IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 6, 2021, accepted August 23, 2021, date of publication August 26, 2021, date of current version September 7, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3107975

Deep Learning for Anomaly Detection in Time-Series Data: Review, Analysis, and Guidelines

KUKJIN CHOI[1,2], JIHUN YI[1], CHANGHWA PARK[1,3], AND SUNGROH YOON[1], (Senior Member, IEEE)

[1]Department of Electrical and Computer Engineering, Seoul National University, Seoul 08826, South Korea
[2]DIT Center, Samsung Electronics, Hwaseong-si 18448, South Korea
[3]AIRS Company, Hyundai Motor Group, Seoul 06797, South Korea

Corresponding author: Sungroh Yoon (sryoon@snu.ac.kr)

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) [2018R1A2B3001628], and Samsung Electronics.

ABSTRACT As industries become automated and connectivity technologies advance, a wide range of systems continues to generate massive amounts of data. Many approaches have been proposed to extract principal indicators from the vast sea of data to represent the entire system state. Detecting anomalies using these indicators on time prevent potential accidents and economic losses. Anomaly detection in multivariate time series data poses a particular challenge because it requires simultaneous consideration of temporal dependencies and relationships between variables. Recent deep learning-based works have made impressive progress in this field. They are highly capable of learning representations of the large-scaled sequences in an unsupervised manner and identifying anomalies from the data. However, most of them are highly specific to the individual use case and thus require domain knowledge for appropriate deployment. This review provides a background on anomaly detection in time-series data and reviews the latest applications in the real world. Also, we comparatively analyze state-of-the-art deep-anomaly-detection models for time series with several benchmark datasets. Finally, we offer guidelines for appropriate model selection and training strategy for deep learning-based time series anomaly detection.

INDEX TERMS Anomaly detection, deep learning, fault diagnosis, industry applications, Internet-of-Things (IoT), time series analysis.

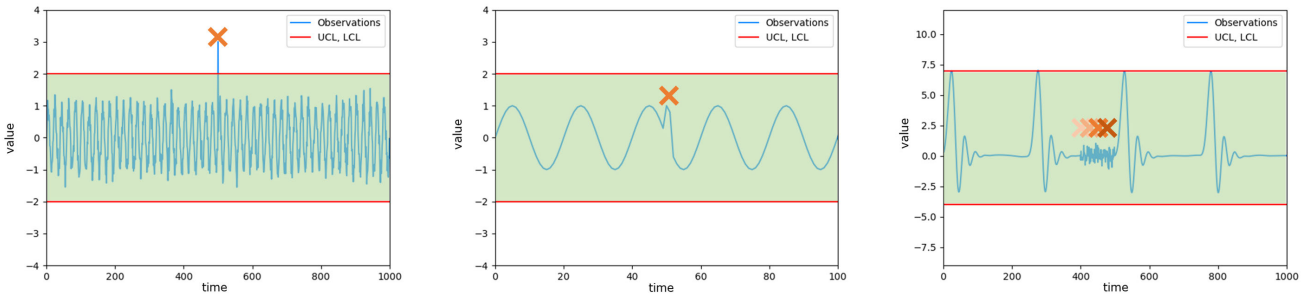
I. INTRODUCTION

Everything on the Earth is a source of signals. Humans have continuously measured and collected signals occurring in nature, such as temperature, wind speed, rainfall, and sunspot intensity, to adapt to the environment. In addition, for decades, various industrial activities have been generating numerous data in most fields of industries such as business (e.g., sales and market trend), finance (e.g., stock price), biomedical (e.g., heart and brain activity), and manufacturing (e.g., yield). In each industrial field, the data owners actively collect and leverage them to improve products, processes, and services. In particular, with the advent of Industry 4.0, industries have started to intensively utilize numerous sensors to monitor their facilities and systems simultaneously, resulting in increased efficiency, safety, and security [1].

Among the various data types, time-series data has been studied for a long time in academia, such as medicine, meteorology, and economics, and is now an essential target of analysis in most practical applications. Time-series analysis refers to a range of tasks that aim to extract meaningful knowledge from time-ordered data; the extracted knowledge can be used not only to diagnose the past behavior but also to predict the future. Widely-known examples of time-series analysis include classification, clustering, forecasting, and anomaly detection.

Anomaly detection, the process of identifying unexpected items or events from data, has become a field of interest for many researchers and practitioners and is now one of the main tasks in data mining and quality assurance [2]. It has been studied in a variety of application domains and has experienced significant progress. Classical methods including linear model-based methods [3], distance-based methods [4], density-based methods [5], and support vector machines [6],

The associate editor coordinating the review of this manuscript and approving it for publication was Chuan Zhou.



(a) point anomaly (b) contextual anomaly (c) collective anomaly

FIGURE 1. **Anomaly types in time-series data.**

are still a viable choice of algorithm. However, as target systems become larger and more complex, those methods face limitations, namely an inability to manipulate multi-dimensional data or address a shortage of labeled anomalies. In particular, detecting anomalies in time-series data is challenging because the order and the causality between observations along the time axis need to be jointly considered. Recently, many approaches have been developed to address these challenges. For instance, Hu *et al.* [7] proposed a novel computational method using a recurrence plot (RP), a square matrix consisting of the times at which a state of a dynamic system recurs. They measure the local recurrence rates (LREC) by scanning the RP with a sliding window and detect anomalies by comparing similarities between the statistics of the LREC curves.

