electronics

MDPI

Article

MGAD: Mutual Information and Graph Embedding Based Anomaly Detection in Multivariate Time Series

Yuehua Huang [1,2,*], Wenfen Liu [1,2], Song Li [1], Ying Guo [1] and Wen Chen [1]

[1] School of Computer Science and Information Security & School of Software Engineering, Guilin University of Electronic Technology, Guilin 541004, China
[2] Guangxi Key Laboratory of Cryptography and Information Security, Guilin 541004, China
* Correspondence: 1802102009@mails.guet.edu.cn

Abstract: Along with the popularity of mobile Internet and smart applications, more and more high-dimensional sensor data have appeared, and these high-dimensional sensor data have hidden information about system performance degradation, system failure, etc., and how to mine them to obtain such information is a very difficult problem. This challenge can be solved by anomaly detection techniques, which is an important field of research in data mining, especially in the domains of network security, credit card fraud detection, industrial fault identification, etc. However, there are many difficulties in anomaly detection in multivariate time-series data, including poor accuracy, fast data generation, lack of labeled data, and how to capture information between sensors. To address these issues, we present a mutual information and graph embedding based anomaly detection algorithm in multivariate time series, called MGAD (mutual information and graph embedding based anomaly detection). The MGAD algorithm consists of four steps: (1) Embedding of sensor data, where heterogeneous sensor data become different vectors in the same vector space; (2) Constructing a relationship graph between sensors using their mutual information about each other; (3) Learning the relationship graph between sensors using a graph attention mechanism, to predict the sensor data at the next moment; (4) Compare the predicted values with the real sensor data to detect potential outliers. Our contributions are as follows: (1) we propose an unsupervised outlier detection called MGAD with a high interpretability and accuracy; (2) massive experiments on benchmark datasets have demonstrated the superior performance of the MGAD algorithm, compared with state-of-the-art baselines in terms of ROC, F1, and AP.

Keywords: mutual information; graph embedding; anomaly detection; multivariate time series

Citation: Huang, Y.; Liu, W.; Li, S.; Guo, Y.; Chen, W. MGAD: Mutual Information and Graph Embedding Based Anomaly Detection in Multivariate Time Series. *Electronics* **2024**, *13*, 1326. https://doi.org/10.3390/electronics13071326

Academic Editor: Stefanos Kollias

Received: 19 February 2024
Revised: 26 March 2024
Accepted: 30 March 2024
Published: 1 April 2024

1. Introduction

As mobile Internet usage grows and IoT applications expand, more sensors are being incorporated into industrial systems, data centers, automobiles, and other infrastructure. Continuous monitoring and control of the devices and sensors is crucial for the upkeep and operation of Internet of Things applications to safeguard the regular functioning of the devices or applications. This is particularly true for vital infrastructure such as power grids, water supply systems, piped gas, heating, etc.

For instance, heating, ventilation, and air conditioning (HVAC) systems result in considerable energy waste and production-related energy consumption. A report indicates that the HVAC system is quickly overtaking other building service systems in terms of energy use. Consequently, it has been difficult to quickly and precisely identify anomalous HVAC system functioning patterns resulting in energy squandering.

Various sensors may be found in each of the various components of a water treatment plant, monitoring things like water level, flow rates, water quality, valve condition, and more. Complex, nonlinear relationships may be formed between the data from various sensors. For instance, opening a valve might alter the pressure and flow rate, which can then trigger other changes as automated systems react to the altered conditions.

A wireless sensor network's reliance on manual monitoring is unrealistic due to the excessive number of sensors and the increasingly complex relationships between them. Instead, anomaly detection, a type of data mining technology, must be applied to the sensor-generated data series to identify potential anomalies in time, perform necessary equipment overhauls, and ultimately guarantee the sensor network's normal operation.

The process of identifying the outliers from normal values is called anomaly detection, sometimes referred to as outlier detection or novelty detection. According to Hawking, "an outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism" [1].

Outlier detection has been an important field of research of concern to industry and academia. By identifying outliers, researchers can obtain vital knowledge that assists in making better decisions or avoiding risks. Thus, outlier detection is widely used in many fields, such as fraud detection [2–6], intelligent transportation [7–10], video content analysis and detection [11–13], network intrusion detection and IoT security [14–20], data generation [21,22], and social media analysis [23–26].

The main contributions of our study are listed below.

(1) We propose an unsupervised outlier detection called MGAD with a high interpretability and accuracy. It innovatively combines the information of the sensors themselves with the mutual information between the sensors, which avoids the problem that previous multivariate time series anomaly detection for sensors either only considers the information of the sensors themselves or the information between the sensors, and thus improves the accuracy of the algorithm;
(2) Massive experiments on benchmark datasets have demonstrated the superior performance of the MGAD algorithm, compared with state-of-the-art baselines in terms of ROC, F1, and AP;
(3) In this paper, mutual information is used to assess the closeness of the relationship between sensors rather than utilizing correlation coefficients, Euclidean distances, or cosine distances, etc., which will give academics something useful to explore.

The rest of this paper is organized as follows. Section 2 discusses the related work of this paper. In Section 3, we explain our anomaly detection algorithm in detail. The experiments and results are discussed in Section 4. Finally, Section 5 presents the findings, contributions, limitations, and future work.

2. Related Work

2.1. Anomaly Detection in Multivariate Time Series

The goal of anomaly detection research [27–29], which has been extensively explored across several domains (such as network security, time series, graphs, etc.), is to identify the situations in which one observation substantially deviates from the other observations in the same dataset.

Our initial research presented ISOD (interpretable single-dimension outlier detection), an unsupervised outlier detection technique with good interpretability and scalability that is based on quantiles and skewness coefficients [30]. However, ISOD targets static data and is not very applicable to multivariate time series data.

In multivariate time series anomaly detection, long short-term memory and autoencoders (LSTM-AEs) have been widely employed [31–33]. These approaches' primary goal is to extract robust features from time series based on long short-term memory (LSTM) to learn typical patterns. Subsequently, an encoder and decoder are used to recreate the data using the acquired normal patterns. The divergence between the reconstructed and original data allows for the detection of anomalous patterns. LSTM-NDT [34] uses LSTM for multivariate time series prediction and then uses the prediction error to identify spacecraft anomalies. Using an LSTM-based encoder–decoder model, EncDnc-AE [35] reconstructs normal data and uses reconstruction errors to identify anomalies to extract latent patterns from multi-sensor time series.

These LSTM-based anomaly detection algorithms have some drawbacks. First of all, its calibration might be difficult and time-consuming as it needs a lot of data inputs and processing resources. Second, it can be hard to maximize the performance of LSTMs since they include a lot of parameters that are frequently hard to limit or check. Thirdly, although LSTMs can preserve the sequential temporal sequence of inputs, they could have trouble picking up long-term dependencies.

2.2. Graph Embedding

Graph mining has gained popularity because graphs may represent intricate data structures. Graph embedding is a fundamental technique in graph mining that maps a graph into a vector space, where each graph is represented by an embedding vector, to learn the latent representation of a graph. Grover [36] proposed Node2vec, which is based on the random walk. Using a random walk, Node2vec first converts the graph structure into a sentence structure. Every graph in the sentence corresponds to a word. Next, the skip-gram is used to embed the word. Perozzi et al. [37] suggested using a random walk in the graph embedding method to mine the hidden representation and retrieve local information. Furthermore, graph embedding has been used for several tasks. For instance, object tracking can be implemented by combining the Bayesian inference framework with graph embedding [38].

