Anomaly detection for high-dimensional space using deep hypersphere fused with probability approach

Jian Zheng · Jingyi Li · Cong Liu · Jianfeng Wang · Jiang Li · Hongling Liu

Abstract
Data distribution presents sparsity in a high-dimensional space, thus difficulty affording sufficient information to distinguish anomalies from normal instances. Moreover, a high-dimensional space may exist many subspaces, obviously, anomalies can exist in any subspaces. This also creates trouble for anomaly mining. Consequently, it is a challenge for anomaly mining in a high-dimensional space. To address this, here proposed a deep hypersphere method fused with probabilistic approach for anomaly mining. In the proposed method, the deep neural network is used as a feature extractor to capture those layered low-dimensional features from the data lying in a high-dimensional space. To promote the ability of the deep neural network to capture these features, the probability approach of sample binary-classification is fused into the loss function, thereby forming the probability deep neural network. Then, the hypersphere is used as an anomalous detector. In the low-dimensional features extracted by the deep neural network, the anomalous detector separates anomaly features from normal features. Finally, experimental results on synthetic and real-world data sets show that the proposed method not only outperforms the state-of-the-art methods in the precision of mined anomalies, but also this hybrid method consisting of deep neural networks and traditional detection methods has outstanding capabilities of mining high-dimensional anomalies. We find that deep neural networks fusing the probabilistic method of sample multi-classification can capture these desired low-dimensional features; moreover, these captured low-dimensional features present more obvious layered characteristics. We also demonstrate that as long as these captured features represent a fewer anomaly instances, it can sufficiently identify anomalies from normal instances.

Keywords Anomaly mining · Deep neural network · High-dimensional space · Hypersphere

Introduction
The “curse of dimensionality” brings three major challenges for anomaly mining. The first one is the computational power of detection methods. High-dimensional space may exist many subspaces, and anomalies may exist any subspaces. A brute-force method is computationally prohibitive because of searching an exponential number of subspaces. Another challenge is that the relative contrast between data objects become more and more similar in a high-dimensional space [1, 2]. Most existing anomaly detection methods implicitly or explicitly rely on the distance between data objects. For a low-dimensional space, the distance assessment metrics are easy to mine anomalies, such as the distance-based methods in [3, 4]. Unfortunately, in a high-dimensional space, the metric of assessing distance may not be able to measure the similarity between data [5–7]. The third challenge is that data distribution becomes sparse in a high-dimensional space, thereby hard providing sufficient information to distinguish anomalies from normal instances [8–10]. Overall, it is a tough task to mine anomalies existing in a high-dimensional space.

Currently, anomaly detection methods are divided into the following categories: (I) distance-based, such as K-Nearest Neighbor (KNN) [3, 4]. Although such method does not have to assume data distribution and require training samples, the distance between data in a high-dimensional space...
is not easy to calculate. (II) Cluster-based, e.g., the model in [11], such method requires assume data distribution. (III) Reconstruction error-based, for this method, the threshold is set in advance [12], such as Matrix Factorization (MF) [13]. When the reconstructed error is larger than set the threshold, this is considered to be anomalies. If threshold is not set properly, the precision of detection methods suffers from seriously negative effects. (IV) Classification-based, e.g., One Class-Support Vector Machine (OC-SVM) [14], as for this method, anomalies are linearly separated by SVM from normal instances but the curse of dimensionality limits the ability of SVM to linearly separate features [15, 16]. (V) Deep network architectures-based, e.g., Deep Autoencoder (DAE) [17], Generative Adversarial Networks (GANs) [18], deep networks can capture the layered features used to identify anomalies and normal instances from the background space [19, 20]. For instance, GANs exhibit excellent ability for anomaly detection even if the reconstructed anomaly instances are very poor [21]. Nevertheless, GANs are prone to pattern collapse during training. In addition, Goh et al. use recurrent neural networks (RNNs) [22] for anomaly detection in cyber physical systems. (VI) Hybrid method-based, i.e., consisting of deep networks and tradition detection methods, such as Deep Neural Networks based K-classification (DNN-K) [23], Deep Neural Networks–Support Vector Machine (DNN–SVM) [24], and Deep Neural network–support Vector Data Description (DNN–SVDD) [25]. Hybrid methods are much more extensible while the computational complexity augments as the depth of hybrid architecture increases [26].

In addition to above involved methods, hyperspheres are also commonly used for anomaly detection. For example, in [27, 28], the hypersphere is used for binary-classification of normal classes and anomaly classes. Usually, hyperspheres are sensitive to a lack of data, moreover, have also no advantages at learning complex invariants. Whereas, in a good feature space, hyperspheres can exhibit attractive capability of separating data.

Given these complementary advantages of a hypersphere and a deep network, this is very attractive to study a hybrid methods of both for anomaly detection. In this work, our motivation is to mine a limited number of potential anomalies existing in a high-dimensional space. In addition, we also look at exploring the searching efficiency of subspaces existing in a high-dimensional space, aiming at giving the computational complexity of anomaly detection algorithms. Hence, we developed a hybrid model consisting of a hypersphere and a deep neural network to meet our studied targets. First, the deep neural network in the proposed model captures these layered low-dimensional features from high-dimensional data. To promote the ability of the deep neural network to learn these layered low-dimensional features, we consider the probability approach of sample binary-classification to be fused into the loss function, thereby forming the probability deep neural network. Then, the hypersphere linearly separates these captured low-dimensional features. Finally, the proposed method is verified on synthetic and real-world data sets.

We summarize the main contributions of this work as follows.

(1) High dimensionality increases the complexity of the data space, facing to the curse of dimensionality, the deep neural networks fusing the probabilistic method of sample multi-classification can capture these desired low-dimensional features from high-dimensional data; moreover, these captured low-dimensional features present more significant layered characteristics.

(2) As long as these extracted features can represent a fewer anomalous instances, it is sufficient to identify anomalies from normal instances.

(3) Upon a high-dimensional space, the hybrid approach composing of a deep neural network and a traditional detection method has stronger mining power than deep detection approaches or traditional detection approaches.

