset, which represents the recall rate of the negative examples. The ROC graph is made based on these two reference values. When making the ROC graph, the TPR is the y-axis and the FPR is the x-axis. Draw a straight line in the ROC chart $x = a$, where the straight line and the ROC curve focus $(a, y(a))$ of an algorithm express how much positive case recall rate the algorithm can have when the counterexample recall rate reaches a; the same is true for horizontal straight lines. In order to generate a single value as a reference indicator, the concept of AUC (area under curve) is added here, that is, the

size of the total area under the curve in the ROC graph. When the negative example recall rate is small, and the positive example recall rate is large, the performance of the algorithm is better. At this time, the ROC curve should tend to the upper left, as "left convex" as possible, and the area under the curve becomes larger, so the larger the ROC-AUC, the better the algorithm performance.

Two reference values are also needed in the AUC-PR evaluation index, one is the precision:

$$P = \frac{TP}{TP + FP} \tag{12}$$

This can be expressed as the ratio between the positive example in the prediction and all the predictions, indicating the accuracy of the positive example in the prediction; the other is the recall rate:

$$R = \frac{TP}{TP + FN} \tag{13}$$

The ratio between the positive example in the prediction and all the positive examples in the dataset represents the recall rate of the positive example; when plotting the PR graph, use P as the y-axis and R as the x-axis. Generally, the PR curve is when x is greater than a certain value a, which behaves as a decreasing function. When the dataset is unchanged and the algorithm performance does not change, to increase the recall rate (x-axis), the total number of predictions needs to be increased. At this time, the number of inverse columns predicted will increase, resulting in positive examples in the prediction and the accuracy (y-axis) decreases. Similarly, in order to generate a single value as a reference indicator, AUC (area under curve) is also added here. When the algorithm can achieve a higher recall rate and accuracy of positive examples when the total number of predictions is small, it indicates that the performance of the algorithm is good. At this time, the PR curve tends to the upper right, that is, the better the performance of the curve, then it should be as "right convex" as possible at this time, and the area under the curve should be as large as possible. Therefore, the larger the AUC-PR, the better the algorithm performance.

The study in [35] found that when the data in the dataset was out of balance, that is, the difference between positive and negative examples was very large, which is the same as the characteristics of anomaly detection data, this led to a smaller difference in AUC-ROC, and AUC-PR had more effective differences between performance algorithms. In order to further improve the accuracy of the evaluation index, we applied the method of calculating the average accuracy in [36] to calculate AUC-PR, where AUC-ROC was used as a reference index to further measure the superiority of the algorithm. In order to prove the superiority of V-DevNet, we added the Wilcoxon signed-rank test [37], where the idea of this method is to take the rank of the absolute value of the difference between the observed values and the central position assumed by the original hypothesis (or the two sets of sample values) and add them up separately according to different signs as its test statistic.

In using the above evaluation method, the relevant AUC-PR and AUC-ROC indicators of the variational deviation network and the other five comparison methods are shown in Tables 1 and 2. In the above table, we can see that the variational deviation network in data 7 and 9 showed the best performance. Among them, under the AUC-ROC performance index, the variance deviation network with other experimental methods by ANOVA test, we found that the p-value was less than 0.05, so they were significantly different; by comparing the mean, we found that the performance of the variational deviation network was significantly higher than that of the deviation network (2%), REPEN (12%), Deep-SVDD (6%), FSNet (25%), iForest (33%). Under the AUC-PR performance index of the variance deviation network with other experimental methods by the ANOVA test, the p-value was also less than 0.05, so they were significantly different; by comparing the mean, our experiment found that the performance of the variational deviation network was significantly higher than that of the deviation network (6%), REPEN (130%), Deep-SVDD (30%), FSNet (124%), iForest (332%). The experimental results showed that we used variational self-coding to learn the reference scores with a data-driven approach and to guide the optimization of the

deviation network with the reference scores, resulting in significantly higher experimental results than the other methods. In the study of time complexity, we found that the overall running time of V-DevNet grew linearly with the size and dimensionality of the data, which provides a strong guarantee for the use of the algorithm.

6. Conclusions

This article added a variational auto-encoder model on the basis of the deviation network, and instantiated a new V-DevNet framework. Instead of learning the reconstructed features, the framework learns the normal distribution of each data as a reference score. Through training in a data-driven manner, the values of μ_R and σ_R that match the data are obtained. It can more accurately indicate the normal distribution of normal labeled data, and has good solvability. The performance in the two indicators of AUC-ROC and AUC-PR was higher than that of the deviation network (reference score was randomly generated), and was significantly higher than the other four types of anomaly detection methods (specific comparative data are shown in Table 1).