Its efficacy in modeling multi-relational data in recommender systems and natural language processing has also been confirmed [39]. On the other hand, the time series data are continuously changing, while the inputs to these models are static graphs. In order to overcome this difficulty, a graph representation learning technique has been developed that embeds the dynamic graph. Yuxuan Gu et al. used graph embedding techniques for anomaly detection in HVAC systems [40].

These graph embedding-based anomaly detection algorithms, while preserving potential relationships between nodes, do not have a way to capture long-term dependencies between nodes' historical data, which is really what LSTMs are good at.

Therefore, this paper tries to combine an LSTM with graph embedding, which on one hand can extract the complex potential relationships between nodes, and on the other hand can also mine the dependencies between nodes' historical data.

2.3. Graph Neural Network

Graph neural networks (GNNs) are widely used in research to identify anomalies in multivariate time data [41]. In order to discover anomalous periods through prediction and reconstruction, gated recurrent units are utilized to record patterns in the time series, while graph attention networks are utilized to learn correlations across multivariate time series [42]. The LUNAR anomaly detection technique, which is based on GNNs, was proposed by Goodge et al. [43]. LUNAR aggregates vertex domain information to detect abnormalities and adds learnability to anomaly detection using GNNs.

The impact of correlation between time series is not taken into account in the aforementioned anomaly detection techniques, which learn normal patterns using LSTM and reconstruct them to calculate residuals. To collect the domain information of vertices and disregard the global information of the graph, approaches based on GNNs take into account attention mechanisms and convolution.

R-GCN [44] is a type of graph neural network (GNN) that can deal with structured data from time series and is proposed to analyze transactions in a blockchain-based platform using the stochastic gradient boosting (SGB) technique.

A novel detection and multi-classification vision-based approach for IoT-malware is proposed [45]. Rather than starting from scratch with training models, this strategy leverages the advantages of the deep transfer learning methodology and combines several ensembling strategies and a fine-tuning method to improve detection and classification performance.

GNNs can perform relatively good feature extraction on data in Euclidean space, as CNNs have performed with success in image recognition. But the relationships between nodes in graph data, which are in non-Euclidean space, are far more complex than the relationships between pixel points of an image. Therefore, the GCN technique, a graph neural network, has been invented to mine the complex relationships between images. This is an important technical basis of this paper.

3. Proposed Algorithm

3.1. Problem Statement

In our study, the training data consist of sensor data (a kind of multivariate time series) from N sensors over T_{train} time ticks. The training sensors' data are indicated as $S = [s_1, s_2, \cdots, s_N]_{N \times T_{train}} \in R^{N \times T_{train}}$ which is used to train our proposed algorithm. At each time t, the sensor data are $s_t = \{s_{1t}, s_{2t}, \cdots s_{Nt}\}$. Similarly, the test dataset is denoted as being derived from N sensors over T_{test} time ticks. This algorithm finally outputs a binary value: if it is 1, it means that there is an anomaly at that time tick; if it is 0, that moment is normal.

In the context of data mining, unsupervised algorithms generally train the algorithms using a training dataset that consists entirely of normal data, so we default to a dataset S that is entirely normal data. Correspondingly, the test dataset T will contain a small amount of abnormal data. Our goal in training the algorithm using the training dataset is to find as many potentially abnormal data as possible in the test dataset.

3.2. Algorithm Overview

The MGAD algorithm consists of four steps:

(1) Embedding of sensor data, where heterogeneous sensor data become different vectors in the same vector space; it employs embedding vectors to obtain the unique characteristics of each sensor.
(2) Constructing a relationship graph between sensors using their mutual information about each other;
(3) Learning the relationship graph between sensors using a graph attention mechanism, to predict the sensor data at the next moment;
(4) Comparing the predicted values with the real sensor data to detect potential outliers.

The above four steps are shown in Figure 1. In Section 4, we compared the performance of the MGAD algorithm with eight other benchmark algorithms on four public datasets, using F1, AP, and ROC for performance evaluation. We use 70% of the dataset for training and 30% for testing.

3.3. Sensors Embedding

Distinct sensors can have highly distinct properties in numerous sensor data contexts, and these differences might be intricately tied to one another. In agricultural IoT applications, for example, it is often necessary to take temperature and humidity data. Temperature and humidity at the same location are strongly correlated with each other, while comparing temperature data from different locations often does not make much sense.

Therefore, we would like to use a flexible approach to capture potential relationships in multivariate sensor data. In addition, different sensors do not have the same range of values, discrete or continuous values, and units, so we will use sensor embedding techniques to turn each sensor's data into a vector in the same vector space to represent their relationships with each other.

$$v_i \in R^d, i \in \{1, 2, \cdots N\} \tag{1}$$

In the experimental part of this paper, we will use Deepwalk [37] for graph embedding. After being randomly initialized, these embedding vectors are trained with the rest of the

model, as shown in Equation (1). In Equation (1), v_i is the vector obtained after embedding and d is the dimension of the vector.

These embedding vectors will be used in two ways in our algorithm MGAD: (1) to learn the relationships between sensors in our structure, and (2) to execute attention over neighbors in our attention mechanism in a way that supports heterogeneous effects for various kinds of sensors.

As a result, sensors with similar embedding values should be highly likely to be associated with one another. The similarity between these embeddings thus suggests a similarity of behaviors.