Deep learning, a subfield of machine learning algorithms inspired by the structure and function of the brain, has been getting attention in recent years. Deep-learning methods learn the complex dynamics in the data, while making no assumptions about the underlying patterns within the data. This property makes them the most attractive choice for time-series analysis these days. For instance, Yan *et al.* [8] proposed to combine ensembled long short term memory (LSTM) neural networks, which memorize long term patterns in time series, with the stationary wavelet transform (SWT), to forecast the energy consumption. Their experimental results showed that the proposed deep-learning method outperforms classical computational methods.

The goal of this study is to review state-of-the-art deep learning-based anomaly detection methods for time-series data. To the best of our knowledge, previous reviews [1], [2], [9]–[14] on this subject matter do no more than simply categorize models according to their mechanisms and describe their characteristics. In this paper, in addition to classifying the models according to their methodologies, we further analyze in detail how they define interrelationships between variables, learn the temporal context, and identify anomalies in multivariate time series. Also, we provide guidelines to practitioners based on comparative experimental analyses using several benchmark datasets. Our analyses provide practitioners with helpful insights for choosing the best-suited method(s) for the problem(s) they are trying to solve.

The rest of the paper is organized as follows: in Section II, we provide elementary backgrounds on anomaly detection and time series. In Section III, we present various industrial use cases. In Section IV, we present notable conventional methods and discuss the underlying factors that have made them no longer sufficient for recent applications. In Section V, we review recent anomaly detection methods in-depth according to how they define the inter-correlations between variables, model the temporal context, and set anomaly criteria. Through Section VI-A to VI-B, we evaluate the deep learning-based anomaly detection methods on several benchmark datasets and provide a comparative review. Finally in Section VII, we provide general guidelines for model selection to fit given conditions and problems.

II. BACKGROUND

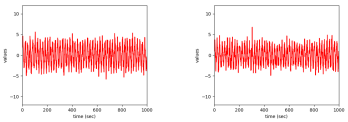
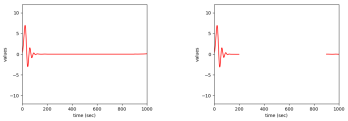
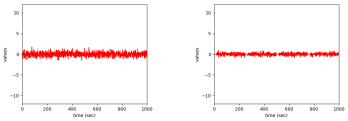
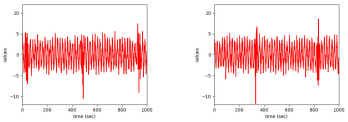
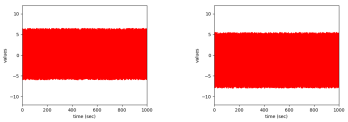
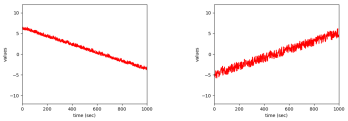
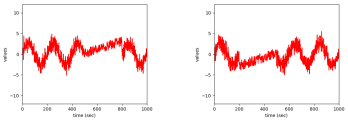
A. ANOMALIES IN TIME-SERIES DATA

We begin with introductory remarks on the definition of anomalies. Several attempts have been made to describe the nature of anomalous data (i.e., statistical outliers). Hawkins [15] described an outlier as an observation that deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism. In this context, we can describe the anomaly in time-series data as the data point(s) at time step(s) that shows unexpected behaviors that differ significantly from previous time steps. Following the previous works of literature, we categorize the types of anomalies related to time-series data as follows.

1) POINT ANOMALY

Point anomaly is a data point or a sequence that abruptly deviates from the norm (Fig. 1(a)). Such anomalies may appear to be temporal noise and are often caused by sensor errors or abnormal system operations. For detection, operators traditionally set upper and lower control limits, commonly referred to as UCL and LCL, respectively, based on prior data. Values that exist outside those limits are regarded as point anomalies.

TABLE 1. Description of the detailed classification of anomalies in time-series data.

Anomaly patterns	Description	Examples
Normal (assumption)	The amplitude and frequency are stable over time steps, and the time response is symmetrical	
Missing	Most/all of the data are missing, and the time/frequency response becomes 0	
Minor	Compared to normal sensor data, the vibration amplitude is very small	
Outlier	One or more outliers appear in the time response	
Square	The time response oscillates within a limiting range like a square wave	
Trend	The data has an obvious non-stationary and monotonous trend	
Drift	The vibration response is nonstationary, with random drift	

2) CONTEXTUAL ANOMALY

Similar to a point anomaly, a contextual anomaly represents a data point or sequence observed over a short time but does not deviate from the normal range in the same way as predefined UCL- and LCL-delimited anomalies. However, considering the given context (Fig. 1(b)), the data points are out of the expected pattern or shape. For this reason, these anomalies can be difficult to detect.

3) COLLECTIVE ANOMALY

This type of anomaly refers to a set of data points that should be considered an anomaly because they gradually show a different pattern from normal data over time (Fig. 1(c)). Individual values within this type of anomaly may seem trouble-free, but collectively, they raise suspicion. Since they are not easily recognizable at once, contexts over the long term are of particular importance in detecting them.

4) OTHER ANOMALY TYPES

Since anomaly is something outside the normal state, what is abnormal depends on what we define to be normal. Generally speaking, anomalies can be classified into one of the three aforementioned types, but other perspectives may subdivide anomalies into more specific categories. Table 1 shows the taxonomy of anomaly patterns and their examples described from [16], [17].