**Methodology**

**Layered features extraction**

Usually, the background space is high dimension, which is not conducive for anomaly mining, so we first capture these low-dimensional features from the data in the background space using deep neural networks. This purpose is to reduce the dimensionality of a searching space for anomaly mining.

The loss function is one of the critical hyper-parameters for deep neural networks because of affecting the learning capability of deep neural networks [29–31]. Anomaly detection can be treated to be binary-classification of samples, i.e., anomaly and normal classes, based on this, we consider the loss function from the perspective of calculating sample classification probability.

Given a sample set \( x = \{x_1, x_2, \ldots, x_l\} \), and \( l > 0 \). Item \( P(C_i) \) denotes the probability that the point \( x_j \) should be in class \( C_i \in \text{Class list } C \), and \( i = 1,2, \ldots, m \). In addition, let us assume that the classification is mutually independent, i.e., conditional independence. The result of conditional independence is given in the following equation:

\[
P(C|s_j) = P(C_1, C_2, ..., C_i|s_j) = \prod_{i=1}^{m} P(C_i|s_j),
\]  

(1)
where \( s_j \) is the class \( j \) from the class set. The posterior probability for tagging \( x \) is as follows:

\[
P(s_j|C) = \frac{P(s_j)P(C|s_j)}{P(C)} = \frac{P(s_j) \prod_{i=1}^{m} P(C_i|x_j)}{p(C)},
\]

\( j = 1, 2, \ldots \).

Since the denominator in Eq. (2) is not based on \( s_j \), this part can be ignored [32]. Therefore, \( s_j \) can be calculated using the following equation:

\[
\Omega(x) \propto P(s_j) \prod_{i=1}^{m} P(C_i|x_j).
\]

(3)

According to the [32], the probability estimate \( P(C_i|x_j) \) and the prior probability for class \( C_j \) are replaced using \( \frac{M_i(j,u_i)}{D_j+1} \) and \( \frac{x_j}{n} \), respectively. Hence, Eq. (3) is modified as follows:

\[
\Omega(x) \propto \left\{ \frac{m}{\prod_{i=1}^{m} M_i(k, u_i) + \frac{1}{B}} \right\}^B,
\]

(4)

where \( C \) is the number of the classes. \( M_i(k, u_i) \) represents the number of data set elements which have the class of \( s_k \), and are assigned to class \( s_m \). \( D_k \) determines the number of class \( s_m \). \( B \) is a constant. In regard to the \( B \) value, Titterington [32] has proposed the values 1, 0.8, or 0.5 as a reference. For the detailed proof respecting Eqs. (3) and (4), please see the [32]. Let us assume \( C_1 \) and \( C_2 \) are anomaly classes and normal classes, respectively. As such, we can simplify Eq. (4), having that

\[
\Omega(x) \propto \left\{ \frac{3}{\prod_{i=1}^{3} M_i(1, 2) + \frac{1}{3}} \right\}^B.
\]

(5)

Equation (5) demonstrates the probability that the point \( x \) should be classified into class \( C_i \) is \( \Omega(x) \).

Sparsity can be encouraged by adding a regularization term that takes a large value when the average activation value \( \bar{\rho}_i \) of a neuron \( i \) and its desired value \( \rho_i \), are not close in the value [33]. One such sparsity regularization term can be the \( K-L \) divergence, as follows:

\[
KL_{\text{sparse}} = \sum_{i=1}^{n} \rho \log \left( \frac{\rho}{\bar{\rho}_i} \right) + (1-\rho) \log \left( \frac{1-\rho}{1-\bar{\rho}_i} \right).
\]

(6)

We calculate the average output activation measure of the neuron \( i \) with probability manner, having

\[
\hat{\rho}_i = \frac{1}{n} \sum_{j=1}^{n} h \left( w_i^{(1)^T} x_j + \Omega(x) + b_i^{(1)} \right),
\]

where \( n \) is the total number of training examples. \( w_i^{(1)^T} \) is the ith row of the weight matrix \( W^{(1)} \), and \( b_i^{(1)} \) is the ith entry of the bias vector \( b^{(1)} \). The loss function \( L(w, b) \) is given in the following equation:

\[
L(w, b) = ||e - \hat{e}|| + KL_{\text{sparse}},
\]

(8)

where \( e \) and \( \hat{e} \) are the inputting and the reconstructed inputting, respectively. Equation (8) demonstrates the probability that anomalous points and normal points should belong to the \( C_1 \) classes and \( C_2 \) classes in a given sample.

**Anomaly separation**

Hypersphere can be defined with center \( a \) and radius \( R \), given a data set \( y = \{ y_1, y_2, \ldots, y_i, \ldots \}, i = 1, 2, \ldots \). the error function using a hypersphere to learn a compact space around \( y_i \) can be defined as follows [25]:

\[
\Delta(a, r) = R^2 + \lambda \sum_{i} \xi_i
\]

\[
||y_i - a||^2 \leq R^2 + \xi_i, \forall i = 1, 2, \ldots, \xi_i \geq 0
\]

where \( \xi_i \) is a slack variable that lets some data points fall outside the hypersphere. \( \xi_i \) is an auxiliary variable rather than part of the parameter set. \( \lambda \) is used for the proportion to trade-off data outside the hypersphere, allowing users to predefine. The distance between the sample and the center of the sphere can be calculated (the detailed derivation see [25]) as follows [25]:

\[
y - a^2 = (y^2 - y) - 2 \sum_{i} \alpha_i \xi_i y_i + \sum_{i, j} \alpha_i \alpha_j \xi_i \xi_j y_j
\]

(10)

where \( \alpha_i \) and \( \alpha_j \) are Lagrange multipliers, and \( j = 1, 2, \ldots \). Using the mapping transformation, Eq. (10) can be converted into the following equation:

\[
\varphi(y) - a^2 = K(y, y) - 2 \sum_{i} \alpha_i K(y, y) + \sum_{i, j} \alpha_i \alpha_j K(y_i, y_j)
\]

