Alexandria Engineering Journal 108 (2024) 436–444

Contents lists available at ScienceDirect

Alexandria Engineering Journal

journal homepage: www.elsevier.com/locate/aej

Original article

DDANF: Deep denoising autoencoder normalizing flow for unsupervised multivariate time series anomaly detection

Xigang Zhao [a], Peng Liu [a,*], Saïd Mahmoudi [b], Sahil Garg [c,d], Georges Kaddoum [c], Mohammad Mehedi Hassan [e]

[a] School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou, 310018, China
[b] Computer Science Department, Faculty of Engineering, University of Mons, Mons, 7000, Belgium
[c] École de technologie supérieure, Montreal, H3C 1K3, Canada
[d] Centre for Research Impact & Outcome, Chitkara University Institute of Engineering and Technology, Chitkara University, Rajpura, 140401, Punjab, India
[e] Department of Information Systems, College of Computer and Information Sciences, King Saud University, Riyadh, 11543, Saudi Arabia

ARTICLE INFO

Keywords:
Anomaly detection
Normalizing flow
Time series
Autoencoder

ABSTRACT

In recent years, the proliferation of IoT technologies and the widespread adoption of wireless sensors across various critical infrastructures such as power plants, service monitoring systems, space and earth exploration missions, and water treatment facilities have resulted in the generation of vast quantities of multivariate time series data. Within this context, unsupervised anomaly detection has emerged as a pivotal yet challenging problem in time series research, necessitating machine learning models capable of identifying rare anomalies amidst massive datasets. Traditionally, unsupervised methods have approached this issue by learning representations of primary patterns within sequences and detecting deviations through reconstruction errors. However, the effectiveness of this approach is often limited due to the intricate dynamics and diverse patterns inherent in these dynamic systems. Moreover, many existing unsupervised anomaly detection techniques fail to fully exploit inter-feature relationships within multivariate time series data, thereby overlooking a crucial criterion for accurate detection. To address these shortcomings, this paper introduces a novel unsupervised method for multivariate time series anomaly detection based on normalized flows and autoencoders. Central to our approach is the incorporation of a channel shuffling mechanism during training, enhancing the model's capacity to discern inter-channel patterns and anomalies. Concurrently, the application of normalized flows within the autoencoder framework serves to constrain the latent space, effectively isolating anomalies and improving detection accuracy. Experimental validation conducted on two large-scale public datasets demonstrates the efficacy of the proposed method compared to established benchmarks, highlighting its superior performance.

1. Introduction

With the rapid advancement of Internet of Things (IoT) technology, various real-world systems ranging from large industrial machinery to intelligent robots and IT servers are increasingly equipped with sensors capable of capturing extensive multivariate time series data. This data serves as a valuable resource for predicting future trends, identifying patterns, and analyzing historical operational conditions. Of particular significance in time series analysis is anomaly detection, which plays a crucial role in enhancing operational efficiency and security. Effective anomaly detection not only helps mitigate equipment wear and tear but also enables timely detection of external threats. This capability is critically needed across diverse applications including mechanical fault diagnosis, network intrusion detection, and financial fraud prevention. Thus, the development of robust anomaly detection methods remains a pressing priority in leveraging IoT data for improved system performance and resilience in various domains.

Due to the vast scale of time series data and the infrequent occurrence of anomalies within them, detecting anomalies in time series poses a significant challenge due to its highly imbalanced nature. Moreover, the manual annotation of such voluminous time series data is prohibitively expensive and impractical. Consequently, research in time series anomaly detection has predominantly focused on unsupervised learning approaches. In recent decades, a plethora of classical methods have emerged to address this challenge [1]. Machine learning techniques, including density-based local outlier factor (LOF) [2], distance-based K-nearest neighbors algorithm [3], clustering-based One-Class

* Corresponding author.
E-mail address: perryliu@hdu.edu.cn (P. Liu).

https://doi.org/10.1016/j.aej.2024.07.013
Received 24 April 2024; Received in revised form 18 June 2024; Accepted 4 July 2024
Available online 2 August 2024

Support Vector Machine (OC-SVM) [4], and label estimation methods [5], have demonstrated promising results in terms of accuracy and computational efficiency. However, these methods often encounter the curse of dimensionality, particularly when applied to high-dimensional data [6]. This issue escalates in today's information-rich society where high-dimensional time series data is prevalent across various domains. Addressing these computational challenges remains imperative for advancing the effectiveness of anomaly detection methods in real-world applications. In recent years, the rapid advancement of deep learning and neural networks has significantly enhanced the capability to handle high-dimensional data beyond the capacity of traditional methods. Deep learning techniques are increasingly prevalent in the realm of anomaly detection in time series data, leveraging innovations such as autoencoders [7] and deep generative models [8]. These methods share a fundamental premise: normal data can be more accurately reconstructed or generated from latent representations compared to anomalous instances [9]. However, research has uncovered challenges specific to these approaches, notably autoencoders' susceptibility to overfitting, which can lead to diminished reconstruction errors for anomalies [9]. Moreover, as autoencoders scale in complexity, they risk memorizing rather than truly learning the underlying data structure, compromising anomaly detection accuracy. Similarly, methods based on generative adversarial networks (GANs) confront issues such as training instability and mode collapse [10], necessitating meticulous model tuning efforts by researchers. Additionally, multivariate time series data distinguishes itself from univariate counterparts by encompassing multiple variables or channels with intricate interdependencies, which anomalies may disrupt. Regrettably, most existing anomaly detection methods do not explicitly account for these inter-variable relationships. Addressing these challenges is crucial for advancing anomaly detection capabilities in complex real-world applications.

In this paper, we introduce a novel method for detecting anomalies in multivariate time series data, termed Deep Denoising Autoencoder Normalizing Flow (DDANF). DDANF is a reconstruction model based on autoencoders. However, unlike conventional autoencoders, it utilizes Normalizing Flow and denoising methods to escape the dilemma of overfitting inherent in autoencoders. Additionally, the model is trained unsupervised exclusively on normal data, which effectively reduces the cost of data acquisition. Furthermore, the channel shuffling mechanism endows DDANF with the capability to learn the associations between channels or variables. One distinctive feature of DDANF is its utilization of Normalizing Flow within the autoencoder's bottleneck layer. This integration constrains the latent space, preventing the autoencoder from merely replicating input data and ensuring that anomalies cannot be accurately reconstructed. Moreover, Normalizing Flow enables DDANF to assess anomaly likelihoods based on perceptual features, enhancing the model's interpretability at a semantic level. Additionally, DDANF incorporates a novel channel shuffling and sequential information embedding module. This module enhances the model's ability to discern relationships and dependencies among different channels or variables explicitly. By shuffling channels, DDANF learns to capture intricate inter-variable associations, which are critical for accurate anomaly detection in complex multivariate datasets. As with many unsupervised methods, DDANF underscores its practicality by relying exclusively on normal data for training, aligning with real-world scenarios where labeled anomaly instances are sparse. This approach not only improves computational efficiency but also enhances the model's robustness in anomaly detection tasks across various domains. Overall, DDANF represents a significant advancement in multivariate time series anomaly detection, leveraging deep learning techniques to address key challenges and offering promising prospects for practical application in real-world settings.