In summary, an anomaly is a data point whose occurrence was either extremely rare in the past or is logically impossible. However, in multivariate time-series data, it may not be valid to classify anomalies as in the previous examples. Multivariate time-series data require additional consideration of the relationship between variables along with the time axis. As the number of variables increases, more diversified patterns occur. Then, an abnormal pattern may be irregular, and the difference between normal and abnormal state

may be ambiguous. Scanning the individual univariate time-series data and aggregating them to identify anomalies do not guarantee the accuracy of detection results because few anomaly points can be obscured by the other normal variables and significantly affect the entire target system. Reducing the dimensions by extracting clear variables or features or using a model complex enough to detect various patterns can address such problems.

B. PROPERTIES OF TIME-SERIES DATA

Although time is an essential concept in nearly all tasks, working with time-sensitive data requires lots of careful consideration. Nevertheless, if the characteristics of time-series data are well-understood, anomalies can be effectively detected by utilizing the contextual information from signals. Therefore, we describe the fundamentals of time-series data in a nutshell. The factors discussed here include temporality, dimensionality, nonstationarity, and noise.

1) TEMPORALITY

A time series is generally considered to be a collection of observations indexed in a time order [18]. The data are captured at equal intervals, and each successive data point in the series depends on its past values. Hence, there is some implication of the temporal correlation or dependence between each consecutive observation [19]. A joint distribution of sequence of observations can be expressed using the chaining product rule as (1).

$$p(x^1, x^2, \ldots, x^T) = p(x^1) \prod_{t=2}^{T} p(x^t | x^1, x^2, \ldots, x^{t-1}), \quad (1)$$

where x^t is a data point observed at time $t \in T \subseteq \mathbb{Z}^+$ and each conditional probability $p(\cdot|\cdot)$ indicates the temporal dependence between current state and previous ones.

2) DIMENSIONALITY

Dimensionality refers to the number of individual data attributes captured in each observation [9]. According to the dimensionality, time-series data is largely divided into univariate and multivariate types. The dimensionality of time-series data influence computational costs and analysis-method choices.

- *Univariate:* This type describes an ordered set of real-valued observations, where each data point is measured at a specific time, $t \in T \subseteq \mathbb{Z}^+$. Then, $x^t \in \mathbb{R}$ is a data point measured at time t and is a realized value of a certain random variable, X^t [2].
- *Multivariate:* This type describes an ordered set of multidimensional vectors, $X = \{\mathbf{x}^t\}_{t \in T}$, each of which is recorded at a specific time, $t \in T \subseteq \mathbb{Z}^+$, and contains real-valued observations. In practical circumstances, this can be seen as a group of univariate time-series data streams representing the state of the target system.

Anomaly detection for univariate time series only considers the relations between the current state and the previous states, i.e., temporal dependence. But for a multivariate stream, both the temporal dependence and the correlations between observations should be considered. Despite the added trickiness, multivariate time series data has now become a typical type of data for analyzing various behaviors created by combinations of several variables.

3) NONSTATIONARITY

A time series is said to be stationary if its statistical properties do not change over time. More explicitly, for any $\tau \in \mathbb{N}$, a continuous stochastic process $\mathbf{x} = \{x^t\}_{t \in T \subset \mathbb{Z}^+}$ is strongly stationary if following condition is satisfied, as in (2).

$$F_{\mathbf{x}}(x^{1+\tau}, \ldots, x^{t+\tau}) = F_{\mathbf{x}}(x^1, \ldots, x^t), \quad (2)$$

where $F_{\mathbf{x}}$ denotes the joint distribution function. Ideally, we want a stationary time series for modeling, but many of the desired properties are not satisfied in real-world scenarios. Volatile features, such as seasonality, concept drift, and change points, make time-series data non-stationary.

- *Seasonality:* This refers to a periodic and recurrent pattern caused by factors such as weather, holidays, marketing promotions, and the behaviors of economic agents [20]. In short, it is a periodic fluctuation over a limited time scale. For example, power consumption is high during the day and low during the night. Likewise, online sales increase rapidly over the Black Friday weekend and then decrease again.
- *Concept Drift:* The nonstationarity of many real environments may lead to changes in the underlying statistical distribution of a data stream over time. This phenomenon goes by many names in literature, the most common of which is *concept drift* [21]. This is a central issue, because it can derail the performance of models learned from historical data [22].
- *Change Points:* In the manufacturing industry, the normal state of equipment often changes for several reasons. For instance, process conditions change as operations are stopped and restarted with a different setting.

Because most time-series data are nonstationary, data points that indicate spurious anomalies at certain timestamps may not be truly anomalous on a larger scale. Hence, detection methods that adapt to changes in data structures are required for long-term deployment.

4) NOISE

In signal processing, noise is a general term for unwanted changes to signals during their capture, storage, transmission, processing, or conversion [23]. It is considered a bread-and-butter issue in real-world systems. In many cases, noise is due to minor fluctuations in the sensor sensitivity and will have essentially no effect on the overall data structure. However, when the separation between noise and anomaly in a noisy system is difficult, noise seriously affects the performance of detection models [24]. Therefore, it is crucial to understand the nature of the noise and reduce noise during the preprocessing stage.

III. INDUSTRIAL APPLICATIONS

Various industries have increased their competitiveness by adapting to the changing environment using the latest digital technology. Cloud computing, big data, mobile devices, IoT, and artificial intelligence (AI) have led to the hyper-connectivity and super-intelligence of industrial sites. Combining digital components with physical world phenomena helps reduce operating costs, increase business agility and flexibility, and create new revenue models. Anomaly detection using these technologies is particularly essential to industry because it is highly demanded by real-world applications, such as fault detection in manufacturing, leak detection in gas-chemical processes, cyber intrusion detection, and structural health monitoring in infrastructures.