(11)

where \( \varphi() \) is a mapping function. \( K() \) is a kernel function of satisfying Mercer theorem. Certainly, there are many the kernel functions of satisfying Mercer theorem. In this work, we select the Matern52 kernel in [34], having that

\[
K(y, y) = \theta_0 \left( 1 + \sqrt{C_r \xi^2} \right) + A_r \xi^2 \exp \left\{ -\sqrt{B_r \xi^2} \right\}
\]

(12)

where \( C_r, A_r, B_r \) are constant coefficients.
There are two reasons for selecting the Matern52 kernel as our kernel function, (i) the Matern52 kernel can make radius warping concave and non-decreasing [34, 35], so as to be prone to focus more on areas with small radii. (ii) The Matern52 kernel, which is a continuous positive definite kernel, can flexibly control the searches in the normal data region because of being non-stationary [36]. This is very conducive to promoting the separated accuracy of anomaly and normal features.

The output of a hypersphere can be calculated in the following equation:

\[ R^2 - \varphi(y) = a^2. \]  

(13)

In Eq. (13), the sample is normal if the output is a positive value. Otherwise, the sample is anomalies.

The proposed model is composed of the deep neural network and the hypersphere, so the final learning function \( \nabla(L(w, b), \Delta(a, r)) \) of our model includes the loss function \( L(w, b) \) in Eq. (8) of the probability deep neural network and the error function \( \Delta(a, r) \) in Eq. (9) of the hypersphere, as follows:

\[ \nabla(L(w, b), \Delta(a, r)) = \min(L(w, b) + \Delta(a, r)). \]  

(14)

Model

In this section, we interpret the rationality of the proposed model and describe the model architecture. In addition, some hyper parameters are configured, as well as, the model training is also presented.

Rationality

Since the background space is high dimension, anomaly detection is considered to be performed in the low-dimensional feature space, instead of the background space. Based on this fact, we opt for the deep neural network to capture low-dimensional features from high-dimensional data. To allow these captured low-dimensional features more layered, the binary-classification probability of samples in Eq. (5) is fused into the loss function in Eq. (8). Our though originates from the fact that anomaly detection can be treated as a binary-classification of samples, so we consider the probability approach of sample binary-classification.

As for the proposed model, the performance not only relies on these captured low-dimensional features, but also depends on the kernel in the hypersphere. In view of this, using the kernel in Eq. (12) is beneficial for promoting the separated capability of the hypersphere to anomaly features. The hypersphere is trained using the error function in Eq. (9), then outputs these separated features. As such, the proposed model exhibits outstanding ability to identify anomalies from the input samples.

Model architecture

The proposed model, which consists of the probability deep neural network and the hypersphere (namely, DNNH), has three modules, including an encoding module, a hypersphere module and a decoding module, as shown in Fig. 1. For the encoding module in Fig. 1, there are two hidden layers. The ith hidden layer is denoted as \( H_i(e) \), and \( i = 1, 2 \). Given a inputting sample \( Z = \{z_1, z_2, \ldots, z_n\} \), \( Z \) is mapped onto the input layer in the encoding module. Then, \( H_i(e) \) captures the low-dimensional features \( F_s = \{F_a, F_n\} \), where \( F_a = \{f_{a_1}, f_{a_2}, \ldots, f_{a_u}\} \), \( F_n = \{f_{n_1}, f_{n_2}, \ldots, f_{n_v}\} \), and \( u + v = m \).

For the decoding module in Fig. 1, similar to the encoding module, there are two hidden layers. The jth hidden layer is denoted as \( H_j(d) \), and \( j = 1, 2 \). After receiving \( F_s \), \( H_j(d) \) reconstructs the input \( Z \). Finally, the output layer in the decoding module sends out the learned normal and anomaly classes.

Hyper parameter configuration

Regarding these hyper parameters in DNNS, we carefully studied part of them, e.g., optimizer, learning rate and activation function. Due to the other hyper parameters have no substantial effects on results, their default values are adopted.

Optimizer. Adam is used as the optimizer of DNNH. This reasons are that (i) Adam inherits the capability of AdamGrad to deal with sparse gradients [37]. (ii) As for Adam, the capability of handling sparse gradients is stronger than existing optimizers, such as RMSprop, SGD, Momentum and Nesterov etc. (iii) Adam can provide different learning rates based on different hyper parameters.

Learning rate. Using Adam as the optimizer of DNNH, there is no have to initialize the learning rate for DNNH.

Activation function. The function Sigmoid is considered to be used as the activation function. Compared to other activation functions, e.g., tanh, Relu and elu, the output of Sigmoid is only 0 and 1, so this is very suitable for judging anomalies and normal points.
In addition, we also need to consider the $B$ value in Eq. (5), so $B$ adopts the reference value in [32], i.e., let $B$ be equal to 1.

**Training and testing**

For the model training, we dynamically adjust the iteration epoch according the observed training precision, until DNNH can converge, the training is finished. While for the model testing, the testing set is used to verify the performance of DNNH.

**Experimental settings**

In “Training and testing”, experimental data sets are described, including 15 synthetic data sets and 6 real-world high-dimensional data sets. In “Experimental settings”, those compared approaches and their parameters are illustrated. Experimental assessment metrics are given in “Data sets”.

**Data sets**

We generated three type of synthetic data sets using the manner in [38], as shown in Fig. 2, each of type contains 5 sub data sets. For each sub data sets in the same type, data dimensionality gradually increases from 1000 to 5000. The first type in Fig. 2a, denoted as $T_1$, represents that normal data and anomalies are mixed by random manner. The second type in Fig. 2b, denoted as $T_2$, represents that anomalies locate outside normal data. The third type in Fig. 2c, denoted as $T_3$, represents that normal data surrounds anomalies. For the 15 synthetic data sets, we consider a few limited number of anomalies, i.e., anomaly ratio is equal to 3%. The detailed description of the 15 synthetic data sets is listed in Table 1 of Appendix A.