The rest of this paper is organized as follows. Section 2 discusses related work on unsupervised time series anomaly detection. Section 3 is a detailed description of the proposed solution, and Section 4 presents the experiments results and detailed analysis. Finally, Section 5 concludes the paper.

2. Related work

The field of unsupervised time series anomaly detection can be systematically categorized into three distinct approaches based on their foundational implementation principles: representation and reconstruction methods, prediction-based methods, and probability distribution-based methods.

2.1. Based on representation and reconstruction

The method based on representation and reconstruction refers to learning the representation of normal and abnormal data or only the representation of normal data by optimizing the model, and then using it to reconstruct the input, judging anomalies based on the differences between representations and reconstruction errors. The key aspect of this type of approach is that the model is forced to capture some underlying patterns in the data that can be used as a discriminator of normal and abnormal. Reconstruction-based method is a widely used unsupervised time series anomaly detection setting, and there have been many methods designed in this lineup [6,11–17]. For example, USAD [6] employs a two-level autoencoders architecture and uses adversarial training to balance these two parts of the architecture. Its motivation is to reduce the reconstruction error of normal data while increasing the reconstruction error of anomalies, playing a minimax game between these objectives. Similar to USAD, TSAE [11] builds a two-stage autoencoders structure that decomposes time series into a long-term component and a short-term component, and then reconstructs each part separately. The reconstruction method based on autoencoders is characterized by its clear principle and simple structure. However, it is prone to the pitfalls of overfitting and mechanically memorizing data. Some methods are based on deep generative model, such as TAnoGAN [12], MadGAN [13], and MARU-GAN [15], these methods are inspired by either AnoGAN [18] or EGBAD [19]. They train a generative adversarial network on normal data, learning how to transform between two data domains. This transformation is later used to reconstruct samples, and data with large reconstruction errors are considered anomalous. However, these methods are also affected by unstable training and mode collapse.

Similarly, normalizing flow [20] is a popular generative model for simplifying complex data distributions into more manageable forms, such as the standard normal distribution, through a series of invertible and differentiable transformations. Normalizing flow has been effectively applied in various fields [21–23], including anomaly detection, where they excel at identifying deviations from normal patterns. Recent developments in anomaly detection leveraging normalizing flow have shown remarkable performance on industrial datasets. Approaches like Differnet [22] and CFLOW-AD [23] use deep learning to extract features and apply normalizing flow to estimate the likelihood of data, flagging anomalies based on statistical thresholds. However, these methods may overlook the structural nuances of the data. Incorporating autoencoders with normalizing flow overcomes this by establishing a mapping that captures the intrinsic structure of the data, thereby enhancing anomaly detection capabilities of the models.

2.2. Based on prediction

The prediction-based approach [24–27], typically utilizes sequential representation networks or autoregressive methods to forecast subsequent data, such as long short-term memory (LSTM) [24], gated recurrent units (GRU) [28], and transformer [29], etc. The network needs to learn and understand the temporal dependencies within a sequence length to make better prediction. The error between prediction and actual can be used to detect anomalies. Using LSTM to detect spacecraft anomalies was presented in [24], which modeled predictable normal events while distinguishing unanticipated anomalies, demonstrating the feasibility of LSTM in predicting telemetry

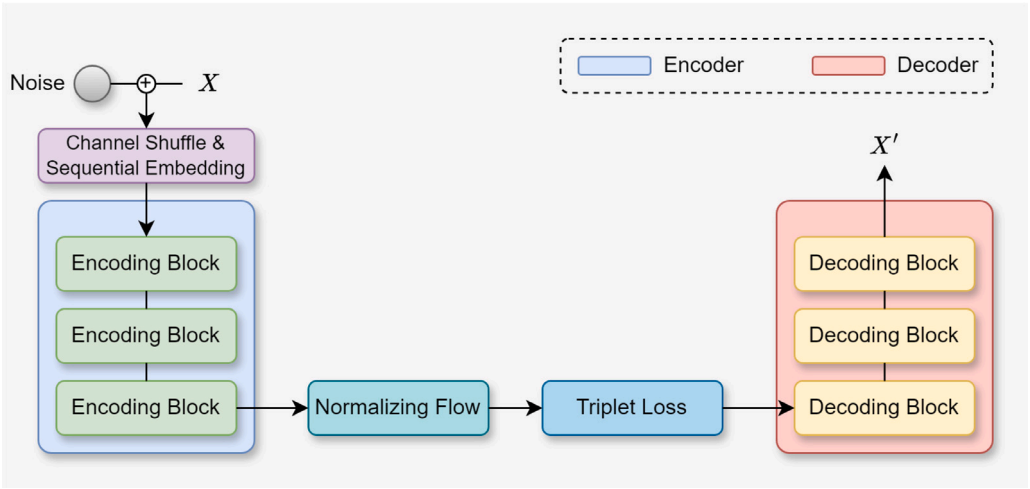


Fig. 1. The architecture of DDANF.

data of spacecraft. The first anomaly detection framework using future frame prediction network was proposed in [25], which also came with temporal and spatial constraints to achieve accurate prediction and detection. Due to the divergent objectives between time series forecasting and anomaly detection, models predicated on forecasting may be susceptible to the impact of such inconsistency.

2.3. Based on probability distribution

The goal of the probability distribution-based approach [30–32] is to model the probability distribution of time series. DAGMM [30] first maps the samples to a low-dimensional space, and then uses an estimation network to evaluate the energy of the samples in the framework of Gaussian mixture models [30]. By jointly optimizing the mapping model and the mixture model, the balance between reconstruction error and density estimation of latent representations is achieved, making the model less likely to fall into local optima. Motivated by real anomaly data and use cases commonly found in monitoring cloud services, a method for directly modeling the probability distribution of time series is proposed in [31] for detecting the status of network microservices and cloud resources, which can be used for anomaly detection in streaming data. Given the complexity of time series data, accurately modeling their data distribution is often quite challenging. Furthermore, as time progresses and the environment evolves, models typically require retraining to adapt to the new data distribution.