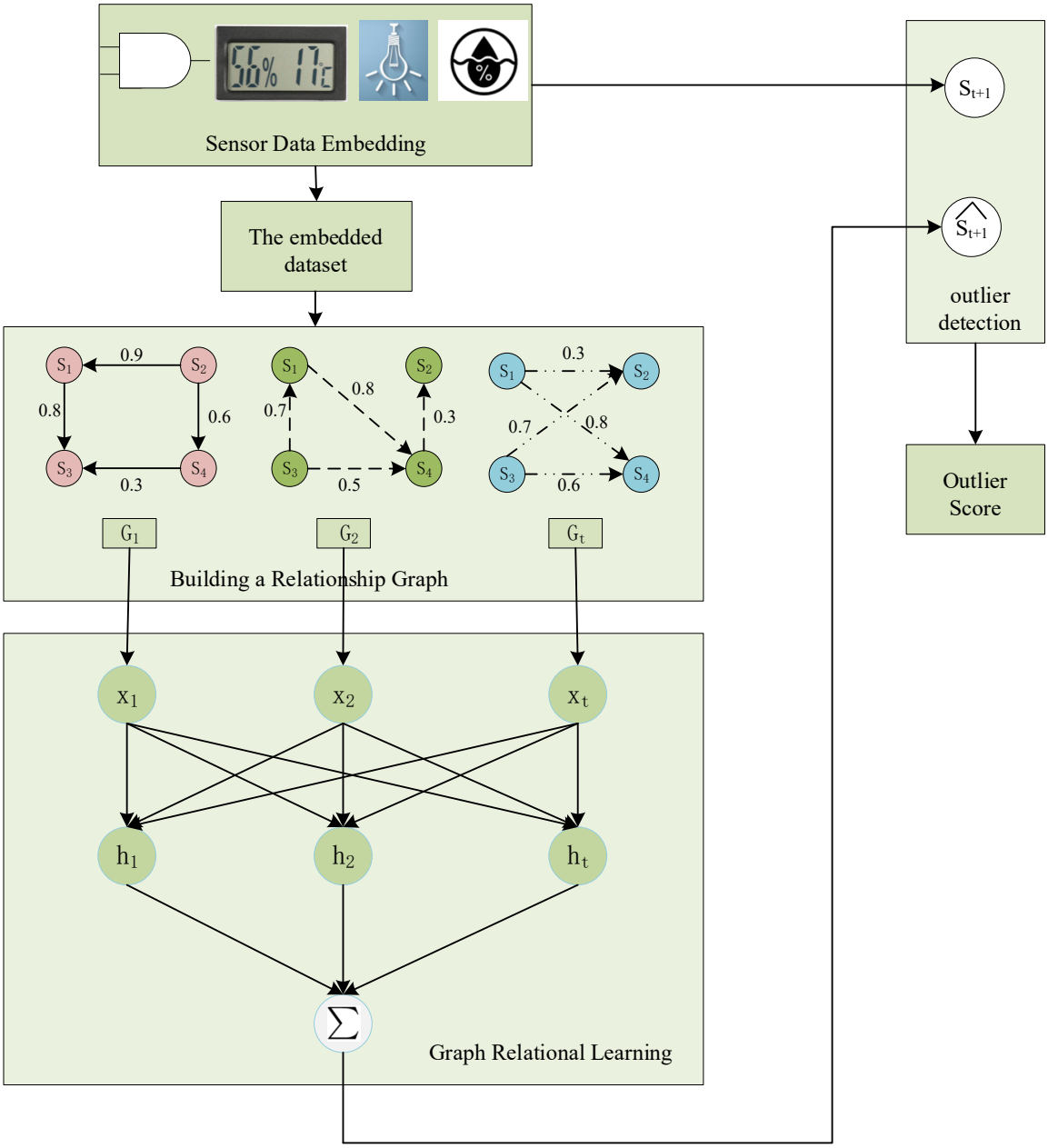

Figure 1. Schematic diagram of the MGAD algorithm.

3.4. Mutual Information Graph Construction

The advantage of this algorithm is that not only the data from the sensors themselves but also the information between the sensors are used to detect anomalies. We will use a graph to accomplish this, where the nodes stand in for sensors and the edges for the dependencies between them, which is mutual information in this scenario.

According to the definition of mutual information, mutual information is symmetric, which means the mutual information is the same for both vertices. However, for the convenience of computation and saving computational resources, we only keep the first half of the large mutual information, as shown in Equation (2). In contrast, mutual information can be obtained by using the equations in Section 2.2.

$$A_{ij} = \begin{cases} MI_{ij}, \text{ when } MI_{ij} \textit{ in } \text{the top } \frac{N}{2} \text{ of the neighbors of sensor } i \\ 0, \text{ elswise} \end{cases} \tag{2}$$

Since the dependency patterns between sensors do not have to be symmetric, we employ a directed graph. This directed graph is represented by an adjacency matrix A, where A_{ij} represents the presence of a directed edge from sensor i to sensor j. In Equation (2), MI_{ij} is the mutual information between node i and node j.

We end up with a mutual information graph as shown in Equation (3). In Equation (3), V is the set of vertices and E is the set of edges between vertices.

$$\begin{aligned} &G = (E,\ V) \\ &V = \{v_i\}\ i \in \{1, 2, \cdots N\} \\ &E = \left\{A_{ij}\right\}\ i, j \in \{1, 2, \cdots N\} \end{aligned} \tag{3}$$

3.5. Graph Structure Learning

We employ a forecasting-based strategy in which, using historical data, we project each sensor's anticipated behavior at any given time. This makes it simple for the user to identify the sensors that significantly deviate from their expected behavior. Additionally, each sensor's expected and observed behavior can be compared by the user to help them understand why the model considers a particular sensor to be anomalous.

3.5.1. Input Layer

We define the input of the proposed algorithm, as shown in Equation (4). $x^{(t)}$ is the input vector consisting of the data from N sensors at moment t.

$$x^{(t)} = \left[v_{ij}\right]_{N \times w} \in R^{N \times w} \tag{4}$$

3.5.2. Hidden Layer

We use a graph attention-based feature extractor to fuse a node's information with its neighbors based on the learned graph structure, thereby capturing the relationships between sensors.

Our feature extractor incorporates the sensor embedding vectors v_i, which characterize the various behaviors of different types of sensors, unlike existing graph attention mechanisms. We calculate the aggregated representation z_i of node i, as shown in Equation (5).

$$\begin{aligned} &z_i^{(t)} = \mathrm{ReLU}(\gamma_{i,i} W x_i^{(t)} + \sum_{j \in N(i)} \gamma_{i,j} W x_j^{(t)}) \textit{ while } N(i) = \left\{j \middle| A_{ij} > 0\right\} \\ &\gamma_{i,j} = \frac{\exp(\chi(i,j))}{\sum_{k \in N(i) \cup \{i\}} \exp(\chi(i,k))} \\ &\chi(i,j) = \mathrm{LeakyReLU}(a \times (g_i^{(t)} \oplus g_j^{(t)})) \\ &g_i^{(t)} = v_i \oplus W x_i^{(t)} \end{aligned} \tag{5}$$

We use LeakyReLU as the nonlinear activation to compute the attention coefficient and normalize the attention coefficients using the softmax function.

In Equation (5), $\oplus$ means concatenation, which is a trainable weight matrix that applies a shared linear transformation to every node. $x_i^{(t)}$ is sensor i's input feature. v_i can obtained from Equation (1). a is a vector of learned coefficients for the attention mechanism, and γ_{ij} is the attention coefficient between node i and its neighbour node j.

3.5.3. Output Layer

From the above feature extractor, we obtain representations for all N nodes, $\left\{z_1^{(t)}, z_2^{(t)}, \cdots, z_N^{(t)}\right\}$. For each $z_i^{(t)}$, to predict the vector of sensor values at time step $t+1$, we element-wise multiply (denoted as $\times$) it with the corresponding time series embedding and use the results across all nodes as the input of stacked fully-connected layers.

The model's predicted output is denoted as $\hat{s^{(t)}}$, as shown in Equation (6).

$$\hat{s^{(t)}} = f_\theta(v_1 \times z_1^{(t)}, v_2 \times z_2^{(t)}, \cdots, v_N \times z_N^{(t)}) \tag{6}$$

We use the mean squared error between the predicted output $\hat{s^{(t)}}$ and the observed data $s^{(t)}$, as the loss function for minimization; in order for the algorithm to be more general, we used the mean square error as the loss function, as shown in Equation (7).

$$L_{MSE} = \frac{1}{T_{train}} \sum_{t=w+1}^{T_{train}} \left\| \hat{s^{(t)}} - s^{(t)} \right\|_2^2 \tag{7}$$

3.6. Anomaly Scoring

Our goal is to identify and explain anomalies that deviate from the learned relationships. As we will demonstrate in our experiments, our model achieves this by computing the unique anomalousness scores for each sensor and combining them into a single anomalousness score for each time tick. This enables the user to localize which sensors are anomalous.