The six real-world data sets are adopted, whose data dimensions are greater than 1000 dimension. In addition, we also use two 2 benchmark data sets for the cross-verification in regard to the dividing data. Since the eight real-world high-dimensional data sets are usually used for classification tasks or clustering tasks, we converted them to anomaly detection data sets using the manner in [38]. Table 2 of Appendix A gives detailed description in regard to the eight real-world data sets.

**Competing methods and their parameters**

We selected the nine state-of-the-art approaches from five types of anomalous detection methods, i.e., (I) distance-based, KNN in [3]. (II) Reconstructed error-based, MF in [14]. (III) Classification-based, OC-SVM in [15]. (IV) Deep networks-based, DAE in [17], GANs in [18], RNNs in [22] (V) Hybrid architectures-based, DNN-K in [23], DNN–SVM in [24], DNN–SVDD in [25].

For the nine competitors, their optimal parameters observed in the corresponding literature were used. Unless
otherwise state, all experiments are run on the same experimental settings. In addition, those parameters that are not stated adopt default values.

In this work, receiver operating characteristic curve (ROC) and corresponding area under the curve (AUC) are commonly used to assess the accuracy of anomaly detection. In addition, mean square error (mse) and standard deviation (sd) are also applied to assess the detection results of methods. The calculation formulas of mse and sd are given in the following equation:

\[
\text{mse} = \frac{1}{D} \sum_{i=1}^{D} (y_i - p_i)^2 \\
\text{sd} = \sqrt{\frac{1}{D} \sum_{i=1}^{D} \left( p_i - \frac{1}{D} \sum_{i=1}^{D} p_i \right)^2},
\]

(15)

where \(y_i\) is the actual value, and \(p_i\) is the predicted value. \(D\) is the input data volume. To have the fair results, all experiments were run independently 100 times. Then, we analyze these results with statistical significance of the synthetic data set by of \(t\) test (\(p\) value < 0.05 for mse).

Results

In this section, all experimental results are presented, including mse, sd and detection accuracy, aiming at presenting the capabilities of DNNH for anomaly detection, and giving some insights respecting anomaly detection upon a high-dimensional space.

All results show that the detected performance of DNNH is significantly better than that of the nine competitors in considered cases. Moreover, these experimental results do not show in general a difference between the ability of anomaly detection upon a high-dimensional space for DNNH and the nine competitors.

Cross-validation

Due to the division of experimental data sets’ effects on the training accuracy of DNNH, the division testing on data sets needs to be considered so as to get the optimal configuration proportion for the training set and the testing set. Using benchmark data set B1, B2, we configured the following divided proportion for the training set and the testing set, respectively, i.e., training /testing set = 0.9/0.1, 0.8/0.2, 0.7/0.3, 0.6/0.4, 0.5/0.5.

The results are shown in Fig. 7 of Appendix B. When the proportion of training/testing set is equal to 0.8/0.2, DNNH gains the best performance, i.e., the AUC is 90.80% and 92.25% on the benchmark data set B1, B2, respectively. As such, all subsequent experiments adopt the value 0.8/0.2 to divide the training/testing set.

Experiments on synthetic data sets

The results on the two metrics of mse and sd show that DNNH is all lower than the nine competitors for the errors of mined anomalies, as shown in Table 3 of Appendix C. These statistical results in Table 3 of Appendix C indicates that there are no differences in general between DNNH and these competitors in the mining accuracy.

Mining accuracy

The results of the mined accuracy in Fig. 3 show that the capabilities of anomalous detection methods decrease as the dimensionality of input data increases. For different data distributions, i.e., on three type of data sets in Fig. 3a–c, DNNH maintains the high mining accuracy and also outperforms the nine competitors. In particular, when the dimensionality of the input data is equal to 5000, DNNH gains the accuracy of over 93%. However, the tradition methods are below 71% in the mined accuracy, e.g., the [3], the [14] and the [15]. While for deep methods and hybrid methods, they are below 85% in terms of accuracy. Together, these results implies that DNNH is not sensitive to data distribution, and DNNH has outstanding advantages for the high-dimensional anomaly detection.

Anomaly clustering

Figure 4 displays these visualized results of anomaly clustering on synthetic data set \(T1(5), T2(5)\) and \(T3(5)\), where these regions surrounded by a black line represent the results of anomaly clustering detected using DNNH and the nine competitors. Figure 4a shows that the quantity and quality of anomaly clustering detected using DNNH outperforms that of nine competitors. (Please observe the area surrounded by a black line). While for the traditional methods, such as OC-SVM in [15], MF in [14] and KNN in [3], the poorest detected results are obtained. Similarly, in Fig. 4b, c, DNNH not only gains these advanced results of the anomaly clustering, but also is better than the nine competitors. As such, for high-dimensional data presenting different distributions, DNNH outperforms these competitors in regard to the detected anomaly clustering.
Fig. 3  Mining performance on different the dimensionality of the input data

(a) $T_1$

(b) $T_2$

(c) $T_1$
Fig. 4 Clustering results. Anomalies are marked as red fork. Normal is marked as blue circle. Black line illustrates the clustered regions of the detected anomalies.
Fig. 5 Running time of method

(a) $T_1$

(b) $T_2$

(c) $T_3$
Figure 5 displays the execution time of DNNH and nine competitors. Results show that the execution time of the traditional methods, e.g., the methods in [3] in [14] and in [15], is lower than that of deep methods and hybrid methods on most data sets. While DNNH is lower than hybrid methods, such as the methods in [23], in [24] and the in [25].

During calculating Eq. (9), DNNH needs to do multiple iterations. The number of iterations $T$ relies on the input data dimension $D$ and data volume $V$, i.e., $T = I_1D + I_2V$, where $I_1$ and $I_2$ are a constant. In the case of fixed data volume, $T$ mainly depends on data dimension and increases as data dimension increases. Hence, the time complexity of DNNH is $O(n) = I_1D + I_2V$, i.e., $O(n)\approx O(n^2) < < O(n^3)$. 