3. DDANF

We propose a multivariate time series anomaly detection method that effectively combines the differences between the original time series and the reconstructed sequence with the likelihood of sequence feature distribution. The proposed model comprises multiple components, primarily consisting of three modules: an encoder, a normalizing flow, and a decoder. Additionally, we add an auxiliary module before the encoder, namely, channel shuffling and sequential information embedding module. Specifically, the input is first added to the noise, which is intended to increase the difficulty of network training, achieve a certain regularization effect, and reduce overfitting. The added noise follows a Gaussian distribution with a mean of 0 and a variance of 0.5, and its dimensions are consistent with the dimensions of the model's input sequence. The input sequence, which is contaminated with noise, undergoes a random channel shuffling process and is then passed through the encoder along with additional embedding information in a sequential manner. The input sequence is subsequently processed through multiple encoding blocks, wherein each block extracts features from the preceding block's output. After passing through the final encoding block, the input data flow is given as output to a normalizing flow network immediately following the encoder. The normalizing flow is composed of multiple layers of affine coupled transformations, and its output is further fed into the decoder after being contracted by triplet loss. Next, the decoder will output the reconstructed time series. The overall loss function considers both the reconstruction loss of the initial input sequence and the output sequence at the low-level semantic level, as well as the likelihood of the sequence feature distribution at the high-level perceptual level. Finally, we calculate the anomaly score of the sequence through the flow loss and reconstruction error in the inference process.

The overall pipeline of the proposed method is illustrated in Fig. 1, and we will provide detailed explanations of these modules in the following sections. This section is organized as follows: we first define the problem to be addressed in Section 3.1, followed by a necessary introduction to normalizing flow in Section 3.2 for better comprehension of subsequent parts. Finally, in Section 3.3, we delve into the description of each module beyond the flow.

3.1. Problem statement

Formally, a time series can be represented as a collection of multiple data points,

$$\mathcal{T} = \{X_1, X_2, \ldots, X_T\}, X \in \mathbb{R}^m, \tag{1}$$

where each data point X_t represents an observation at time t with m features. Typically, a time series is constructed as a collection of data points ordered by their observation time, with equal intervals between each observation. For multivariate time series, $m > 1$, otherwise it is a univariate time series. In this paper, we mainly focus on multivariate time series. To simplify the description, we have $m > 1$ unless otherwise specified in the following context.

For unsupervised anomaly detection in time series, the object is to identify outlier data points that deviate significantly from the majority of observations without any prior labeled information. In the scenario where $\mathcal{T}$ serves as the training input, the model needs to fully utilize the characteristics of input data X_t, mine the temporal correlation and variables (or channels) association therein, and learn a well-reconstructed representation of the input. For input data $X_t, t \neq T$, the model should provide an anomaly scoring function $\mathcal{A}$, where $\mathcal{A}(normal) < \mathcal{A}(abnormal)$ for all normal and anomalous data.

Due to typically vast size of time series data in the real world, establishing complete temporal associations can be extremely challenging. In most existing time series anomaly detection algorithms, a sliding

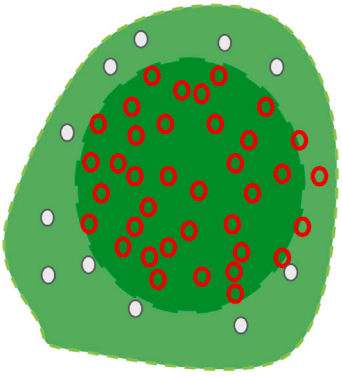


Fig. 2. Conceptual diagram of the latent space, from the unknown latent (light green) to the predefined latent space (dark green). The model restricts the encoding to the defined latent space as much as possible, as shown by the red circles in the figure. Points outside the latent space (white dots) will not be encoded and decoded accurately.

window approach is employed, whereby the time series is partitioned into windows of a certain step, so that a window can be expressed as

$$\mathcal{W}_t = \{X_{t-s+1}, \dots, X_{t-1}, X_t\}, \tag{2}$$

where s denotes the stride. The original time series is divided into different time windows, and anomaly detection is performed on a window-wise basis. To facilitate subsequent metric calculations, we assign a binary label $y \in \{0, 1\}$ to unseen windows based on the magnitude of their anomaly scores, with a label of 1 assigned to $\mathcal{W}_t$ that is determined to be anomalous, and a label of 0 assigned otherwise.

3.2. Normalizing flow

The encoder–decoder architecture is prone to the issue of the model mechanically memorizing input rather than truly understanding the characteristics of the input data, resulting in accurate reconstruction of anomalous data and leading to suboptimal results [33]. The unrestricted potential space is a significant factor behind this reason. Therefore, in this paper, we utilize a special generative model—normalizing flow [34], to transfer the unrestricted undefined latent space formed by the encoder into a restricted predefined latent space, as described in Fig. 2. In this way, a constraint is imposed on the potential threshold layer of the model such that it has a significant gap to normal and abnormal reconstructions.

The flow-based model is a type of deep generative model composed of a series of invertible transformations, which utilize the rule of change of variables to achieve reversible mapping from a complex distribution $\mathcal{X}$ to a simple distribution $\mathcal{Z}$. Formally, we denote f_θ as an invertible transformation function from $\mathcal{X} \in \mathbb{R}^d$ to $\mathcal{Z} \in \mathbb{R}^d$, which can be realized in the real world using affine coupling network. As such, we can obtain a combination of affine coupling layers:

$$f_\theta = f_L \circ f_{L-1} \circ \dots \circ f_1, \tag{3}$$

where θ represents the learnable parameters of the coupling layers, and L denotes the total number of layers. The d-dimensional input and output features of the normalizing flow model can be defined as $y_0 = x \in \mathcal{X}, y_L = z \in \mathcal{Z}$. Thus the latent variables can be expressed as $y_l = f_l(y_{l-1})$, and the remaining layers $\{y_l\}_{l=1}^{L-1}$ are considered as the output results of the intermediate latent layers. After obtaining the overall coupling transformation, we can estimate the distribution $p(x)$ of the initial input through a predefined prior distribution with the help of the change of variables:

$$p_x(x) = p_z(z)\left|\frac{dz}{dx}\right| = p_z(f(x))\left|\frac{df(x)}{dx}\right|, \tag{4}$$

taking the logarithm and adding multiple coupling layers,

$$\log p_\theta(x) = \log p_{\mathcal{Z}}(f_\theta(x)) + \sum_{l=1}^{L} \log\left|\det \mathcal{J}_{f_l}(y_{l-1})\right|, \tag{5}$$

the item $\mathcal{J}$ can be expressed as a Jacobian matrix under the condition of multi-layer transformation, where **det** represents the determinant of the matrix. The training objective of the normalizing flow is to continuously approximate the true target distribution $p(x)$ through $p_\theta(x)$. The set of parameters, θ, is obtained by optimizing the log-likelihood of $p(x)$ as follows:

$$\theta^* = \arg\min_{\theta\in\Theta} \mathbb{E}_{x\sim p_{\mathcal{X}}}\left[-\log p_\theta(x)\right]. \tag{6}$$