(a) etching machine[1] (b) CCSS[2] (c) AGV[3]

FIGURE 2. The examples by equipment type: (a) A production equipment named etching machine in semiconductor manufacturing creates chip features by selectively removing dielectric and metal materials on a wafer; (b) An infrastructure facility called the central chemical supply system safely supplies high-purity chemicals to the semiconductor manufacturing process; and (c) A logistics automation equipment called automated guided vehicle transports product components in work areas.

A. SMART MANUFACTURING

The idea of *smart factory* conceptualizes a highly digitalized and connected combination of facilities and equipment that can improve productivity and quality through automation and self-optimization. In an automated manufacturing process, equipment conditions are most closely related to quality and productivity. Stable operation leads to better quality, and efficient operation reduces manufacturing time and improves productivity. Therefore, it is crucial to detect faults immediately or forecast possible anomalies in equipment.

The equipment applied in smart factories includes the production equipment, the infrastructure facility, and the logistics automation equipment (Fig. 2). The production equipment manufactures products efficiently while maintaining quality. The infrastructure facility supplies power, water, gas, and chemicals to the manufacturing process; it also purifies wastewater and chemical waste. The logistics automation equipment carries products from one place to another.

While several machine learning techniques have been utilized to detect damage, faults, and abnormalities in these types of industrial equipment [25]–[28], deep-learning models have shown a great promise.

1) PRODUCTION EQUIPMENT

Data-driven models help equipment operation in large manufacturing factories because they can detect possible failures without extensive domain knowledge. Hsieh *et al.* [29] adopted an autoencoder (AE) based on long short-term memory (LSTM) to learn the normal state of equipment and detect anomalies in multivariate streams occurring in production equipment components. LSTM-based AE contains an encoder and a decoder, each of which consists of LSTM networks, variants of recurrent neural networks (RNN).

In most manufacturing work areas, computer numerical control (CNC) is utilized to shape and machine metal and other rigid materials by cutting, boring, grinding, shearing, or other deformations. Luo *et al.* [30] proposed an early fault detection model for a CNC machine. They employed a stacked autoencoder (SAE) to mine sensitive fault features from large-scale vibration data during long-term operations. They used cosine similarity function as a health indicator for predictive maintenance.

After convolutional neural networks (CNN) revolutionized the field of computer vision [31], researchers also began to apply CNN to time-series data analysis [32]. CNN-based fault detection and diagnosis models showed their competence in handling multivariate time-series data captured from semiconductor manufacturing processes in [33]–[35].

2) INFRASTRUCTURE FACILITIES

Pumps, chillers, and scrubbers are representative infrastructure facilities for maintaining environmental conditions (e.g., temperature, purification, and pressure). In particular, industrial pumps are used for various reasons, such as sustaining a vacuum state in equipment or pipes and exhausting gases and sludge. Pumps are usually driven in parallel. Thus, even if one pump behaves abnormally, the other pump can compensate for it, leaving the operator unnoticed. This scenario provides tolerance for abnormalities, but the heavily loaded pumps will inevitably wear faster. Therefore, accurate detection and prediction of anomalies are required to enhance the stability of the manufacturing process. In this regard, Lindermann *et al.* [36] employed a discrete wavelet transform (DWT) and LSTM-AE to detect anomalies across multiple pumps. Another method used CNN to recognize failures with converted images from vibration signals of pumps [37].

Heating, ventilating, and air conditioning (HVAC) is a representative system that is key to providing indoor environmental comfort via temperature control, oxygen replenishment, and removal of moisture and contaminants. Recently, deep learning-based anomaly detection and diagnosis models for this system have been proposed in [38], [39].

During chemical processes, abrupt changes in the air supply or the contamination levels can significantly damage the product quality. Therefore, several anomaly detection studies have been conducted over the years. Wu and Zhao citewu2020fault employed a pre-trained AlexNet, one

[1] https://www.lamresearch.com/wp-content/uploads/2018/01
[2] https://expo.semi.org/korea2020/Custom
[3] https://researchforecast.com/global-automated-guided-vehicle-market

of CNN models, to extract general features from data and perform transfer learning using the joint maximum mean discrepancy. The proposed model showed a great generalization performance to various chemical processes. Another example [41] used LSTM for the early detection of faults via particle attrition in a chemical-looping system. Contaminant detection and treatment are essential in wastewater treatment (WWT) as well. A recent study leveraged LSTM to monitor and detect faults in the WWT process, showing a remarkable performance [42].

3) LOGISTICS AUTOMATION SYSTEM

The manufacturing industry's recent interest in highly flexible production systems is related to the increasing demand for more individualized products [43]. This situation requires production flexibility, which has been enhanced by autonomous guided vehicles (AGV) that transport product components between work areas during the manufacturing process [44]. AGV reduce the cost of human intervention and allows on-demand changes regarding product types.

Despite numerous advantages, there are several crucial obstacles that must be overcome when using AGV. For example, if one of the vehicles is damaged or malfunctions, it can cause a bottleneck, and the others have to move further, resulting in significant economic loss. To take an appropriate action when such a problem occurs, the condition of vehicles must be monitored at all times. Acosta and Kanarachos [45] presented a method that estimates nonlinear vehicle dynamics based on signals in the vehicle. They employed a structure composed of an Extended Kalman Filter (EKF) and neural networks to predict the lateral tire forces and the road grip potential. EKF assumes the distribution of uncertainty as nonlinear Gaussian and estimates this by repeating prediction and correction. Gräber *et al.* [46] proposed a side-slip angle estimator using RNN with gated recurrent units (GRU). Because RNN, especially with GRU, explicitly models long-term dependencies, it achieves an excellent estimation quality while generalizing over different conditions. Although conventional approaches like EKF are still dominant in the industry, a well-designed RNN with sufficient data can be a competitive solution since it relies on fewer model assumptions like the underlying physical equations.