The process of calculating the anomaly score is shown in Equation (8).

$$\begin{aligned} \Gamma_i(t) &= \frac{\left| s_i^{(t)} - \hat{s_i^{(t)}} \right| - \min v_i}{\max v_i - \min v_i}, \ \mathrm{i} \in [t = w+1, \ \mathrm{T}_{train}] \\ O(t) &= \max_{i \in \{1,2,\cdots,N\}} \Gamma_i(t) \end{aligned} \tag{8}$$

$\max v_i, \min v_i$ is the max value and min value of sensor i in the current time window. In Equation (8), $\Gamma_i(t)$ is the anomaly score of sensor i in time tick t. Among the anomaly scores of N sensors, the maximum value is taken to obtain the final anomaly score.

3.7. Pseudocode of MGAD

Finally, we give the pseudocode of the MGAD algorithm as Algorithm 1.

3.8. Time Complexity Analysis

The third step in the MGAD algorithm is the most time consuming and we focus on the time complexity of the third step. In the third step (Section 3.5), graph structure learning leads to $O(\sum_{l=1}^{D} M_l^2 \cdot K_l^2 \cdot C_{l-1} \cdot C_l)$ time complexity. Thus, MGAD has $O(\sum_{l=1}^{D} M_l^2 \cdot K_l^2 \cdot C_{l-1} \cdot C_l)$ time complexity. D is the number of convolutional layers that a neural network has, i.e., the depth of the network.

Compared with the benchmark algorithms used in the experiments in Section 5, the time complexity of the MGAD algorithm is higher than that of KNN, PCA, OCSVM, and AE, because the MGAD algorithm is an anomaly detection algorithm based on graph convolutional neural networks. The time complexity of the MGAD algorithm is close to that of the LSTM-VAE, DAGMM, AnoGAN, and MAD-GAN. These algorithms are neural

network-based algorithms and their time complexity is determined by the depth of the neural network.

Algorithm 1 MGAD

Input: input data $x^{(t)} = \left[v_{ij}\right]_{N\times w} \in R^{N\times w}$

Output: Outlier scores $\{o_1, o_2, \ldots, o_i, \ldots, o_N\}$

1. sensors embedding to obtain the embedding vector for each sensor

$$v_i \in R^d, i \in \{1, 2, \cdots N\}$$

2. mutual information graph construction

$$A_{ij} = \begin{cases} MI_{ij}, \text{ when } MI_{ij} \textit{ in} \text{ the top } \frac{N}{2} \text{ of the neighbors of sensor } i \\ 0, \text{ elswise} \end{cases}$$

$$G = (E,\ V)$$
$$E = \{v_i\}\ i \in \{1, 2, \cdots N\}$$
$$V = \left\{A_{ij}\right\}\ i, j \in \{1, 2, \cdots N\}$$

3. **for** i in max epoch number:

carry out graph structure learning

(1) input layer $x^{(t)} = \left[v_{ij}\right]_{N\times w} \in R^{N\times w}$

(2) hidden layer

$$z_i^{(t)} = \mathrm{ReLU}(\gamma_{i,i} W x_i^{(t)} + \sum_{j\in N(i)} \gamma_{i,j} W x_j^{(t)}) \textit{ while } N(i) = \left\{j \middle| A_{ij} > 0\right\}$$
$$\gamma_{i,j} = \frac{\exp(\chi(i,j))}{\sum_{k\in N(i)\cup\{i\}} \exp(\chi(i,k))}$$
$$\chi(i,j) = \mathrm{LeakyReLU}(a \times (g_i^{(t)} \oplus g_j^{(t)}))$$
$$g_i^{(t)} = v_i \oplus W x_i^{(t)}$$

(3) output layer

$$\hat{s^{(t)}} = f_\theta(v_1 \times z_1^{(t)}, v_2 \times z_2^{(t)}, \cdots, v_N \times z_N^{(t)})$$
$$L_{MSE} = \frac{1}{T_{train}} \sum_{t=w+1}^{T_{train}} \left\| \hat{s^{(t)}} - s^{(t)} \right\|_2^2$$

4. end **for**

5. obtain the anomaly score

$$\Gamma_i(t) = \frac{\left|s_i^{(t)} - \hat{s_i^{(t)}}\right| - \min v_i}{\max v_i - \min v_i},\ \mathrm{i} \in [t = w+1,\ \mathrm{T}_{train}]$$
$$O(t) = \max_{i\in\{1,2,\cdots,N\}} \Gamma_i(t)$$

6. **return** $\{o_1, o_2, \ldots, o_i, \ldots, o_N\}$

4. Experimental Results and Discussion

This section outlines the experimental dataset, baselines, and evaluation metrics used in the assessment of the proposed algorithm. We also give a detailed discussion about the experimental results.

4.1. Datasets

Four real-world datasets of varying sizes and types were used in a series of comparative experiments that we ran to confirm the efficacy of the proposed algorithm. They came from a variety of domains. Information about these real-world datasets, including number of data, number of feature, and percentage of anomaly, is shown in Table 1.

Table 1. Information of real-world datasets.

Dataset	Number of Data	Number of Feature	Percentage of Anomaly
SWaT	92,501	51	11.97%
WADI	136,070	127	5.99%
Credit-g	284,807	31	4.59%
GECCO IoT	248,535	11	8.52%

(1) The Secure Water Treatment (SWaT) dataset comes from a water treatment test bed coordinated by Singapore's Public Utility Board [46] (https://itrust.sutd.edu.sg/itrust-labs_datasets/dataset_info/ accessed on 5 January 2024);
(2) Water Distribution (WADI), which is an extension of SwaT, is a distribution system comprising a larger number of water distribution pipelines [47] (https://itrust.sutd.edu.sg/itrust-labs_datasets/dataset_info/ accessed on 5 January 2024);
(3) Credit-g contains credit card transaction data [48] (https://www.openml.org/d/1597 accessed on 5 January 2024);
(4) The GECCO IoT dataset contains IoT data for drinking water monitoring and was provided by Thüringer Fernwasserversorgung and the IMProvT research project [49] (https://www.spotseven.de/gecco/gecco-challenge/gecco-challenge-2018/ accessed on 5 January 2024).

SwaT, WADI, and Credit-g are time series data, which contain point-wise outliers. GECCO IoT contains event-driven sequential data, which contain point-wise outliers.

4.2. Baselines

We compared the performance of the MGAD algorithm with eight state-of-the-art outlier detection algorithms, including statistical approaches, machine-learning approaches, and neural network-based methods. These eight outlier detection algorithms are as follows:

(1) KNN: K nearest neighbors generates an anomaly score based on the distance between each point and its kth nearest neighbor [50];
(2) OCSVM is trained using normal data to identify the limits of normal and abnormal data [51];
(3) PCA: Principal component analysis discovers a low-dimensional projection that largely accounts for the data's variance. The reconstruction error of this projection is called the anomaly score [52];
(4) AE: Autoencoders comprise a decoder and an encoder that rebuilds data samples. The anomaly score is the reconstruction error [27].
(5) LSTM-VAE: This algorithm combines LSTM and VAE by substituting the feed-forward network in a VAE with LSTM. The anomaly score is the reconstruction error [53];
(6) DAGMM: This algorithm combine deep Autoencoders with a Gaussian mixture model to generate a low-dimensional representation, and reconstruction error is the anomaly score [54];
(7) AnoGAN: This algorithm uses a deep convolutional generative adversarial network to learn a manifold of normal anatomical variability, accompanying a novel anomaly scoring scheme based on the mapping from image space to a latent space [55];
(8) MAD-GAN: After training a GAN model on normal data, each sample's anomaly score is calculated using the reconstruction-based method and the LSTM-RNN discriminator [56].