For the majority of real-world scenarios, it is appropriate to use a normal distribution (Gaussian distribution) for description. Similar to [35], we adopt the normal distribution $\mathcal{N}(\mu, \mathbf{\Sigma})$ as the defined prior distribution. Therefore, given that $p_{\mathcal{Z}}(z)$ follows a multivariate normal distribution, according to the formula (5), we can obtain:

$$\begin{aligned} p_{\mathcal{Z}}(z) =& (2\pi)^{-\frac{d}{2}} \det(\Sigma^{-\frac{1}{2}}) \exp(\\ & -\frac{1}{2}(z-\mu)^T \Sigma^{-1}(z-\mu)), \end{aligned} \tag{7}$$

where μ denotes the mean of the multivariate Gaussian distribution and Σ denotes the covariance. For simplicity, we further assume that the prior distribution obeys the standard Gaussian distribution $\mathcal{N}(0, \mathbf{I})$. Consequently, we can derive the following result according to formula (7):

$$p_{\mathcal{Z}}(z) = (2\pi)^{-\frac{d}{2}} \exp(-\frac{1}{2} z^T z). \tag{8}$$

Therefore, the optimization objective in (6) can be further expressed as:

$$\begin{aligned} \theta^* = \arg\min_{\theta\in\Theta} \mathbb{E}_{x\sim p_{\mathcal{X}}} \Big[& \frac{d}{2}\log(2\pi) + \frac{1}{2} f_\theta(x)^T f_\theta(x) - \\ & \sum_{l=1}^{L} \log\left|\det \mathcal{J}_{f_l}(y_{l-1})\right| \Big]. \end{aligned} \tag{9}$$

Ultimately, the maximum likelihood loss function used for optimizing the normalizing flow can be defined as the following equation:

$$\begin{aligned} \mathcal{L}_{flow} = \mathbb{E}_{x\sim p_{\mathcal{X}}} \Big[& \frac{d}{2}\log(2\pi) + \frac{1}{2} f_\theta(x)^T f_\theta(x) - \\ & \sum_{l=1}^{L} \log\left|\det \mathcal{J}_{f_l}(y_{l-1})\right| \Big]. \end{aligned} \tag{10}$$

Intuitively, when training the standardized flow solely on normal sequences, the encoding of normal sequences will be mapped to the high-density region of the prior Gaussian distribution, while the encoding of anomalous sequences will gravitate towards the low-density region due to their lack of exposure during training. As a result, the decoder will be unable to accurately reconstruct the input time series.

3.3. Module design of DDANF

Channel shuffle and embedding. Multivariate time series not only contain the association information of the temporal dimension, but also the association information of multiple features (or channels), which is a significant difference from univariate time series. Each feature or channel of a multivariate time series represents a unique series, i.e., a univariate time series. There may exist strong or weak dependency relationships between these time series, which cannot be discerned from the data itself. In terms of anomalies, the manifestation of an anomaly on one channel often affects other channels due to the existence of such dependencies. Therefore, if the model is able to recognize the differences between different channels as well as the potential correlations between channels, it will have additional evidence for more optimal anomaly detection. In contrast, assuming these channels

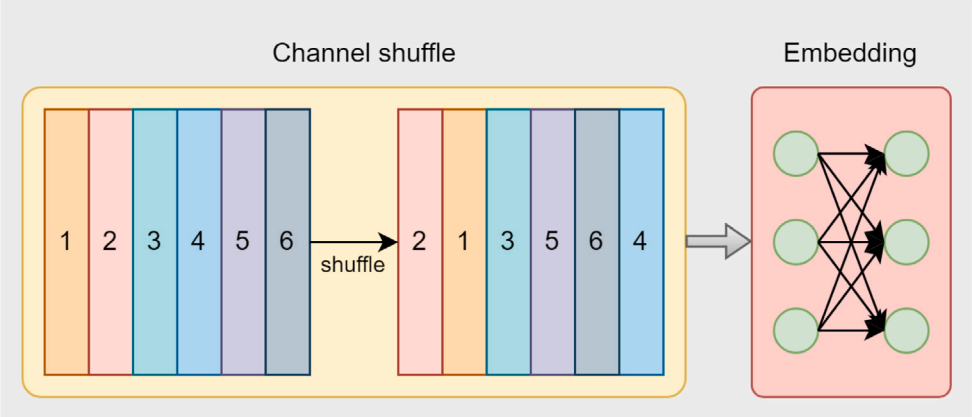


Fig. 3. The abstract concept diagram of channel shuffling and sequential information embedding. In the diagram, the numbers represent the indices of different channel positions, while the embedding layer is responsible for embedding the randomly shuffled index sequence into the input sequence.

are independent of each other will degrade the data into univariate data, which will prevent the model from leveraging the advantages of multivariate time series and hinder accurate anomaly detection. The model commences with a channel shuffling and embedding module, constructing data that incorporates channel information, which is subsequently fed into the encoder.

For the given multivariate time series, each variable is regarded as a channel, with the index of the variable serving as the channel's index. During the channel shuffling operation, the model randomly rearranges the indices of these channels. Concurrently, to enable the model to better recognize the relative positions between channels, we process the sequence of channel indices through an embedding layer, embedding the order information of the channels into the model's input. To be specific, as shown in Fig. 3, when the input sequence processed by noise enters the channel shuffling module, the channel of the input sequence will be randomly shuffled and reorganized to form a new sequence. Meanwhile, this sequence will be embedded with channel sequential information through an embedding layer network. Subsequently, the sequence will be feature-extracted in the encoder.

Encoder. The encoder is responsible for receiving input sequences containing channel information, extracting features through multiple layers of encoding blocks, and finally passing the resulting encoding to the normalizing flow. More specifically, the encoder of the model is composed of multiple stacked encoding blocks, with four blocks selected in this paper. Each encoding block first extracts information through convolutional operations, followed by batch normalization and activation layers, and finally undergoes the above operations again for downsampling. This is the standard process for one encoding block. Each time the sequence passes through an encoding block, its length decreases and the number of channels increases.