Another solution would be to monitor the route the AGV is traveling rather than the AGV itself or to avoid congested sections. Since the early 2000s, in semiconductor manufacturing plants, tens-of-thousands of AGV have transported wafers along ceiling rails (i.e., the overhead hoist transport). In these systems, neural network-based methods [47], [48] have been proposed for rail condition diagnosis. They monitor the positions of the upper- and lower- rail cables and the cable holders. Another method used a decision tree [49] to detect unplanned stopping or slowing of vehicles in factories.

B. SMART ENERGY MANAGEMENT

Stable supply and efficient consumption of energy are essential to cope with rapid climate changes and resource shortages. Thus, anomaly detection in energy supply and consumption processes has become increasingly important. In terms of supply, if a power outage occurs, it causes significant losses to consumers. In contrast, if energy is unnecessarily consumed, higher prices are paid and energy is wasted.

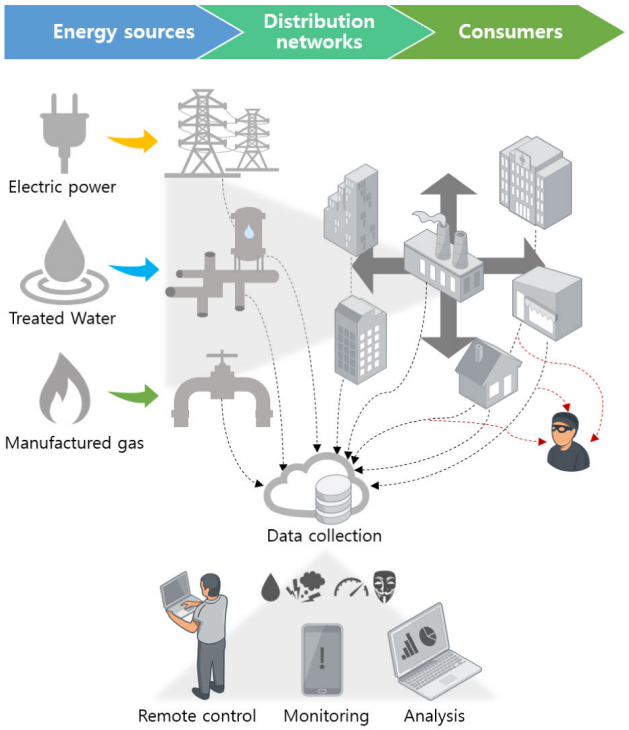


FIGURE 3. Smart energy management systems collect data from energy supply and consumption processes. It provides real-time monitoring to alert possible failures (e.g., leaks, overloads, cyber intrusions), helps stakeholders analyze data, and sometimes renders remote control.

As illustrated in Fig. 3, a large amount of data are collected and reported in the smart-energy management system. This provides all involved individuals the opportunity to better understand and predict consumption patterns. Autonomous collection devices, in turn, reduce the requirement for manual meter readings [11]. Furthermore, real-time early detection of possible failures allows energy suppliers to deal with problems ahead of time instead of relying on reactionary efforts. The success of smart-energy systems in the power sector has enabled the full embodiment of the smart-grid paradigm in water and natural gas fields [50].

1) ELECTRIC POWER

Several applications in [51]–[53] have been proposed to detect anomalies in multivariate time-series data generated by power plants. They take advantage of various deep neural-network models (e.g., convolutional LSTM, CNN, and attention layers), achieving remarkable results. Aside from anomaly identification, collected metrics data are used to diagnose the severity of problems.

A wide variety of approaches has also been proposed to detect consumer-side losses, such as abnormal consumption

patterns, unnecessary waste, and theft [54]–[56]. Diagnosis results are reported to the consumer using the energy-management systems to prevent problems and develop future strategies.

2) TREATED WATER

Water treatment and distribution systems determine the quality of both potable and industrial water supplies. Water-treatment facilities mainly exist in secure areas, but distribution networks are comprised of countless pipelines that span large areas. Since distribution networks are widespread and often vulnerable, the risk of physical attacks always exists. To make the matter worse, a cyber-intrusion poses a bigger threat, and the related damages have a significant impact. In this regard, several real-world datasets (e.g., SWaT and WADI) have been released [57], [58] so that researchers can use them without the need to collect vast numbers of data personally.

Li *et al.* [59] adopted a generative adversarial network (GAN) to detect anomalies in multivariate time-series data and validated their method on the aforementioned datasets. More recently, a method using a temporal hierarchical one-class network (THOC) [60], a combined structure with several layers of dilated RNN and multiscale support vector data description (MVDD), has shown a superior performance to the other state-of-the-art networks.

Several tools that detect abnormalities in consumption patterns also exist. Representatively, Vercruyssen *et al.* [61] exploited an active-learning strategy using constraint-based clustering and label propagation to monitor water consumption.