Experiments on real-world data sets

The results on real-world data sets show that the mined accuracy of DNNH outperforms nine competitors, as shown in Fig. 6. Especially, on the ultra-high dimensional data set R1 (data dimension is equal to 10,000), DNNH reaches above 71% of the mined accuracy, while the traditional methods, e.g., KNN [3], MF [14] and OC-SVM [15], almost fail on data set R1, i.e., their mined accuracy is less than 20%. Obviously, hybrid approaches and deep approaches are superior to traditional methods in terms of the mine performance. This is because deep architectures can capture low-dimensional features from high-dimensional data, so as to decrease the dimensionality of the input data.

According to experimental results on synthetic data sets and real-world data sets, several observations can be obtained from Figs. 3, 4, 5 and 6.

\begin{enumerate}
\item High dimensionality of the input data increases the complexity of data space, through fusing the probability method of sample binary-classification into deep neural networks, the hidden layers can extract these low-dimensional layered features of being used to distinguish anomalies from normal instance.
\item If the extracted features can represent fewer anomaly instances, this enough identify anomalies from normal classes.
\item On a high-dimensional space, hybrid methods consisting of deep networks and traditional methods show excellent the mined capabilities of anomalies.
\item The time complexity of deep detection algorithms includes the depth of network architectures and the dimensionality of input data. Usually, the data lying in a high-dimensional space is prone to a deeper network architecture to learn more meaningful features, so that the computational complexity of deep detection algorithms increases along with the number of network layers. While the computational complexity of hybrid algorithms contains the complexity of deep detection algorithms and traditional detection algorithms.
\end{enumerate}
Discussion

DNNH has outstanding the mined capability for high-dimensional anomalies, so we give a detailed explanation as following. First, Eq. (8) ensures the layered characteristic of the captured low-dimensional anomaly features and low-dimensional normal features. Equation (5) reduces the probability that anomalous points are classified into the \( C_2 \) classes (i.e., normal classes) during mining anomalies. Then, the kernel in Eq. (12) achieves the linear separation of the two types of captured low-dimensional features. Moreover, through learning the error function \( \Delta(a, r) \) in Eq. (9), the hypersphere allows that the captured low-dimensional features are far away from the center \( a \) of the sphere. Finally, through learning the final learning function \( \nabla(L(w, b), \Delta(a, r)) \) in Eq. (14), DNNH gains these advanced results of anomaly mining upon a high-dimensional space.

Conclusion

In this work, a hybrid method is proposed for anomaly mining upon a high-dimensional space. In the proposed method, the probability deep neural network first captures the low-dimensional features from the background space. Then, the captured low-dimensional features are separated using the hypersphere, so as to realize the distinction between anomaly and normal classes. Experimental results show that the proposed method outperforms the advanced anomalous detection methods in mining ability. We demonstrate that deep neural networks can capture these desired low-dimensional layered features through fusing the probability method of sample multi-classification. Moreover, the extracted features that represent a fewer anomaly instances can sufficiently identify anomalies from normal instances. In future work, we will look at exploring anomaly detection methods targeted to irrelevant attributes interference on a high-dimensional space, i.e., how to identify the anomalies masked by irrelevant attributes on a high-dimensional space.

Funding

The research funding is Supported by the Science and Technology Research Program of Chongqing Municipal Education Commission of China under Grant KJQN201903003. And the Science and Technology Research Program of Chongqing Municipal Education Commission of China under Grant KJQN202003001.

Declarations

Conflict of interest All authors have no conflicts of interest to declare that are relevant to the content of this article.

Ethics approval and consent to participate All authors declare that this work does not include humans and animals, as well as never collects data from human subjects.

Consent for publication All authors agree with availability of data and materials.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.
Appendices

Appendix A: Data set description

See Tables 1, 2.

Table 1 Description of the synthetic data sets

| Data set type | Serial number | Anomaly ratio (%) | Data dimensionality | Data volume |
|---------------|---------------|-------------------|---------------------|------------|
| T1            | T1(1)         | 3                 | 1000                | 800        |
|               | T1(2)         | 3                 | 2000                | 800        |
|               | T1(3)         | 3                 | 3000                | 800        |
|               | T1(4)         | 3                 | 4000                | 800        |
|               | T1(5)         | 3                 | 5000                | 800        |
| T2            | T2(1)         | 3                 | 1000                | 800        |
|               | T2(2)         | 3                 | 2000                | 800        |
|               | T2(3)         | 3                 | 3000                | 800        |
|               | T2(4)         | 3                 | 4000                | 800        |
|               | T2(5)         | 3                 | 5000                | 800        |
| T3            | T3(1)         | 3                 | 1000                | 800        |
|               | T3(2)         | 3                 | 2000                | 800        |
|               | T3(3)         | 3                 | 3000                | 800        |
|               | T3(4)         | 3                 | 4000                | 800        |
|               | T3(5)         | 3                 | 5000                | 800        |

Table 2 Description of real-world data sets

| Serial number | Benchmark data sets | Description (normal vs. outliers) | Number | Outlier ratio (%) | Data dimension |
|---------------|---------------------|-----------------------------------|--------|-------------------|----------------|
|               |                     | Normal                           | Anomaly|                   |                |
| B1            | Shuttle             | Class ‘1’ vs. Others             | 1000   | 13                | 1.28           | 9              |
| B2            | Waveform            | Others vs. Class ‘0’             | 3343   | 100               | 2.9            | 21             |

| Serial number | High-dimensional data sets | Description (normal vs. outliers) | Number | Outlier ratio (%) | Data dimension |
|---------------|----------------------------|-----------------------------------|--------|-------------------|----------------|
|               |                            | Normal                           | Anomaly|                   |                |
| R1            | Arcene                     | Normal patterns vs. cancer       | 8,459,427 | 540,573   | 6.39           | 10,000         |
| R2            | p53Mutant                  | Inactive vs. active              | 16,449 | 143              | 0.86           | 5408           |
| R3            | Gisette                    | Zero and non-zero values         | 28,278,760 | 4,221,240 | 12.98          | 5000           |
| R4            | InternetAds                | Ads vs. other images             | 3264   | 454              | 12.21          | 1555           |
| R5            | Micro mass                 | Zero and non-zero values         | 464,819 | 3181       | 0.68           | 1300           |
| R6            | Malware                    | Zero and non-zero values         | 2,894,954 | 37,772    | 0.12%          | 1087           |

Appendix B: Cross-validation on benchmark data sets

See Fig. 7.
Appendix C: Statistical results on synthetic data sets

See Table 3.