Implementation of normalizing flow. The architecture of the normalizing flow adopts an invertible neural network, and the affine coupling layer is the most important part of the flow architecture. In this paper, we use a Resnet-type [36] network structure as the coupling layer, and design a two-layer convolution, batch normalization, and activation process inside it, so as to realize a coupling layer sub-network. Finally, multiple sub-networks are stacked to form a normalizing flow that receives the output code from the encoder and outputs the latent code after a series of coupling transformations.

Decoder. The latent representations from normalizing flow will be contracted and converged under the influence of the triplet loss function. It further restricts the latent space of the normalizing flow, making it more likely that abnormal latent codes will fall outside the latent space, which in turn makes it more difficult for the decoder to accurately reconstruct these anomalies due to the fact that the model has only been trained and learned within the latent space. Therefore, the ability of the model to distinguish between normal and abnormal instances is further enhanced. The triplet loss learning method used in this paper is a self-supervised method for learning latent representations, which enables the threshold layer of the model to produce more discriminative embeddings. For each latent variable z, its positive counterpart z^+ is obtained by adding random noise while its negative counterpart z^- is obtained through randomly selecting an instance. The triplet loss function can then be formulated as follows:

$$\begin{aligned} \mathcal{L}_{triplet}(z) &= \max\left\{0, D^+ - D^- + \varepsilon\right\}, \\ D^+ &= Dist(z, z^+), \\ D^- &= \min\left\{Dist(z^+, z^-), Dist(z, z^-)\right\}, \end{aligned} \tag{11}$$

where ε represents the marginal parameter, and $Dist$ denotes the method used to calculate the distance between a pair of instances. The representation with triplet loss applied is fed to the decoder as input. The structure of the decoder is similar to that of the encoder but in reverse order, except that the decoder uses deconvolution.

Overall objective function. In the training phase, the overall loss function of the model consists of the flow loss $\mathcal{L}_{flow}$ as shown in formula (11), the reconstruction loss $\mathcal{L}_{rec}$ for the input and output sequences at the semantic level,

$$\mathcal{L}_{rec} = \|X - X'\|^2, \tag{12}$$

and the triplet loss $\mathcal{L}_{triplet}$ as shown in formula (10). The overall objective function is as follows:

$$\mathcal{L} = \alpha\mathcal{L}_{flow} + (1-\alpha)\mathcal{L}_{rec} + \mathcal{L}_{triplet}, \tag{13}$$

where α is a trade-off coefficient that controls the importance of different terms.

Anomaly score. The anomaly scoring function utilized in the inference stage consists of two components, with one being the mathematical expectation of log-likelihoods of flow. Specifically, the log likelihood $\log p$ under the standard Gaussian prior can be obtained according to formula (5), which can be converted to exponential form for the convenience of calculating the anomaly score. And since the anomaly score measures the likelihood of anomalies, the first term of the anomaly scoring function $\mathcal{A}$ is:

$$\mathcal{F}(x) = 1 - \exp(\log p(x)). \tag{14}$$

The other item is the reconstruction loss of the input sequence X and the output sequence X' at the semantic level, and in the inference stage we use the Dynamic Time Warping (DTW) approach to calculate the reconstruction loss. Ultimately, the anomaly scoring function has the following form:

$$\mathcal{A}(X) = \mathcal{F}(z) + DTW(X, X'). \tag{15}$$

Table 1
Quantitative results for different methods.

Datasets	Methods	Precision	Recall	F1 score
SWaT	FB	10.17	10.17	10.10
	KNN	12.28	11.75	12.01
	EncDec-AD (2016) [37]	11.40	68.25	19.53
	EGAN (2018) [19]	23.27	67.37	34.59
	SSL (2022) [38]	38.64	74.06	50.79
	LSTM-AD (2015) [39]	43.45	82.13	56.83
	MAD-GAN (2019) [13]	58.10	65.37	61.52
	MARU-GAN (2020) [15]	48.93	85.01	62.11
	IF (2008) [40]	72.43	59.05	65.06
	DAGMM (2018) [30]	80.18	67.29	73.17
	Ours	89.44	64.53	**74.97**
WADI	KNN	9.00	7.75	8.33
	FB	10.05	8.55	9.24
	EGAN (2018) [19]	11.27	52.92	18.58
	EncDec-AD (2016)[37]	11.40	68.25	19.53
	DAGMM (2018) [30]	22.28	19.76	20.94
	SSL (2022) [38]	99.34	14.95	25.99
	MAD-GAN (2019) [13]	22.78	59.13	32.89
	LSTM-VAE (2018) [41]	46.19	32.12	37.89
	OA (2019) [42]	26.52	97.99	41.74
	Ours	37.70	50.80	**43.28**

For these metrics, a higher value indicates a better performance. The best F1 score is shown in bold.
The aforementioned baselines are arranged in ascending order according to the F1-score.

4. Experiments and analysis

4.1. Datasets

Two public datasets are used in the experiments of this paper. The following is a detailed description of these datasets.

SWaT: SWaT is the Safe Water Treatment dataset [43], which is composed of data collected from 51 sensors on a scaled-down industrial water treatment testbed. This dataset provides continuous sensor observations for 11 days, with the first 7 days consisting of only normal observations, and the following 4 days injecting multiple anomalies into the data using various attack methods.

WADI: The WADI dataset, short for water distribution dataset [44], was obtained by running an urban water distribution system. The system was operated continuously for 16 days and collected observations from 123 sensors. During these days, data was collected under normal operating conditions for the first 14 days, while the remaining time included observations with anomalies.

4.2. Evaluation metrics

In this paper, we use precision (P), recall (R) and F1 score (F1) as metrics to evaluate the detection performance of the model:

$$P = \frac{TP}{TP + FP}, \tag{16}$$
$$R = \frac{TP}{TP + FN},$$
$$F1 = 2\frac{P \cdot R}{P + R},$$

where TP stands for true positive, FP for false positive, and FN for true negative. In the experiment, positive means the label is 1, otherwise the label is 0.

4.3. Overall performance

To assess the performance of the DDANF model, we compared it with several popular unsupervised anomaly detection methods. These methods include: the Isolated Forest (IF) approach [40]; K-nearest neighbors (KNN); Feature Bagging (FB); EncDec-AD [37]: an anomaly detection method that utilizes the encoder–decoder model; LSTM-AD [39]: anomaly detection using the LSTM structure; generative detection methods: Efficient GAN (EGAN) [19], MAD-GAN [13] and MARU-GAN [15]; DAGMM [30]: an anomaly detection method that utilizes a Gaussian mixture model; LSTM-VAE [41]: an anomaly detection method that combines LSTM and variational autoencoder; OmniAnomaly (OA) [42]: time series anomaly detection using random recurrent neural networks; BEATGAN [45]: a model detects anomalies using the adversarially generated time series; SSL [38]: a framework consists of two augmentation techniques in time series that capture two different patterns of original samples before feeding them to the classifier.