3) MANUFACTURED GAS

Crude oil, hard coal, and natural gas are manufactured into petroleum products and transformed into solids, liquids, and gases worldwide. Similar to the water-treatment process, the purification and refinement processes directly affect quality of petroleum products. Inspired by a successful image segmentation network, Wen *et al.* presented a time-series anomaly detection model using a CNN [62] that adopted a transfer-learning framework to resolve data sparsity issues. They demonstrated its effectiveness with the gasoil plant heating-loop dataset [63], which includes cyber-attacks on utility systems as a variety of data points. Moreover, energy management systems are required to manage gas storage and transport thoroughly and constantly, not only for cost reduction but also for environmental safety. On that matter, a recent CNN-based model was proposed [64] to detect gas leaks by monitoring flow noise inside the pipes.

C. CLOUD COMPUTING SYSTEM

In cloud computing, client data are stored and managed in remote data centers by a service provider [65]. These providers are required to allocate appropriate resources to users in real-time while storing sensitive information securely. As cloud services become more popular, intrusion detection has become crucial. Hence, providers now leverage logs and time-series data to monitor the states of servers and networks to detect deviations from normal patterns. Hundreds of thousands of suspicious events are continuously detected by such monitoring systems every day. Therefore, time-series anomaly detection on cloud systems with subsequent diagnosis of the current state and tracing of the root causes is important to maintain high service availability [66], [67].

1) SERVER MACHINE

On a server, multivariate time-series metrics, such as the processor load, the network usage, and the memory status, are made available. Su *et al.* [68] proposed a variational AE (VAE) with gated recurrent units (GRU) for monitoring a server machine, named OmniAnomaly. They combine the hidden state of the GRU $\mathbf{e_t}$ and the stochastic variable of the previous time step $\mathbf{z_{t-1}}$ in *qnet*, which acts as an encoder. And the resulting value is fed to the dense layer to sample the current stochastic variable $\mathbf{z_t}$. This variable passes through the planar normalizing flow so that it can learn the complex posterior well, and it is connected to $\mathbf{z_{t-1}}$ using linear Gaussian state space model in the *pnet*, which acts as a decoder, to obtain temporal dependence. After that, the value $\mathbf{x}'_\mathbf{t}$ is sampled from the estimated distribution through the reconstruction process. For similar purpose, hierarchical temporal memory (HTM) and Bayesian network-based approaches have been proposed [69]. Meanwhile, CNN-based approaches [70], [71] have been verified to be effective on several datasets from global cloud enterprises.

2) NETWORK AND FRAMEWORK

Moreover, as the network traffic grows exponentially, it becomes ever more necessary to constantly monitor network systems and distributed processing frameworks. Audibert *et al.* [72] proposed AE, in which one encoder and two decoders are trained adversarially, to identify network anomalies. In addition, Zhao *et al.* [73] recently suggested a graph attention network-based method to detect anomalies in a big-data processing framework. They explicitly modeled correlations between sensors via attention layers, captured temporal dependence with GRU, and increased performance by jointly applying forecasting and reconstruction results.

3) CYBERSECURITY

In addition to ordinary physical threats, malicious cyber-attacks have become critical issues for the reliability and security of cloud systems. For this reason, numerous methods have been proposed to protect customers' sensitive information [74]–[76].

D. STRUCTURAL HEALTH MONITORING

Civil infrastructure, including buildings, bridges, levees, pipelines, are composed of large and complex structures that carry large loads while operating in tough environments. These structures are designed to operate safely under

expected loading ranges, but corrosion and damage can occur due to repeated exposure to operation over their lifespans. If the damages are not detected on time, a structure becomes more vulnerable to failure or results in a safety accident. Structural health monitoring (SHM) evaluates their loads and responses and identifies abnormal behaviors to maintain these structures [77]. Some anomalies in SHM data caused by imperfect sensors and the poor quality of data transmission must be eliminated because they can cause false alarms and affect the structural performance assessment. However, eliminating them requires expertise and is very time-consuming.

In this respect, several approaches have been proposed recently. Bao *et al.* [16], imitating the recognition process of humans, transformed data as image files and fed them into stacked autoencoders (SAE) for anomaly classification. They trained each layer of the network one at a time, and this training scheme is referred to as *greedy layer-wise training*. After this phase is completed, they fine-tune all layers to improve the results. They verified the performance of the proposed framework with real-world data from a long-span cable-stayed bridge in China.

Similarly, Tang *et al.* [17], taking advantage of the interpretability of visualized data, converted raw time-series data to images and split the continuous data into segments by windowing data without overlap. Afterwards, they fed the pre-processed data into a CNN-based classification model. Each segment was decomposed into the time domain and frequency domain with Fast Fourier Transform (FFT) and fused as an image by stacking time response image and frequency response image.

IV. CHALLENGES OF CLASSICAL APPROACHES

Even before deep learning was popular, people had developed various mathematical and statistical models to analyze time-series data, applying them widely across various fields. Here, we introduce some representative methods and describe the challenges that remain to be solved.

A. CLASSICAL APPROACHES

1) TIME/FREQUENCY DOMAIN ANALYSIS

Time-series data can be analyzed in the time domain using the width and the height of measured thresholds. Another straightforward yet efficient method is to apply Fourier analysis to examine data with frequency-domain representations. According to the Fourier theorem, any periodic function, no matter how complex it is, can be expressed as a combination of periodic components, such as a sum of sines/or cosines. Fourier analysis is a process that recovers the function from those components. Discrete Fourier transform (DFT) is one of the popular methods and takes the following form:

$$X_k = \sum_{t=0}^{T-1} x_t e^{-i2\pi kt/T}, \quad k = 0, \ldots, T-1, \tag{3}$$

where X_k is k-th frequency value transformed from given input data x_t. Once you transform the raw time series to a frequency spectrum, as in (3), and sort it by coefficients, you can acquire the seasonal periods by inverting the highest frequency. In practice, fast Fourier transform (FFT), a speed-up version of DFT, is a preferred choice.