Fig. 7 Verified results on benchmark data set B1 and B2
### Table 3: Results of mse and sd on synthetic data sets

| Data set | Hybrid architectures-based | Deep networks-based | Distance-based | Reconstructed-based | Classification-based |
|----------|---------------------------|---------------------|---------------|---------------------|----------------------|
|          | Our method (DNNH) | Method in [23] (DNN-K) | Method in [24] (DNN–SVDD) | Method in [25] (DNN–SVDD) | Method in [17] (DAE) | Method in [18] (GNNs) | Method in [22] (RNNs) | Method in [3] (KNN) | Method in [14] (MF) | Method in [15] (OC-SVM) |
| T(1)     | 2.26e − 5 | 1.12e − 4 | 1.16e − 4 | 1.16e − 4 [5.28e] | 1.24e − 4 | 1.26e − 4 | 1.28e − 4 | 1.30e − 4 [0.90e] | 1.35e − 4 [8.48e] | 1.31e − 4 [4.49e] |
|          | {1.86e − 5} | [0.43e] | [7.51e − 4] | [0.0019]* | [3.58e − 4] | [7.62e − 4] | [6.21e − 4] | [4.0178]* | [4.0354]* | [4.0364]* |
|          | [0.0460]* | [0.0028]* | | | [0.0062]* | [0.0169]* | [0.0109]* | | | |
| T(2)     | 2.56e − 5 | 1.40e − 4 | 1.38e − 4 | 1.45e − 4 [6.56e] | 1.50e − 4 | 1.60e − 4 | 1.66e − 4 | 1.70e − 4 [7.93e] | 1.65e − 4 [3.77e] | 1.64e − 4 [4.30e] |
|          | {1.47e − 5} | [2.22e] | [5.31e − 4] | [0.0013]* | [7.28e − 4] | [1.28e − 4] | [3.28e − 4] | [4.0224]* | [4.0044]* | [4.0047]* |
|          | [0.0020]* | [0.0088]* | | | [0.0223]* | [0.0185]* | [0.0381]* | | | |
| T(3)     | 3.28e − 5 | 1.81e − 4 | 1.84e − 4 | 1.72e − 4 [7.12e] | 1.91e − 4 | 1.85e − 4 | 1.88e − 4 | 1.92e − 4 [7.60e] | 1.92e − 4 [4.51e] | 1.99e − 4 [6.13e] |
|          | {3.01e − 5} | [4.96e] | [6.78e − 4] | [0.0030]* | [2.79e − 4] | [6.46e − 4] | [5.77e − 4] | [4.0217]* | [4.0284]* | [4.0060]* |
|          | [0.0123]* | [0.0014]* | | | [0.0239]* | [0.0073]* | [0.0283]* | | | |
| T(4)     | 3.34e − 5 | 2.09e − 4 | 2.02e − 4 | 2.05e − 4 [1.23e] | 2.33e − 4 | 2.10e − 4 | 2.23e − 4 | 2.71e − 4 [0.87e] | 2.64e − 4 [0.64e] | 2.54e − 4 [1.01e] |
|          | {1.65e − 5} | [2.21e] | [2.92e − 4] | [0.0016]* | [1.96e − 4] | [0.21e − 4] | [0.33e − 4] | [0.0033]* | [0.0072]* | [0.0360]* |
|          | [0.0149]* | [0.0019]* | | | [0.0080]* | [0.0360]* | [0.0060]* | | | |
| T(5)     | 3.43e − 5 | 2.53e − 4 | 2.53e − 4 | 2.88e − 4 [5.77e] | 2.94e − 4 | 2.76e − 4 | 2.56e − 4 | 2.94e − 4 [6.72e] | 2.98e − 4 [8.64e] | 2.76e − 4 [1.48e] |
|          | {1.79e − 5} | [3.17e] | [3.86e − 4] | [0.0010]* | [1.45e − 4] | [0.23e − 4] | [0.23e − 4] | [0.0079]* | [0.0202]* | [0.0365]* |
|          | [0.0097]* | [0.0015]* | | | [0.0169]* | [0.0073]* | [0.0283]* | | | |
| T(2)     | 1.17e − 5 | 1.16e − 4 | 1.12e − 4 | 1.16e − 4 [6.95e] | 1.26e − 4 | 1.24e − 4 | 1.21e − 4 | 1.30e − 4 [5.65e] | 1.31e − 4 [1.24e] | 1.16e − 4 [4.75e] |
|          | {1.56e − 5} | [6.47e] | [4.63e − 4] | [0.0018]* | [4.93e − 4] | [1.12e − 4] | [1.07e − 4] | [4.0049]* | [4.0061]* | [4.0030]* |
|          | [0.0013]* | [0.0012]* | | | [0.0125]* | [0.0389]* | [0.0093]* | | | |
| T(2)     | 1.49e − 5 | 1.45e − 4 | 1.38e − 4 | 1.35e − 4 [3.15e] | 1.60e − 4 | 1.40e − 4 | 1.33e − 4 | 1.64e − 4 [4.20e] | 1.65e − 4 [3.12e] | 1.50e − 4 [3.57e] |
|          | {1.66e − 5} | [1.78e] | [1.73e − 4] | [0.0011]* | [4.95e − 4] | [1.99e − 4] | [1.22e − 4] | [0.0281]* | [0.0126]* | [0.0197]* |
|          | [0.0047]* | [0.0013]* | | | [0.0190]* | [0.0036]* | [0.0062]* | | | |
| T(3)     | 1.77e − 5 | 1.70e − 4 | 1.81e − 4 | 1.72e − 4 [3.79e] | 1.84e − 4 | 1.85e − 4 | 1.83e − 4 | 1.92e − 4 [6.25e] | 1.91e − 4 [4.50e] | 1.92e − 4 [1.01e] |
|          | {1.84e − 5} | [0.56e] | [0.90e − 4] | [0.0088]* | [5.44e − 4] | [7.38e − 4] | [5.88e − 4] | [0.0118]* | [0.0345]* | [0.0131]* |
|          | [0.0098]* | [0.0019]* | | | [0.0099]* | [0.0241]* | [0.0084]* | | | |
| T(4)     | 1.85e − 5 | 1.92e − 4 | 1.99e − 4 | 2.02e − 4 [7.50e] | 2.09e − 4 | 2.05e − 4 | 2.09e − 4 | 2.10e − 4 [5.36e] | 2.41e − 4 [3.46e] | 2.33e − 4 [6.26e] |
|          | {2.28e − 5} | [0.31e] | [4.08e − 4] | [0.0063]* | [0.13e − 4] | [4.00e − 4] | [5.11e − 4] | [0.0332]* | [0.0392]* | [0.0238]* |
|          | [0.0043]* | [0.0011]* | | | [0.0339]* | [0.0216]* | [0.0288]* | | | |
Table 3 (continued)