4.3. Evaluation Metrics

4.3.1. ROC (Receiver Operating Characteristic)

The receiver operating characteristic (ROC) curve is an important tool often utilized in assessing the effectiveness of binary classification algorithms. Unlike many other metrics that provide single values, the ROC curve offers a graphical representation of a classifier's performance. A higher ROC value closer to 1 signifies a more accurate detection model. In contrast, an ROC value equal to or lower than 0.5 indicates the futility of the inspection model for practical use.

4.3.2. F1-Score

To assess the effectiveness of the proposed algorithm and baselines, we employ the F1-score (F1) over the test dataset. Equation (9) shows the calculation process of F1-score.

In Equation (9), TP, TN, FP, and FN are the numbers of true positives, true negatives, false positives, and false negatives.

$$F1 = \frac{2 \times \text{Pre}c \times \text{Rec}}{\text{Pre}c \times \text{Rec}}, \; while \; \text{Pre}c = \frac{TP}{TP + FP} \text{ and } \text{Rec} = \frac{TP}{TP + FN} \tag{9}$$

Because the datasets used in our experiments are imbalanced, the metric of the F1-score was chosen because it works well with unbalanced data. We set the threshold for anomaly detection using the maximum anomaly score across the training dataset. An anomaly score above the threshold at the test time will be considered abnormal.

4.3.3. AP (Average Precision)

Evaluating outlier detection models can be challenging, particularly in the absence of labeled data or ground truth data for comparison. One approach to assessing the performance of outlier detection models is by utilizing the average precision (AP). The AP calculates the average precision across all potential thresholds, with a higher AP value indicating a superior model. This metric is particularly effective for outlier detection scenarios with rare anomalies or imbalanced data, as it prioritizes the positive class (anomalies) over the negative class (normal instances).

Nonetheless, the AP may not provide a comprehensive reflection of the model's accuracy or specificity, as it fails to consider true negatives or false negatives. Another method for evaluating outlier detection models is through external validation, which entails comparing the outcomes with alternative sources of information, such as input from domain experts, feedback, or historical data.

4.4. Experimental Setup

In subsequent experiments, we will use a Windows PC equipped with an AMD Ryzen 7 5800H CPU and 16 GB of memory. We implement the proposed algorithm and baselines in PyTorch version 1.7.1, CUDA 9.2.

In the dataset used for experiments, 70% of the data are used as a training dataset and 30% as a test dataset. The models are trained using the Adam optimizer to speed up the training process. We train models for up to 100 epochs and use early stopping with a patience of 10. The learning rate is set to 0.001, the epoch is set to 3000, and the embedding dimension is set to 16.

4.5. Experiment Result and Discussion

In this section, we give the experimental results of MGAD for the datasets in Tables 2–4. The highest score is marked in bold, which means that the algorithm achieves the best performance for this dataset.

Table 2. ROC scores of outlier detector performance.

Algorithm	SWaT	WADI	Credit-g	GECCO IoT	Average ROC
KNN	0.880	0.928	0.871	0.891	0.892
OCSVM	0.981	0.605	0.765	0.620	0.743
PCA	0.640	0.803	0.680	0.893	0.754
AE	0.677	0.880	0.717	**0.910**	0.796
LSTM-VAE	0.602	0.645	0.644	0.699	0.648
DAGMM	0.609	0.891	0.736	0.679	0.729
AnoGAN	0.964	0.891	0.813	0.704	0.843
MAD-GAN	0.834	0.730	**0.956**	0.851	0.843
MGAD	**0.988**	**0.939**	0.867	0.897	**0.923**

Table 3. F1-scores of outlier detector performance.

Algorithm	SWaT	WADI	Credit-g	GECCO IoT	Average F1
KNN	0.736	0.683	0.747	**0.834**	0.750
OCSVM	0.677	0.661	**0.835**	0.782	0.739
PCA	0.656	0.757	0.83	0.7	0.736
AE	**0.833**	0.664	0.825	0.762	0.771
LSTM-VAE	0.668	0.777	0.707	0.723	0.719
DAGMM	0.780	0.725	0.719	0.791	0.754
AnoGAN	0.678	0.658	0.692	0.65	0.670
MAD-GAN	0.698	0.692	0.655	0.739	0.696
MGAD	0.822	**0.841**	0.832	0.822	**0.829**

Table 4. AP of outlier detector performance.

Algorithm	SWaT	WADI	Credit-g	GECCO IoT	Average AP
KNN	0.673	**0.911**	0.780	0.858	0.806
OCSVM	0.588	0.721	**0.913**	0.651	0.718
PCA	0.947	0.899	0.572	0.64	0.765
AE	0.935	0.859	0.674	0.724	0.798
LSTM-VAE	0.687	0.651	0.883	0.72	0.735
DAGMM	0.566	0.646	0.819	0.565	0.649
AnoGAN	0.700	0.749	0.766	**0.893**	0.777
MAD-GAN	0.875	0.804	0.829	0.751	0.815
MGAD	**0.966**	0.889	0.886	0.883	**0.906**

As can be seen from Table 2 and Figure 2, the MGAD algorithm achieved the best performance on the SWaT and WADI datasets, the third best performance on the Credit-g dataset, and the second best performance on the GECCO IoT dataset. Lastly, the MGAD algorithm had the highest average ROC.

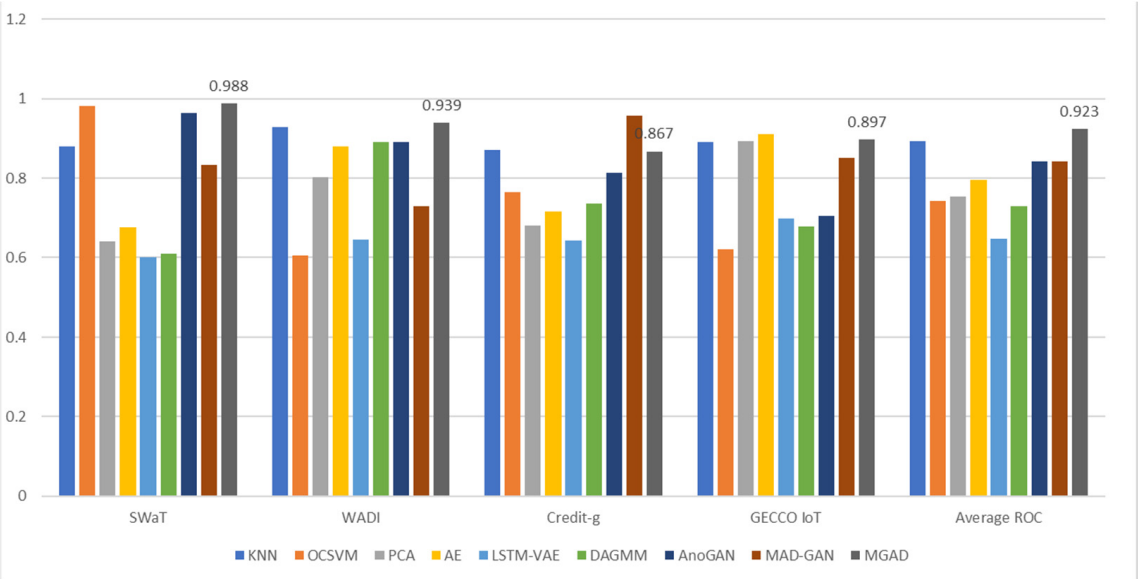

Figure 2. Comparison experiments results of time series detection in ROC scores.