2) STATISTICAL MODEL

To mathematically analyze time-series data, we can generate a statistical model by calculating statistical measures, such as mean, variance, median, quantile, kurtosis, skewness, and many more. With the generated model, newly added time-series data can be inspected to determine whether it belongs to the normal boundary [78].

3) DISTANCE-BASED MODEL

Many algorithms use the explicit-distance between two temporal sequences to quantify the similarity between the two. Based on the obtained similarity metric, newly obtained sequences will be flagged as an anomaly if their distances from the normal one fall outside the expected range. The most common measure of distance is the Euclidean distance, as in (4), which computes the distance as the length of a segment connecting two points.

$$D(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{t=1}^{T} (x_i - y_i)^2}. \tag{4}$$

Dynamic time warping (DTW) is a popular distance measure, allowing nonlinear alignments between two sequences that are locally out of phase [79]. Assume that we have two sequences X and Y, whose lengths are M and N, respectively. DTW between the two sequences are measured as follows:

1) Create cost matrix C using dynamic programming algorithm, as in (5).

$$C(i,j) = D(i,j) + \min \begin{cases} C(i-1, j-1) \\ C(i-1, j) \\ C(i, j-1), \end{cases} \tag{5}$$

   where i is a data point of X, j is of Y, $D(i,j)$ is a distance between i and j, and $C(i,j)$ is a minimum warp distances of two sequences.
2) Trace back from $C_{M,N}$ to $C_{1,1}$ to get the optimal warping path $W(w_1, w_2, \ldots, w_L)$, choosing the previous points with the lowest cumulative distance.
3) Finally, calculate the final distance using W, as in (6).

$$\text{Dist}(W) = \sum_{k=1}^{k=L} w_k. \tag{6}$$

4) PREDICTIVE MODEL

Predictive models are used to forecast future states based on the past and current states. We can deduce the anomaly according to the severity of the discrepancy between the

predicted value and the real one. For example, the autoregressive integrated moving average (ARIMA) [80] are frequently employed models to forecast time series. ARIMA model is composed of three parts:

- Auto-regressive (AR) model is composed of a weighted sum of lagged values, and thus we can model the value of a random variable X at time step t as (7).

$$\mathrm{AR}(p) : X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \ldots + \phi_p X_{t-p} + \epsilon_t, \tag{7}$$

  where $\{\phi_i\}_{i=1}^{p}$ are auto-correlation coefficients, ϵ is an white noise, and p is the order of AR model.
- Moving-average (MA) model computes the weighted sum of lagged prediction errors and is formulated as (8).

$$\mathrm{MA}(q) : X_t = \epsilon_t - \theta_1 \epsilon_{t-1} - \theta_2 \epsilon_{t-2} - \ldots - \theta_q \epsilon_{t-q}, \tag{8}$$

  where $\{\theta_i\}_{i=1}^{1}$ are moving-average coefficients, ϵ_t denotes a model prediction error at time step t, and q is the order of MA model.
- Integrated (I) indicates the time series using differences, and thus a data point at time step t is $\hat{X}_t = X_t - X_{t-1}$, when $d = 1$, where d denotes the order of differencing.

As a result, the ARIMA model with the order-parameters is formulated as follows:

$$\hat{y}_t = \mu + \underbrace{\phi_1 y_{t-1} + \phi_2 y_{t-2} + \ldots + \phi_p y_{t-p}}_{\mathrm{AR}(p)} \underbrace{-\theta_1 \epsilon_{t-1} - \theta_2 \epsilon_{t-2} - \ldots - \theta_q \epsilon_{t-q}}_{\mathrm{MA}(q)}, \tag{9}$$

where μ is a constant and $y_t = Y_t - Y_{t-1}$, when $d = 1$. As described in (9), each value at a specific time step is affected by previous observations and prediction errors, so the ARIMA models the temporality of time series. Also, the differencing process makes the time series stationary, resulting in the ARIMA being effective for non-stationary time series. If the time-series data has a seasonal- or cyclic- variation, we can use a seasonal ARIMA (SARIMA) [81] model. In this case, we introduce additional parameters: P, D, and Q, which deal with the seasonality. These parameters are used in the same manner as p, d, and q.

Fundamentally, ARIMA is not capable of modeling multivariate data. Instead, autoregressive integrated moving average exogenous (ARIMAX) [82] model that has an additional explanatory variable or vector autoregression (VAR) [83] model that uses vectors to accommodate the multivariate terms is used to replace ARIMA.

5) CLUSTERING MODEL

In an unsupervised setting, clustering-based methods are simple yet effective choices for grouping the data and detecting the anomalies. Once you map time-series data into a multidimensional space, clustering algorithms group them close to the centroid of each cluster depending on their similarities.

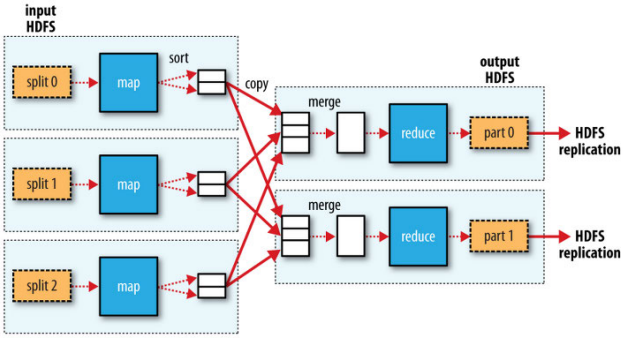


FIGURE 4. Flowchart of MapReduce model.