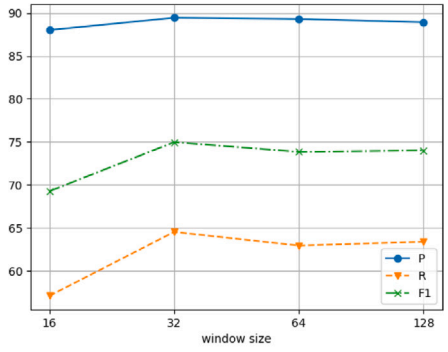
In our experiments, the window size was set to 32, and the number of encoding and decoding blocks (i.e., the number of model layers) was uniformly set to 3. Additionally, the batch size was set to 256, and the learning rate was set to 1.0×10^{-3}. The deep learning framework utilized for the experiment was PyTorch 2.0.0, employing the Adam [46] optimizer, and the experiments were conducted on two NVIDIA Tesla 16 GB GPUs. The detection results of these mainstream methods and our method are presented in Table 1. The first column of the table indicates the datasets on which these methods perform anomaly detection, while the second column lists the name of different detection methods. The rightmost three columns represent the P, R, and F1 score, respectively. We use F1 score as the final evaluation metric, which is consistent with the choice of most popular anomaly detection methods, and the F1 score of the best performing method is highlighted in bold. The results of these detection methods used as comparisons are obtained from [6,15] and the reproductions of the corresponding algorithms, and their average values are taken for fairness.

By analyzing the results in Table 1, it can be observed that traditional machine learning methods, such as KNN and FB, have relatively lower F1 score compared to other mainstream methods, and perform worse on the WADI dataset. In fact, the number of features in the WADI dataset is greater than that of the SWaT dataset, which suggests that one possible reason for this phenomenon is that these traditional methods are unable to adapt to time series with a larger number of channels or features. Furthermore, the table also shows that DDANF has better evaluation metrics compared to methods based on generative adversarial networks, such as MAD-GAN.

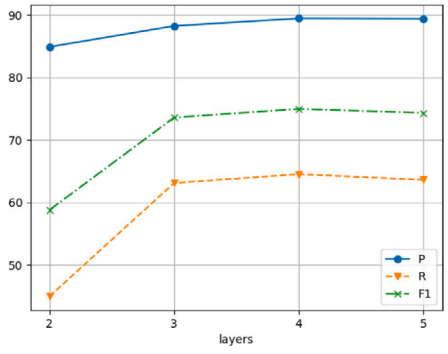
MAD-GAN searches for latent variables corresponding to subsequences of the input time series by multiple iterations during training so that the sequences generated from the latent variables are close enough to the input. Therefore, one reason for the inferior performance

Table 2
AUC-ROC metrics.

Methods	SWaT	WADI
USAD (2020) [6]	0.7903	–
SSL (2022) [38]	0.7301	0.5076
BEATGAN (2023)[45]	0.8192	0.5643
Ours	0.8784	0.7824



(a) window size



(b) model layers

Fig. 4. Effect of parameters.

of MAD-GAN is that the latent variables and generated sequences do not always correspond one-to-one, and the selected latent variables may not fully represent the inputs. The reason why DDANF outperforms other methods is that DDANF distinguishes and learns channel knowledge effectively, while the explicit channels shuffling mechanism serves as a form of regularization. Additionally, by imposing strict constraints on the latent space of the model, shortcuts such as directly copying inputs are avoided, preventing complete reconstruction of anomalies.

In addition to commonly used evaluation metrics such as Precision, Recall, and F1 score, we also report in Table 2 the area under curve (AUC) of DDANF and three anomaly detection models within the past three years, which is the area enclosed with the coordinate axis under the receiver operating characteristic (ROC) curve.

As time series anomaly detection can be considered as an extremely imbalanced classification problem, F1 score is easily influenced by imbalanced data samples and cannot accurately reflect the performance of the method, while AUC is robust to imbalanced samples. Moreover, compared with threshold-independent metrics like AUC, F1 score is more dependent on the selection of classification thresholds. However, for the real-world problem of time series anomaly detection, finding a suitable threshold is not easy and does not reflect actual situations, requiring some expert experience and an appropriately sized dataset. As can be seen in Table 2, our method outperforms recent time series anomaly detection methods.

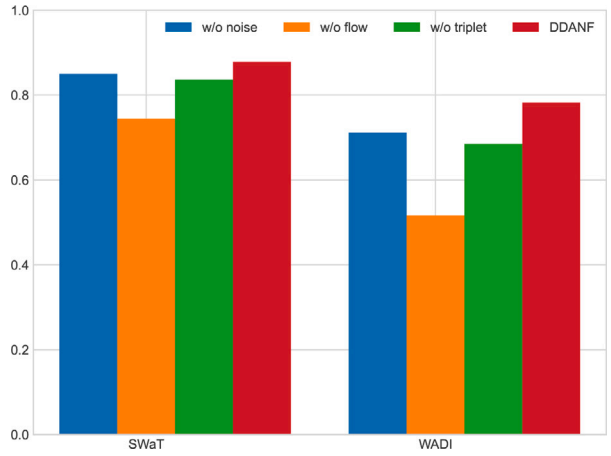


Fig. 5. Impact of different modules.

4.4. Model analysis

Dividing the complete time series into different subsequences using a sliding window is a commonly used approach in unsupervised time series anomaly detection methods, which is also adopted in this paper. Therefore, the selection of sliding window size is crucial for anomaly detection. A small sliding window may cause problems such as model non-convergence and unstable detection, as the model cannot learn the temporal dependencies of the complete time series through short subsequences. In contrast, a large window usually achieves better detection performance, but a larger sliding window is not always better. A window that is too large would lose its practical application significance in real-world scenarios, as it requires collecting enough data to form a window after the attack signal is emitted before detection can be performed. The best results that DDANF can achieve with four exponentially growing windows are shown in Fig. 4(a), with window sizes s taken as 16, 32, 64 and 128. It can be observed that the detection performance no longer increases with increasing window size beyond 32, so the model is robust to windows of different sizes. A sliding window of size 32 was chosen for our experiments.