However, the method proposed in this paper has certain limitations. Because the method in this paper was based on variational self-coding, in order to train the probability distribution of normal data in variational self-coding, here, the normal distribution was used to optimize the probability distribution of normal data to make it close to the mean and variance obtained from the normal distribution to optimize the reference score, although a previous paper based on references mentioned that most of the data conformed to the normal distribution, but could not be fully covered as all normal distribution, so further research will follow to see if the variational self-coding can fit other probability distributions.

In the future, we will consider using V-DevNet in the anomaly detection of image data and text data. We can also add a convolutional neural network to the anomaly scoring network in V-DevNet. I believe that it can be better in V-DevNet. The normal distribution of images and text can be obtained, which will further improve the performance index of V-DevNet.

Author Contributions: Conceptualization, J.W. and J.L.; Methodology, J.W.; Writing—original draft preparation, J.W.; Writing—review and editing, J.L., X.W. and K.W.; Supervision, J.L., X.W., G.L. and K.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Fujian Middle and Youth Project (JAT190679).

Data Availability Statement: Not applicable as the study does not report any data.

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

References

1. Moustafa, N.; Slay, J. UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems (UNSW-NB15 Network Data Set). In Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS), Canberra, Australia, 10–12 November 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 1–6.
2. Thapa, N.; Liu, Z.; Kc, D.B.; Gokaraju, B.; Roy, K. Comparison of Machine Learning and Deep Learning Models for Network Intrusion Detection Systems. *Future Internet* **2020**, *12*, 167. [CrossRef]
3. Ryan, S.; Corizzo, R.; Kiringa, I.; Japkowicz, N. Pattern and Anomaly Localization in Complex and Dynamic Data. In Proceedings of the 2019 18th IEEE International Conference on Machine Learning and Applications (ICMLA), Boca Raton, FL, USA, 16–19 December 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1756–1763.
4. Vlaminck, M.; Heidbuchel, R.; Philips, W.; Luong, H. Region-Based CNN for Anomaly Detection in PV Power Plants Using Aerial Imagery. *Sensors* **2022**, *22*, 1244. [CrossRef] [PubMed]
5. Pang, G.; Cao, L.; Chen, L.; Liu, H. Learning Representations of Ultrahigh-Dimensional Data for Random Distance-Based Outlier Detection. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 2041–2050.
6. Ruff, L.; Vandermeulen, R.; Goernitz, N.; Deecke, L.; Siddiqui, S.A.; Binder, A.; Müller, E.; Kloft, M. Deep One-Class Classification. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; pp. 4393–4402.
7. LeCun, Y.; Bengio, Y.; Hinton, G. Deep Learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]