From Table 3 and Figure 3, the MGAD algorithm achieves the second best performance on the SWaT dataset, the best performance on the WADI dataset, and the second best performance on both the Credit-g dataset and the GECCO IoT dataset.

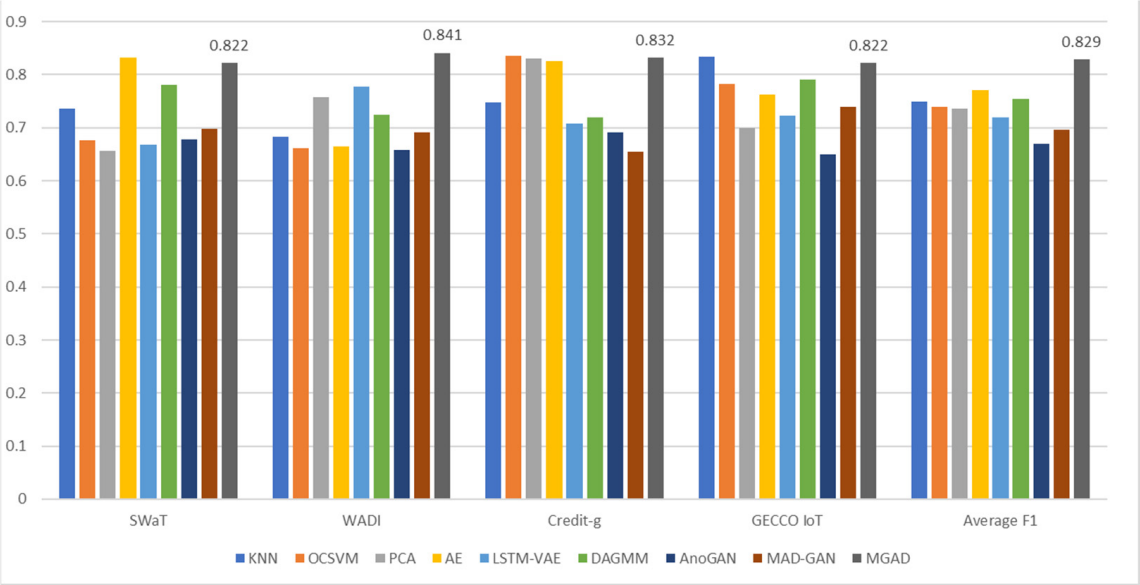

Figure 3. Comparison experiments results of time series detection in F1-scores.

MGAD also has the highest average AP, which can be seen from the data in Table 4 and Figure 4. In all four datasets, MGAD's algorithm is either the optimal or the front runner in terms of performance.

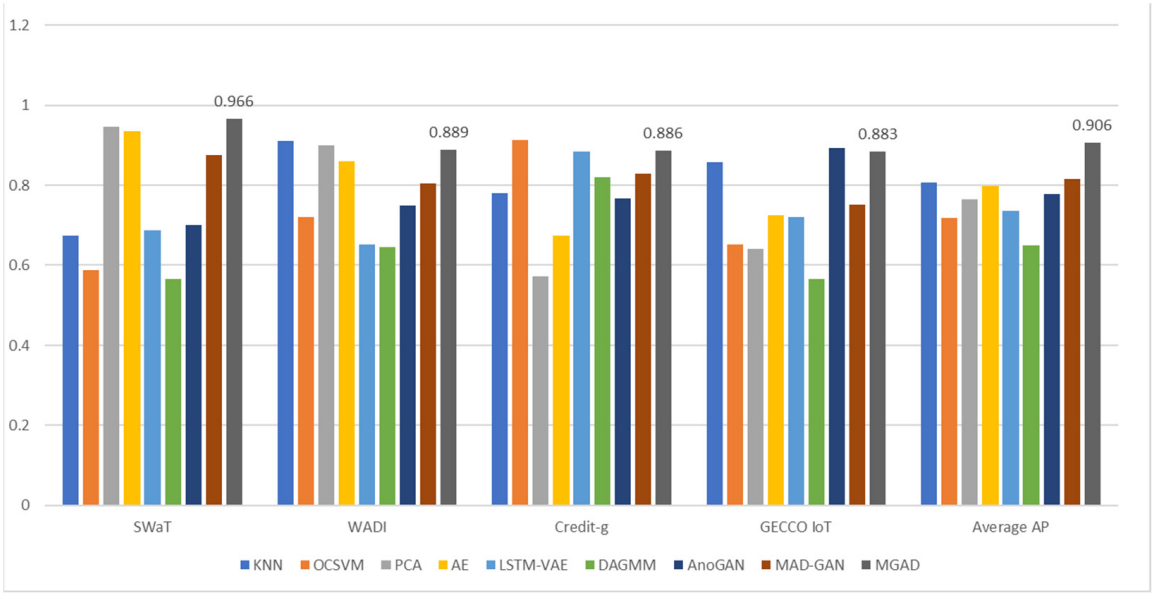

Figure 4. Comparison experiments results of time series detection in AP scores.

With an average ROC of 0.923, an average F1-score of 0.829, and an average accuracy of 0.906, the proposed MGAD achieves the best performance. It is noteworthy that, by analyzing the data in Tables 2–4, the MGAD algorithm does not achieve the best performance in every dataset, but it is a relatively high-level performance, if not the first, which can be seen in Figures 2–4.

5. Conclusions

In this paper, we present a novel unsupervised outlier detection algorithm based on mutual information and graph embedding called MGAD. MGAD can be mainly divided

into four phases: (1) Embedding of sensor data; (2) Constructing a relationship graph between sensors using their mutual information about each other; (3) Learning the relationship graph between sensors using a graph attention mechanism; (4) Comparing the predicted values with the real sensor data to detect potential outliers.

The experimental results in Section 4 show that the MGAD algorithm has a strong detection performance along with a high interpretability. The MGAD algorithm is an anomaly detection algorithm based on graph convolutional neural networks, and thus requires a large amount of data to train the model, which costs a lot of computational resources and training time. In addition, in some sensor applications, the transmission channel reliability of the data is not high, which leads to the presence of noise in the data, which can also bring negative effects on the training of the MGAD algorithm. These are further future research directions for MGAD algorithms.

Author Contributions: Conceptualization, Y.H.; data curation, S.L., Y.G. and W.C.; funding acquisition, W.L.; methodology, Y.H.; software, W.C.; supervision, W.L.; validation, Y.H., S.L., Y.G. and W.C.; visualization, S.L. and Y.G.; writing—original draft, Y.H.; writing—review and editing, Y.H. and W.L. All authors have read and agreed to the published version of the manuscript.