Another concern of our investigation is the effect of the number of layers of the encoder and decoder on the detection results. The encoder and decoder of the model are generally symmetric, i.e., the number of layers of the encoder and the number of layers of the decoder are equal. When there are fewer layers, the number of model parameters is also reduced, allowing the model to complete detection more quickly. However, the layers of the encoding block affects the extraction of sequence information features by the model, so too few layers may result in the encoder failing to extract effective information. As the number of layers in the model increases, the number of model parameters also increases, resulting in longer detection times and the extraction of redundant information. Additionally, increasing the number of layers in the model may lead to slow convergence due to gradient vanishing.

Fig. 4(b) summarizes the results obtained by DDANF using four different layer depths (2, 3, 4, 5). The results show that a stable performance can be maintained when the number of layers in the model is greater than or equal to 3, and adding more layers is futile.

To validate the role of each module within the DDANF, we conducted a detailed ablation study on the same dataset as previously discussed. As depicted in Fig. 5, the horizontal axis of the figure represents the different datasets and the vertical axis represents the AOC values obtained for the different model variants. We established three variants based on the DDANF: w/o noise, w/o flow, and w/o triplet. These variants represent the model without added noise, the model without the normalization flow module, and the model without the triplet loss, respectively, with all other parameters remaining constant.

It can be observed from Fig. 5 that the removal of any module from the model leads to a degree of performance degradation. Notably, among the three different model variants, the degradation caused by w/o flow is significantly more severe than the other two, revealing the pivotal role of the normalization flow in the DDANF model.

5. Conclusions

In this study, for the multivariate time series anomaly detection problem, we propose Deep Denosing Autoencoder Normalizing Flow (DDANF), an unsupervised anomaly detection method based on denoising autoencoder and normalizing flow. In this method, we introduce a channel shuffle and sequential information embedding module to explicitly make the model aware of the existence of channels and learn the differences and complex dependencies between different channels. Deep sequence features are extracted by an encoder and tight restrictions are imposed on the latent space of sequences using normalizing flow and triplet loss, making the generation and reconstruction of the decoder more purposeful. The model is shown to be robust to different parameters through analysis of the model. Our experimental results on two large public datasets show that the proposed method DDANF outperforms other mainstream unsupervised anomaly detection methods, confirming the effectiveness of the method.

For subsequent research endeavors, several directions can be considered: Firstly, existing detection methods largely rely on fixed-length time series data as input. However, in practical applications, it may not be feasible to ensure the availability of time series data of fixed lengths. Expanding anomaly detection algorithms to naturally accommodate variable-length inputs is a research direction worthy of exploration. Secondly, deep learning techniques are highly data-dependent. The current scale of time series anomaly detection datasets and the types of anomalies are somewhat lacking, potentially failing to reveal the true performance of algorithms. Future research could focus more on enhancing the quality of datasets. Thirdly, due to the rarity of anomalies, time series anomaly detection is an extremely imbalanced problem. The commonly used F1-score metric is not particularly well-suited for such issues. Yet, beyond this, there is no widely accepted unified metric. To accurately measure the performance of algorithms, there is a need to explore more robust and targeted evaluation metrics.

CRediT authorship contribution statement

Xigang Zhao: Writing – original draft, Formal analysis, Conceptualization. **Peng Liu:** Writing – original draft, Methodology, Investigation, Formal analysis. **Saïd Mahmoudi:** Validation, Methodology. **Sahil Garg:** Writing – review & editing, Resources, Project administration. **Georges Kaddoum:** Writing – review & editing, Validation, Project administration. **Mohammad Mehedi Hassan:** Writing – review & editing, Supervision, Resources, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by National Natural Science Foundation of China (No. 62172134). The authors are grateful to King Saud University, Riyadh, Saudi Arabia for funding this work through Researchers Supporting Project Number (RSP2024R18).

References

[1] M. Gupta, J. Gao, C.C. Aggarwal, J. Han, Outlier detection for temporal data: A survey, IEEE Trans. Knowl. Data Eng. 26 (9) (2013) 2250–2267.
[2] M.M. Breunig, H.-P. Kriegel, R.T. Ng, J. Sander, LOF: identifying density-based local outliers, in: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, 2000, pp. 93–104.
[3] W.A. Chaovalitwongse, Y.-J. Fan, R.C. Sachdeo, On the time series k-nearest neighbor classification of abnormal brain activity, IEEE Trans. Syst. Man Cybern. A 37 (6) (2007) 1005–1016.
[4] J. Ma, S. Perkins, Time-series novelty detection using one-class support vector machines, in: Proceedings of the International Joint Conference on Neural Networks, 2003., Vol. 3, IEEE, 2003, pp. 1741–1745.
[5] S. Baek, D. Kwon, S.C. Suh, H. Kim, I. Kim, J. Kim, Clustering-based label estimation for network anomaly detection, Digit. Commun. Netw. 7 (1) (2021) 37–44.
[6] J. Audibert, P. Michiardi, F. Guyard, S. Marti, M.A. Zuluaga, Usad: Unsupervised anomaly detection on multivariate time series, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 3395–3404.
[7] P. Wei, B. Wang, X. Dai, L. Li, F. He, A novel intrusion detection model for the CAN bus packet of in-vehicle network based on attention mechanism and autoencoder, Digit. Commun. Netw. 9 (1) (2023) 14–21.
[8] A. Oussidi, A. Elhassouny, Deep generative models: Survey, in: 2018 International Conference on Intelligent Systems and Computer Vision, ISCV, IEEE, 2018, pp. 1–8.
[9] G. Pang, C. Shen, L. Cao, A.V.D. Hengel, Deep learning for anomaly detection: A review, ACM Comput. Surv. (CSUR) 54 (2) (2021) 1–38.
[10] H. Li, Y. Li, Anomaly detection methods based on GAN: a survey, Appl. Intell. 53 (7) (2023) 8209–8231.
[11] S. Naito, Y. Taguchi, K. Nakata, Y. Kato, Anomaly detection for multivariate time series on large-scale fluid handling plant using two-stage autoencoder, in: 2021 International Conference on Data Mining Workshops, ICDMW, IEEE, 2021, pp. 542–551.
[12] M.A. Bashar, R. Nayak, Tanogan: Time series anomaly detection with generative adversarial networks, in: 2020 IEEE Symposium Series on Computational Intelligence, SSCI, IEEE, 2020, pp. 1778–1785.
[13] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, S.-K. Ng, MAD-gan: Multivariate anomaly detection for time series data with generative adversarial networks, in: Artificial Neural Networks and Machine Learning–ICANN 2019: Text and Time Series: 28th International Conference on Artificial Neural Networks, Munich, Germany, September 17–19, 2019, Proceedings, Part IV, Springer, 2019, pp. 703–716.
[14] Y. Li, X. Peng, J. Zhang, Z. Li, M. Wen, DCT-gan: dilated convolutional transformer-based gan for time series anomaly detection, IEEE Trans. Knowl. Data Eng. (2021).
[15] C. Maru, I. Kobayashi, Collective anomaly detection for multivariate data using generative adversarial networks, in: 2020 International Conference on Computational Science and Computational Intelligence, CSCI, IEEE, 2020, pp. 598–604.
[16] Y. Yang, S. Ding, Y. Liu, S. Meng, X. Chi, R. Ma, C. Yan, Fast wireless sensor for anomaly detection based on data stream in an edge-computing-enabled smart greenhouse, Digit. Commun. Netw. 8 (4) (2022) 498–507.
[17] B. Weinger, J. Kim, A. Sim, M. Nakashima, N. Moustafa, K.J. Wu, Enhancing IoT anomaly detection performance for federated learning, Digit. Commun. Netw. 8 (3) (2022) 314–323.
[18] T. Schlegl, P. Seeböck, S.M. Waldstein, U. Schmidt-Erfurth, G. Langs, Unsupervised anomaly detection with generative adversarial networks to guide marker discovery, in: Information Processing in Medical Imaging: 25th International Conference, IPMI 2017, Boone, NC, USA, June 25-30, 2017, Proceedings, Springer, 2017, pp. 146–157.
[19] H. Zenati, C.S. Foo, B. Lecouat, G. Manek, V.R. Chandrasekhar, Efficient gan-based anomaly detection, 2018, arXiv preprint arXiv:1802.06222.
[20] D.P. Kingma, P. Dhariwal, Glow: Generative flow with invertible 1x1 convolutions, Adv. Neural Inf. Process. Syst. 31 (2018).
[21] G. Papamakarios, E. Nalisnick, D.J. Rezende, S. Mohamed, B. Lakshminarayanan, Normalizing flows for probabilistic modeling and inference, J. Mach. Learn. Res. 22 (57) (2021) 1–64.
[22] M. Rudolph, B. Wandt, B. Rosenhahn, Same same but differnet: Semi-supervised defect detection with normalizing flows, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2021, pp. 1907–1916.
[23] D. Gudovskiy, S. Ishizaka, K. Kozuka, Cflow-ad: Real-time unsupervised anomaly detection with localization via conditional normalizing flows, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2022, pp. 98–107.
[24] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, T. Soderstrom, Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 387–395.
[25] W. Liu, W. Luo, D. Lian, S. Gao, Future frame prediction for anomaly detection–a new baseline, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 6536–6545.