| Data set | Hybrid architectures-based | Deep networks-based | Distance-based | Reconstructed-based | Classification-based |
|----------|-----------------------------|---------------------|----------------|---------------------|---------------------|
|          | Our method (DNNH) | Method in [23] (DNN-K) | Method in [24] (DNN−SVDD) | Method in [25] (DNN−SVDD) | Method in [17] (DAE) | Method in [18] (GNNs) | Method in [22] (RNNs) | Method in [3] (KNN) | Method in [14] (MF) | Method in [15] (OC-SVM) |
|          |                 |                     |                     |                     |                     |                     |                     |                     |                     |                     |
| **T2(5)** | 1.96e−5 {2.75e−5} [0.0016] | 2.71e−4 {2.48e−5} [0.0087] | 2.54e−4 {7.27e−4} [0.0079] | 2.64e−4 {5.77e−4} [0.0096] | 2.53e−4 {3.44e−4} [0.0133] | 2.54e−4 {7.26e−4} [0.0205] | 2.67e−4 {7.88e−4} [0.0185] | 2.94e−4 {4.45e−4} [0.0275] | 2.88e−4 {7.93e−4} [0.0344] | 2.98e−4 {2.80e−4} [0.0017] |
|          | 1.13e−4 {8.18e−5} [0.0093] | 1.08e−4 {6.63e−5} [0.0005] | 1.14e−4 {6.65e−5} [0.0007] | 1.17e−4 {1.37e−4} [0.0008] | 1.09e−4 {1.36e−4} [0.0111] | 1.10e−4 {1.22e−4} [0.0115] | 1.11e−4 {1.12e−4} [0.0111] | 1.28e−4 {1.81e−4} [0.0045] | 1.31e−4 {6.13e−4} [0.0042] | 1.26e−4 {5.36e−4} [0.0045] |
|          | 1.42e−4 {2.75e−5} [0.0118] | 1.49e−4 {2.53e−5} [0.0067] | 1.56e−4 {3.47e−5} [0.0024] | 1.64e−4 {5.17e−4} [0.0076] | 1.70e−4 {2.59e−4} [0.0078] | 1.60e−4 {2.22e−4} [0.0118] | 1.73e−4 {7.52e−4} [0.0145] | 1.75e−4 {7.22e−4} [0.0044] | 1.80e−4 {6.32e−4} [0.0091] | 1.80e−4 {6.32e−4} [0.0091] |
|          | 1.80e−4 {6.67e−5} [0.018] | 1.82e−4 {5.43e−5} [0.0080] | 1.75e−4 {5.50e−5} [0.0124] | 1.85e−4 {3.69e−5} [0.0173] | 1.89e−4 {4.85e−5} [0.0097] | 1.88e−4 {4.15e−5} [0.0118] | 1.96e−4 {4.99e−5} [0.0137] | 1.91e−4 {6.04e−5} [0.0231] | 1.97e−4 {1.85e−5} [0.0231] | 1.97e−4 {1.85e−5} [0.0231] |
|          | 1.98e−4 {3.75e−5} [0.0172] | 2.12e−4 {3.72e−5} [0.0075] | 2.09e−4 {5.36e−5} [0.0037] | 2.31e−4 {4.09e−5} [0.0079] | 2.15e−4 {4.70e−5} [0.0097] | 2.11e−4 {4.53e−5} [0.0411] | 2.39e−4 {2.86e−5} [0.0260] | 2.42e−4 {4.81e−5} [0.0249] | 2.56e−4 {7.75e−5} [0.0150] | 2.56e−4 {7.75e−5} [0.0150] |
|          | 2.80e−5 {6.15e−5} [0.0197] | 2.76e−5 {2.01e−5} [0.0017] | 2.67e−5 {1.74e−5} [0.0045] | 2.85e−5 {3.71e−5} [0.0089] | 2.95e−5 {2.80e−5} [0.0072] | 2.75e−5 {1.80e−5} [0.0002] | 3.48e−4 {2.72e−5} [0.0027] | 3.03e−4 {5.88e−5} [0.0267] | 3.14e−4 {7.94e−5} [0.0234] | 3.14e−4 {7.94e−5} [0.0234] |

The mse for t test (p < 0.05). The experiments were carried out independently 100 times. mse, sd. [Significant at p values for t test]
References