8. Chen, J.; Sathe, S.; Aggarwal, C.; Turaga, D. Outlier Detection with Autoencoder Ensembles. In Proceedings of the 2017 SIAM International Conference on Data Mining, Houston, TX, USA, 27–29 April 2017; SIAM: Philadelphia, PA, USA, 2017; pp. 90–98.
9. Hawkins, S.; He, H.; Williams, G.; Baxter, R. Outlier Detection Using Replicator Neural Networks. In Proceedings of the International Conference on Data Warehousing and Knowledge Discovery, Vienna, Austria, 3–6 September 2002; Springer: Berlin/Heidelberg, Germany, 2002; pp. 170–180.
10. Schlegl, T.; Seeböck, P.; Waldstein, S.M.; Schmidt-Erfurth, U.; Langs, G. Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery. In Proceedings of the International Conference on Information Processing in Medical Imaging, Boone, NC, USA, 25–30 June 2017; Springer: Berlin/Heidelberg, Germany, 2017; pp. 146–157.
11. Zenati, H.; Romain, M.; Foo, C.-S.; Lecouat, B.; Chandrasekhar, V. Adversarially Learned Anomaly Detection. In Proceedings of the 2018 IEEE International Conference on Data Mining (ICDM), Beijing, China, 17–20 November 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 727–736.
12. Zhou, C.; Paffenroth, R.C. Anomaly Detection with Robust Deep Autoencoders. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, 13–17 August 2017; pp. 665–674.
13. Sansone, E.; De Natale, F.G.; Zhou, Z.-H. Efficient Training for Positive Unlabeled Learning. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *41*, 2584–2598. [CrossRef] [PubMed]
14. Aggarwal, C.C. Supervised Outlier Detection. In *Outlier Analysis*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 219–248.
15. Li, X.; Liu, B. Learning to Classify Texts Using Positive and Unlabeled Data. In Proceedings of the IJCAI, Acapulco, Mexico, 9–15 August 2003; Citeseer: State College, PA, USA, 2003; Volume 3, pp. 587–592.
16. Pang, G.; Shen, C.; van den Hengel, A. Deep Anomaly Detection with Deviation Networks. In Proceedings of the Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 353–362.
17. Liu, F.T.; Ting, K.M.; Zhou, Z.-H. Isolation-Based Anomaly Detection. *ACM Trans. Knowl. Discov. Data TKDD* **2012**, *6*, 1–39. [CrossRef]
18. Keller, F.; Muller, E.; Bohm, K. HiCS: High Contrast Subspaces for Density-Based Outlier Ranking. In Proceedings of the 2012 IEEE 28th International Conference on Data Engineering, Arlington, VA, USA, 1–5 April 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 1037–1048.
19. Pang, G.; Cao, L.; Chen, L.; Lian, D.; Liu, H. Sparse Modeling-Based Sequential Ensemble Learning for Effective Outlier Detection in High-Dimensional Numeric Data. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018.
20. Li, P.; Hastie, T.J.; Church, K.W. Very Sparse Random Projections. In Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, 20–23 August 2006; pp. 287–296.
21. Tax, D.M.; Duin, R.P. Support Vector Data Description. *Mach. Learn.* **2004**, *54*, 45–66. [CrossRef]
22. Elkan, C.; Noto, K. Learning Classifiers from only Positive and Unlabeled Data. In Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, NV, USA, 24–27 August 2008; pp. 213–220.
23. Ma, Y.; Lei, Y.; Wang, T. A Natural Scene Recognition Learning Based on Label Correlation. In *IEEE Transactions on Emerging Topics in Computational Intelligence*; IEEE: Piscataway, NJ, USA, 2020.
24. Fei-Fei, L.; Fergus, R.; Perona, P. One-Shot Learning of Object Categories. *IEEE Trans. Pattern Anal. Mach. Intell.* **2006**, *28*, 594–611. [CrossRef] [PubMed]
25. Kingma, D.P.; Welling, M. Auto-Encoding Variational Bayes. *arXiv* **2013**, arXiv:1312.6114.
26. Xiang, Y.; Cobben, J.F. A Bayesian Approach for Fault Location in Medium Voltage Grids with Underground Cables. *IEEE Power Energy Technol. Syst. J.* **2015**, *2*, 116–124. [CrossRef]
27. Doersch, C. Tutorial on Variational Autoencoders. *arXiv* **2016**, arXiv:1606.05908.
28. Oluwasanmi, A.; Aftab, M.U.; Baagyere, E.; Qin, Z.; Ahmad, M.; Mazzara, M. Attention Autoencoder for Generative Latent Representational Learning in Anomaly Detection. *Sensors* **2022**, *22*, 123. [CrossRef] [PubMed]
29. Corizzo, R.; Ceci, M.; Pio, G.; Mignone, P.; Japkowicz, N. Spatially-Aware Autoencoders for Detecting Contextual Anomalies in Geo-Distributed Data. In Proceedings of the International Conference on Discovery Science, Halifax, NS, Canada, 11–13 October 2021; Springer: Berlin/Heidelberg, Germany, 2021; pp. 461–471.
30. Kriegel, H.-P.; Kroger, P.; Schubert, E.; Zimek, A. Interpreting and Unifying Outlier Scores. In Proceedings of the 2011 SIAM International Conference on Data Mining, Philadelphia, PA, USA, 6 April 2011; SIAM: Philadelphia, PA, USA, 2011; pp. 13–24.
31. Hadsell, R.; Chopra, S.; LeCun, Y. Dimensionality Reduction by Learning an Invariant Mapping. In Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), New York, NY, USA, 22 June 2006; IEEE: Piscataway, NJ, USA, 2006; Volume 2, pp. 1735–1742.
32. Snell, J.; Swersky, K.; Zemel, R.S. Prototypical Networks for Few-Shot Learning. *arXiv* **2017**, arXiv:1703.05175.
33. Gulli, A.; Pal, S. *Deep Learning with Keras*; Packt Publishing Ltd.: Birmingham, UK, 2017.
34. Hinton, G.; Srivastava, N.; Swersky, K. Neural Networks for Machine Learning Lecture 6a Overview of Mini-Batch Gradient Descent. Available online: https://www.cs.toronto.edu/~{}tijmen/csc321/slides/lecture_slides_lec6.pdf (accessed on 28 February 2022).
35. Davis, J.; Goadrich, M. The Relationship between Precision-Recall and ROC Curves. In Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, PA, USA, 25–29 June 2006; pp. 233–240.

36. Schütze, H.; Manning, C.D.; Raghavan, P. *Introduction to Information Retrieval*; Cambridge University Press Cambridge: Cambridge, UK, 2008; Volume 39.
37. Woolson, R.F. Wilcoxon Signed-Rank Test. In *Wiley Encyclopedia of Clinical Trials*; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 2007; pp. 1–3.