Funding: This study was supported by the National Natural Science Foundation of China (No. 61862011), the Guangxi Natural Science Foundation (No. 2019GXNSFGA245004), and the Innovation Project of Guangxi Graduate Education (No. YCBZ2023128).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available in the article.

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

References

1. Hawkins, D.M. *Identification of Outliers*; Springer: Dordrecht, The Netherlands, 1980; Volume 11.
2. Wang, H.; Wang, W.; Liu, Y.; Alidaee, B. Integrating Machine Learning Algorithms with Quantum Annealing Solvers for Online Fraud Detection. *IEEE Access* **2022**, *10*, 75908–75917. [CrossRef]
3. Lai, G. Artificial Intelligence Techniques for Fraud Detection. *Preprints* **2023**, 2023121115. [CrossRef]
4. Sabitha, R.; Shukla, A.P.; Mehbodniya, A.; Shakkeera, L.; Reddy, P.C.S. A Fuzzy Trust Evaluation of Cloud Collaboration Outlier Detection in Wireless Sensor Networks. *Ad Hoc Sens. Wirel. Netw.* **2022**, *53*, 165–188.
5. Bhattacharjee, P.; Garg, A.; Mitra, P. KAGO: An approximate adaptive grid-based outlier detection approach using kernel density estimate. *Pattern Anal. Appl.* **2021**, *24*, 1825–1846. [CrossRef]
6. Zhang, Y.-L.; Zhou, J.; Zheng, W.; Feng, J.; Li, L.; Liu, Z.; Li, M.; Zhang, Z.; Chen, C.; Li, X.; et al. Distributed Deep Forest and its Application to Automatic Detection of Cash-Out Fraud. *ACM Trans. Intell. Syst. Technol.* **2019**, *10*, 1–19. [CrossRef]
7. Zhang, H.; Zhao, S.; Liu, R.; Wang, W.; Hong, Y.; Hu, R. Automatic Traffic Anomaly Detection on the Road Network with Spatial-Temporal Graph Neural Network Representation Learning. *Wirel. Commun. Mob. Comput.* **2022**, *2022*, 4222827. [CrossRef]
8. Fournier, N.; Farid, Y.Z.; Patire, A. Erroneous High Occupancy Vehicle Lane Data: Detecting Misconfigured Traffic Sensors with Machine Learning. *Transp. Res. Rec. J. Transp. Res. Board* **2022**, *2677*, 1593–1610. [CrossRef]
9. Dixit, P.; Bhattacharya, P.; Tanwar, S.; Gupta, R. Anomaly detection in autonomous electric vehicles using AI techniques: A comprehensive survey. *Expert Syst.* **2022**, *39*, e12754. [CrossRef]
10. Watts, J.; Van Wyk, F.; Rezaei, S.; Wang, Y.; Masoud, N.; Khojandi, A. A Dynamic Deep Reinforcement Learning-Bayesian Framework for Anomaly Detection. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 22884–22894. [CrossRef]
11. Mansour, R.F.; Escorcia-Gutierrez, J.; Gamarra, M.; Villanueva, J.A.; Leal, N. Intelligent video anomaly detection and classification using faster RCNN with deep reinforcement learning model. *Image Vis. Comput.* **2021**, *112*, 104229. [CrossRef]
12. Zhao, Y.; Deng, B.; Shen, C.; Liu, Y.; Lu, H.; Hua, X.S. Spatio-Temporal AutoEncoder for Video Anomaly Detection. In Proceedings of the 25th ACM International Conference on Multimedia (MM), Mountain View, CA, USA, 23–27 October 2017.
13. Dang, T.T.; Ngan, H.Y.; Liu, W. Distance-Based k-Nearest Neighbors Outlier Detection Method in Large-Scale Traffic Data. In Proceedings of the IEEE International Conference on Digital Signal Processing (DSP), Singapore, 21–24 July 2015.
14. Chandola, V.; Banerjee, A.; Kumar, V. Anomaly detection: A survey. *ACM Comput. Surv. (CSUR)* **2009**, *41*, 1–58. [CrossRef]
15. Saleh, M.; Othman, S.H.; Driss, M.; Al-Dhaqm, A.; Ali, A.; Yafooz, W.M.S.; Emara, A.-H.M. A Metamodeling Approach for IoT Forensic Investigation. *Electronics* **2023**, *12*, 524. [CrossRef]
16. Atitallah, S.B.; Driss, M.; Ghezala, H.B. FedMicro-IDA: A federated learning and microservices-based framework for IoT data analytics. *Internet Things* **2023**, *23*, 100845. [CrossRef]