[26] Z. Guo, K. Yu, N. Kumar, W. Wei, S. Mumtaz, M. Guizani, Deep-distributed-learning-based POI recommendation under mobile-edge networks, IEEE Internet Things J. 10 (1) (2022) 303–317.
[27] J. Pan, N. Ye, H. Yu, T. Hong, S. Al-Rubaye, S. Mumtaz, A. Al-Dulaimi, I. Chih-Lin, AI-driven blind signature classification for IoT connectivity: A deep learning approach, IEEE Trans. Wireless Commun. 21 (8) (2022) 6033–6047.
[28] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling, 2014, arXiv preprint arXiv:1412.3555.
[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).
[30] B. Zong, Q. Song, M.R. Min, W. Cheng, C. Lumezanu, D. Cho, H. Chen, Deep autoencoding gaussian mixture model for unsupervised anomaly detection, in: International Conference on Learning Representations, 2018.
[31] F. Ayed, L. Stella, T. Januschowski, J. Gasthaus, Anomaly detection at scale: The case for deep distributional time series models, in: Service-Oriented Computing–ICSOC 2020 Workshops: AIOps, CFTIC, STRAPS, AI-PA, AI-IOTS, and Satellite Events, Dubai, United Arab Emirates, December 14–17, 2020, Proceedings, Springer, 2021, pp. 97–109.
[32] Y. Xiao, K. Xia, H. Yin, Y.-D. Zhang, Z. Qian, Z. Liu, Y. Liang, X. Li, AFSTGCN: Prediction for multivariate time series using an adaptive fused spatial-temporal graph convolutional network, Digit. Commun. Netw. (2022) http://dx.doi.org/10.1016/j.dcan.2022.06.019.
[33] A. Tong, G. Wolf, S. Krishnaswamyt, Fixing bias in reconstruction-based anomaly detection with lipschitz discriminators, in: 2020 IEEE 30th International Workshop on Machine Learning for Signal Processing, MLSP, IEEE, 2020, pp. 1–6.
[34] L. Dinh, D. Krueger, Y. Bengio, Nice: Non-linear independent components estimation, 2014, arXiv preprint arXiv:1410.8516.
[35] Y. Zhao, Q. Ding, X. Zhang, AE-FLOW: Autoencoders with normalizing flows for medical images anomaly detection, in: The Eleventh International Conference on Learning Representations, 2023.
[36] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[37] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, G. Shroff, LSTM-based encoder-decoder for multi-sensor anomaly detection, 2016, arXiv preprint arXiv:1607.00148.
[38] D.H. Tran, V.L. Nguyen, H. Nguyen, Y.M. Jang, Self-supervised learning for time-series anomaly detection in industrial internet of things, Electronics 11 (14) (2022) 2146.
[39] P. Malhotra, L. Vig, G. Shroff, P. Agarwal, et al., Long short term memory networks for anomaly detection in time series, in: ESANN, Vol. 2015, 2015, p. 89.
[40] F.T. Liu, K.M. Ting, Z.-H. Zhou, Isolation forest, in: 2008 Eighth Ieee International Conference on Data Mining, IEEE, 2008, pp. 413–422.
[41] D. Park, Y. Hoshi, C.C. Kemp, A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder, IEEE Robot. Autom. Lett. 3 (3) (2018) 1544–1551.
[42] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, D. Pei, 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, 2019, pp. 2828–2837.
[43] J. Goh, S. Adepu, K.N. Junejo, A. Mathur, A dataset to support research in the design of secure water treatment systems, in: Critical Information Infrastructures Security: 11th International Conference, CRITIS 2016, Paris, France, October 10–12, 2016, Revised Selected Papers 11, Springer, 2017, pp. 88–99.
[44] C.M. Ahmed, V.R. Palleti, A.P. Mathur, 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, 2017, pp. 25–28.
[45] S. Liu, B. Zhou, Q. Ding, B. Hooi, Z. Zhang, H. Shen, X. Cheng, Time series anomaly detection with adversarial reconstruction networks, IEEE Trans. Knowl. Data Eng. 35 (4) (2022) 4293–4306.
[46] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, arXiv preprint arXiv:1412.6980.