1. Kui Yu, Chen H (2019) Markov Boundary-Based Outlier Mining. IEEE Transactions on neural networks and learning systems 30(4):1259–1264
2. Menon V, Kalyani S (2019) Structured and unstructured outlier identification for robust PCA: a fast parameter free algorithm. IEEE Trans Signal Process 67(9):2439–2452
3. Salehi M, Leckie C, Bezdek JC, Vaithianathan T, Zhang X (2016) Fast memory efficient local outlier detection in data streams. IEEE Trans Knowl Data Eng 28:3246–3260
4. Chehreghani MH (2016) K-nearest neighbor search and outlier detection via minimax distances. In: SIAM International Conference on Data Mining, pp 405–413
5. Li X, Lv J, Yi Z (2020) Outlier detection using structural scores in a high-dimensional space. IEEE Trans Cybernet 50(5):2302–2310
6. Gupta M, Gao J, Aggarwal CC, Han J (2014) Outlier detection for temporal data: a survey. IEEE Trans Knowl Data Eng 26(9):2250–2267
7. Radovanovic M, Nanopoulos A, Ivanovic M (2015) Reverse nearest neighbors in unsupervised distance-based outlier detection. IEEE Trans Knowl Data Eng 27(5):1369–1382
8. Cao N, Lin C, Guhan Zhu Y-R, Lin XT, Wen X (2018) Voila: Visual Anomaly Detection and Monitoring with Streaming Spatiotemporal Data. IEEE Trans Visual Comput Graphics 24(1):23–33
9. Soleimani H, Miller DJ (2016) Atd, “Anomalous topic discovery in high dimensional discrete data.” IEEE Trans Knowl Data Eng 28(9):2267–2280
10. Mao J, Wang T, Jin C, Zhou A (2017) Feature grouping-based outlier detection upon streaming trajectories. IEEE Trans Knowl Data Eng 29(12):2696–2709
11. Manzoor E, Milajerdi SM and Akoglu L (2016) Reverse nearest neighbors in unsupervised distance-based outlier detection. IEEE Trans Knowl Data Eng 27(5):1369–1382
12. Reilly C, Ghiyshak A, Imran MA (2016) Distributed anomaly detection using minimum volume elliptical principal component analysis. IEEE Trans Knowl Data Eng 28:2320–2333
13. He X, Zhang H, Kan MY and Chua TS (2016) Fast matrix factorization for online recommendation with implicit feedback. In: ACM SIGIR Special Interest Group on Information Retrieval, pp 549–558
14. Erfani SM, Rajasegarar S, Karunasekera S, Leckie C (2016) High-dimensional and large-scale anomaly detection using a linear one-class svmm with deep learning. Pattern Recogn 58:121–134
15. Ergen T, Mirza AH and Kozat SS (2017) Unsupervised and semi-supervised anomaly detection with lstm neural networks. arXiv arXiv:1710.09207
16. Berg C, Christensen IPR, Ressell P (1984) Harmonic analysis on semigroups. Springer, Berlin
17. Zhou C, Paffenroth RC (2017) Anomaly detection with robust deep autoencoders. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp 665–674
18. Li D, Chen D, Shi L, Jin B, Goh J and Ng S-K (2019) MADGAN: multivariate anomaly detection for time series data with generative adversarial networks. In: International Conference on Artificial Neural Networks (ICANN) 2019: Artificial Neural Networks and Machine Learning – ICANN 2019: Text and Time Series, pp 703–716
19. Le Cun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521:436–444
20. Lusch B, Kutz JN, Brunton SL (2018) Deep learning for universal linear embeddings of nonlinear dynamics. Nat Commun 9:1–10
21. Creswell A, White T, Dumoulin V, Arulkumaran K, Sengupta B, Bharath AA (2018) Generative adversarial networks: An overview. IEEE Signal Process Magazine 35(1):53–65
22. Goh J, Adepu S, Tan M, et al (2017) Anomaly detection in cyber physical systems using recurrent neural networks” [C]. In: 2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE), pp 140–145, IEEE
23. Bengio Y and LeCun Y (2007) Scaling learning algorithms towards AI. In: Bottou L, et al (eds) Large scale kernel machines, pp 1–41
24. Inoue J, Yamagata Y, Chen Y, Poskitt CM and Sun J (2017) Anomaly detection for a water treatment system using unsupervised machine learning. In: Data Mining Workshops (ICDMW), 2017 IEEE International Conference on, pp 1058–1065
25. Kim S, Choi Y, Lee M (2015) Deep learning with support vector data description. Neurocomputing 165:1–17
26. Saxe AM, Koh PW, Chen Z, Bhand M and NgAY (2011) On random weights and unsupervised feature learning. In: Machine Learning, 2011 IEEE International Conference on, pp 1089–1096
27. Peng X, Shen J (2017) A twin-hyperspheres support vector machine with automatic variable weights for data classification. Inf Sci 417:216–235
28. Peng X, Dong Xu (2015) A twin-hypersphere support vector machine classifier and the fast learning algorithm. Inf Sci 221:12–27
29. Araque O, Corcuera-Platas I, Sánchez-Rada JA, Iglesias CA (2017) Enhancing deep learning sentiment analysis with ensemble techniques in social applications. Expert Syst Appl 77:236–246
30. Qinglin Wu, Coggins BE, Zhou P (2016) Unbiased measurements of reconstruction fidelity of sparsely sampled magnetic resonance spectra. Nat Commun 7:1–8
31. Zhang R, Li J, Shaoyu Wu, Meng D (2016) Learning to select supplier portfolios for service supply chain. PLoS One 11(5):1–19
32. Titterington D (1981) Comparison of discriminant techniques applied to a complex data set of head injured patients. J R Stat Soc 144(2):145–175
33. Olshausen BA, Field DJ (1997) Sparse coding with an overcomplete basis set: a strategy employed by v1. Vis Res 37:3311–3325
34. Snoek J, Larochelle H and Adams RP (2012) Practical bayesian optimization of machine learning algorithms. In: Advances in neural information processing systems, pp 2951–2959
35. Oh CY, Gavves E and Welling M (2019) BOCK: Bayesian optimization with cylindrical kernels, arXiv:1806.01619v2
36. Jayasumana S, Hartley R, Salzmann M et al (2014) Optimizing over radial kernels on compact manifolds [C]. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 3802–3809
37. Kingma DP and Ba JL (2015) Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980v8
38. Campos GO, Zimek A, Sander J (2016) On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. Data Min Knowl Disc 30:891–927