17. Alrayes, F.S.; Zakariah, M.; Driss, M.; Boulila, W. Deep Neural Decision Forest (DNDF): A Novel Approach for Enhancing Intrusion Detection Systems in Network Traffic Analysis. *Sensors* **2023**, *23*, 8362. [CrossRef]
18. Ntroumpogiannis, A.; Giannoulis, M.; Myrtakis, N.; Christophides, V.; Simon, E.; Tsamardinos, I. A meta-level analysis of online anomaly detectors. *VLDB J.* **2023**, *32*, 845–886. [CrossRef]
19. Wang, Z.; Shao, L.; Cheng, K.; Liu, Y.; Jiang, J.; Nie, Y.; Li, X.; Kuang, X. ICDF: Intrusion collaborative detection framework based on confidence. *Int. J. Intell. Syst.* **2022**, *37*, 7180–7199. [CrossRef]
20. Heigl, M.; Weigelt, E.; Urmann, A.; Fiala, D.; Schramm, M. Exploiting the Outcome of Outlier Detection for Novel Attack Pattern Recognition on Streaming Data. *Electronics* **2021**, *10*, 2160. [CrossRef]
21. Souiden, I.; Omri, M.N.; Brahmi, Z. A survey of outlier detection in high dimensional data streams. *Comput. Sci. Rev.* **2022**, *44*, 100463. [CrossRef]
22. Pei, Y.; Zaïane, O. A Synthetic Data Generator for Clustering and Outlier Analysis. 2006. Available online: https://era.library.ualberta.ca/items/63beb6a7-cc50-4ffd-990b-64723b1e4bf9 (accessed on 5 January 2024).
23. Chaudhry, H.N.; Javed, Y.; Kulsoom, F.; Mehmood, Z.; Khan, Z.I.; Shoaib, U.; Janjua, S.H. Sentiment Analysis of before and after Elections: Twitter Data of U.S. Election 2020. *Electronics* **2021**, *10*, 2082. [CrossRef]
24. Chalapathy, R.; Toth, E.; Chawla, S. Group Anomaly Detection Using Deep Generative Models. In Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD), Dublin, Ireland, 10–14 September 2019.
25. Chenaghlou, M.; Moshtaghi, M.; Leckie, C.; Salehi, M. Online Clustering for Evolving Data Streams with Online Anomaly Detection. In Proceedings of the 22nd Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), Melbourne, Australia, 3–6 June 2018.
26. Sharma, V.; Kumar, R.; Cheng, W.-H.; Atiquzzaman, M.; Srinivasan, K.; Zomaya, A. NHAD: Neuro-Fuzzy Based Horizontal Anomaly Detection In Online Social Networks. *IEEE Trans. Knowl. Data Eng.* **2018**, *30*, 2171–2184. [CrossRef]
27. Sikder, M.N.K.; Batarseh, F.A. Outlier detection using AI: A survey. In *AI Assurance*; Academic Press: Cambridge, MA, USA, 2023; pp. 231–291.
28. Li, Z.; Zhu, Y.; Van Leeuwen, M. A Survey on Explainable Anomaly Detection. *ACM Trans. Knowl. Discov. Data* **2023**, *18*, 1–54. [CrossRef]
29. Su, X.; Xue, S.; Liu, F.; Wu, J.; Yang, J.; Zhou, C.; Hu, W.; Paris, C.; Nepal, S.; Jin, D.; et al. A Comprehensive Survey on Community Detection with Deep Learning. *IEEE Trans. Neural Netw. Learn. Syst.* 2022; *Early Access*.
30. Huang, Y.; Liu, W.; Li, S.; Guo, Y.; Chen, W. Interpretable Single-dimension Outlier Detection (ISOD): An Unsupervised Outlier Detection Method Based on Quantiles and Skewness Coefficients. *Appl. Sci.* **2023**, *14*, 136. [CrossRef]
31. Su, Y.; Zhao, Y.; Niu, C.; Liu, R.; Sun, W.; Pei, D. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019.
32. Wang, Y.; Du, X.; Lu, Z.; Duan, Q.; Wu, J. Improved LSTM-based Time-Series Anomaly Detection in Rail Transit Operation Environments. *IEEE Trans. Ind. Inform.* **2022**, *18*, 9027–9036. [CrossRef]
33. Wei, Y.; Jang-Jaccard, J.; Xu, W.; Sabrina, F.; Camtepe, S.; Boulic, M. LSTM-autoencoder-based anomaly detection for indoor air quality time-series data. *IEEE Sens. J.* **2023**, *23*, 3787–3800. [CrossRef]
34. Hundman, K.; Constantinou, V.; Laporte, C.; Colwell, I.; Soderstrom, T. Detecting Spacecraft Anomalies Using LSTMs and Nonparametric Dynamic Thresholding. In Proceedings of the 24th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), London, UK, 19–23 August 2018.
35. Malhotra, P.; Ramakrishnan, A.; Anand, G.; Vig, L.; Agarwal, P.; Shroff, G. LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection. *arXiv* **2016**, arXiv:1607.00148.
36. Grover, A.; Leskovec, J. node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016.
37. Perozzi, B.; Al-Rfou, R.; Skiena, S. Deepwalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 24–27 August 2014.
38. Zhang, X.; Hu, W.; Chen, S.; Maybank, S. Graph-embedding-based learning for robust object tracking. *IEEE Trans. Ind. Electron.* **2013**, *61*, 1072–1084. [CrossRef]
39. Deng, A.; Hooi, B. Graph neural network-based anomaly detection in multivariate time series. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 2–9 February 2021.
40. Gu, Y.; Li, G.; Gu, J.; Jung, J.J. Graph embedding-based Anomaly localization for HVAC system. *J. Build. Eng.* **2023**, *77*, 107511. [CrossRef]
41. Safaei, M.; Driss, M.; Boulila, W.; Sundararajan, E.A.; Safaei, M. Global outliers detection in wireless sensor networks: A novel approach integrating time-series analysis, entropy, and random forest-based classification. *Softw. Pract. Exp.* **2022**, *52*, 277–295. [CrossRef]
42. Zhao, H.; Wang, Y.; Duan, J.; Huang, C.; Cao, D.; Tong, Y.; Xu, B.; Bai, J.; Tong, J.; Zhang, Q. Multivariate time-series anomaly detection via graph attention network. In Proceedings of the 2020 IEEE International Conference on Data Mining (ICDM), Sorrento, Italy, 17–20 November 2020; IEEE: Piscataway, NJ, USA, 2020.

43. Goodge, A.; Hooi, B.; Ng, S.K.; Ng, W.S. Lunar: Unifying local outlier detection methods via graph neural networks. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 22 February–1 March 2022.
44. Rajmohan, R.; Kumar, T.A.; Sandhya, S.G.; Hu, Y.-C. R-GCN: A residual-gated recurrent unit convolution network model for anomaly detection in blockchain transactions. *Multimed. Tools Appl.* **2024**, 1–25. [CrossRef]
45. Atitallah, S.B.; Driss, M.; Almomani, I. A novel detection and multi-classification approach for IoT-malware using random forest voting of fine-tuning convolutional neural networks. *Sensors* **2022**, *22*, 4302. [CrossRef]
46. Mathur, A.P.; Tippenhauer, N.O. SWaT: A water treatment testbed for research and training on ICS security. In Proceedings of the 2016 International Workshop on Cyber-Physical Systems for Smart Water Networks (CySWater), Vienna, Austria, 11 April 2016; IEEE: Piscataway, NJ, USA, 2016.
47. Ahmed, C.M.; Palleti, V.R.; Mathur, A.P. WADI: A water distribution testbed for research in the design of secure cyber physical systems. In Proceedings of the 3rd International Workshop on Cyber-Physical Systems for Smart Water Networks, Pittsburgh, PA, USA, 18–21 April 2017.
48. Dal Pozzolo, A.; Caelen, O.; Johnson, R.A.; Bontempi, G. Calibrating probability with undersampling for unbalanced classification. In Proceedings of the 2015 IEEE Symposium Series on Computational Intelligence, Cape Town, South Africa, 7–10 December 2015; IEEE: Piscataway, NJ, USA, 2015.
49. Darban, Z.Z.; Webb, G.I.; Pan, S.; Aggarwal, C.C.; Salehi, M. Deep learning for time series anomaly detection: A survey. *arXiv* **2022**, arXiv:2211.05244.
50. Angiulli, F.; Pizzuti, C. Fast outlier detection in high dimensional spaces. In Proceedings of the European Conference on Principles of Data Mining and Knowledge Discovery, Helsinki, Finland, 19–23 August 2002; Springer: Berlin/Heidelberg, Germany, 2002.
51. Schölkopf, B.; Williamson, R.C.; Smola, A.; Shawe-Taylor, J.; Platt, J. Support vector method for novelty detection. *Adv. Neural Inf. Process. Syst.* **2000**, *12*, 582–588.
52. Shyu, M.L.; Chen, S.C.; Sarinnapakorn, K.; Chang, L. A novel anomaly detection scheme based on principal component classifier. In Proceedings of the IEEE Foundations and New Directions of Data Mining Workshop, Melbourne, FL, USA, 19–22 November 2003; IEEE Press: Piscataway, NJ, USA, 2003.
53. Park, D.; Hoshi, Y.; Kemp, C.C. Kemp, A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder. *IEEE Robot. Autom. Lett.* **2018**, *3*, 1544–1551. [CrossRef]
54. Zong, B.; Song, Q.; Min, M.R.; Cheng, W.; Lumezanu, C.; Cho, D.; Chen, H. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
55. 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 25th Biennial International Conference on Information Processing in Medical Imaging (IPMI), Boone, NC, USA, 25–30 June 2017.
56. Li, D.; Chen, D.; Jin, B.; Shi, L.; Goh, J.; Ng, S.K. MAD-GAN: Multivariate Anomaly Detection for Time Series Data with Generative Adversarial Networks. In Proceedings of the 28th International Conference on Artificial Neural Networks (ICANN), Munich, Germany, 17–19 September 2019.