Models classify newly received data samples as anomalies if they are far from pre-defined clusters or have low probability of belonging in any of the clusters.

Popular data clustering methods include the k-means algorithm [84], one-class support vector machine (OCSVM) [85], Gaussian mixture model (GMM) [86], and density-based spatial clustering of applications with noise (DBSCAN) [87]. The above methods may be insufficient to be applied when datasets have mixed attributes, such as numerical and categorical values. To resolve this issue, the k-prototypes algorithm [88], a simple combination of k-means and k-modes algorithm, was proposed. The k-prototypes algorithm measures dissimilarity between two mixed-type objects X and Y, which are described by attributes $A_1^r, A_2^r, \ldots, A_p^r, A_{p+1}^c, \ldots, A_m^c$. The dissimilarity is measured as [88, eq. (10)].

$$d_2(X, Y) = \underbrace{\sum_{j=1}^{p} (x_j - y_j)^2}_{\text{numeric attributes}} + \gamma \underbrace{\sum_{j=p+1}^{m} \delta(x_j, y_j)}_{\text{categorical attributes}}, \tag{10}$$

where the first term is the Euclidean distance between the numeric attributes and the second one is a simple matching dissimilarity between the categorical attributes.

The above clustering methods are still representative benchmarks but are becoming outdated. Recently, data has become more large-scaled, and thus it requires clustering algorithms that can deal with the massive size of data in both sequential and parallel computing environments. In order to effectively process large amounts of data, we can consider two approaches; One is to increase the computational speed by reducing the size of the data, and the other is to split the data into small chunks and process them in parallel.

Structural clustering algorithm for networks (SCAN) [89] is one of the successful density-based clustering algorithms for a graph, a fundamental data structure. Several works [90]–[92] use nodes/edges pruning techniques to reduce the number of structural similarity comparisons, thereby boosting the efficiency of SCAN without sacrificing the clustering quality for graphs with millions or even billions of edges. These methods skip vertices that are shared between the neighbors or remove outliers before update clusters.

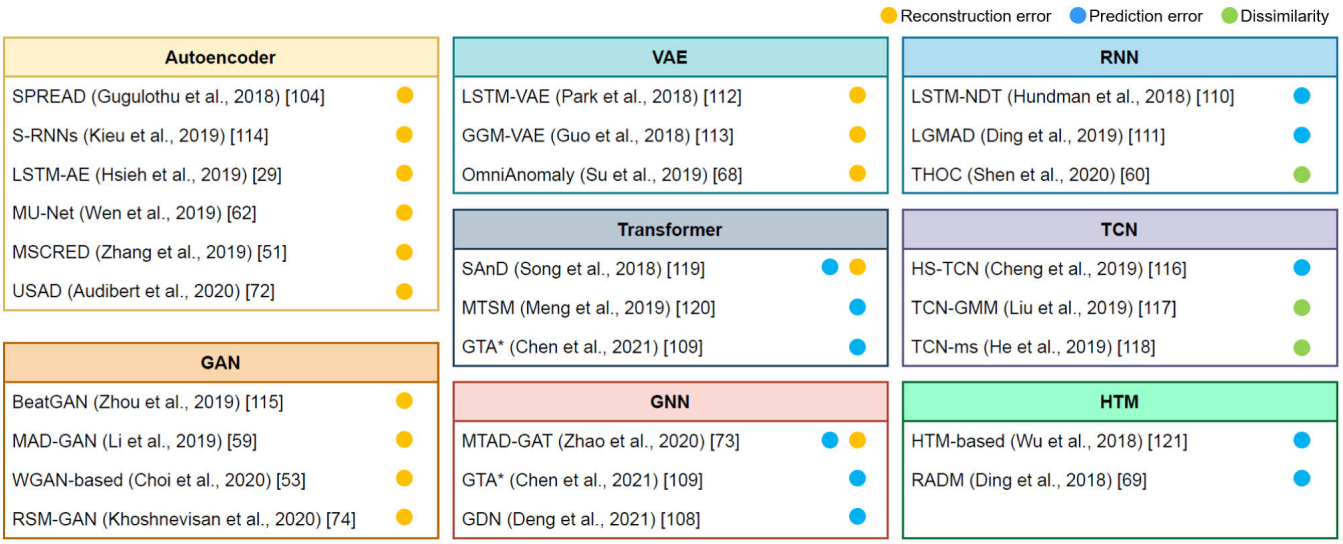


FIGURE 5. A taxonomy of recent deep learning-based time-series anomaly detection methods. HTM, hierarchical temporal memory; RNN, recurrent neural networks; TCN, temporal convolutional networks; GNN, graph neural networks; GAN, generative adversarial networks; VAE, variational autoencoder. Most of the models do not use only one structure or method but combine several ones. We classify the models based on the main structural characteristics of each model and denote types of anomaly scores with colored circles. * is an exception because the roles and influences of Transformer and GNN are clearly separated.

Similarly, Li [93] improve DBSCAN, a density-based clustering algorithm for numerical data, to prevent redundant computations with the fast nearest neighbor query that exploits the triangular inequality.

The second approach is to distribute the data among several machines or processors to accelerate processing of an extensive volume of data. MapReduce [94] is one of the most widely used parallel processing models for data-intensive applications. As illustrated in [95, Fig. 4], this model consists of two main functions: the Map and the Reduce functions. Considering k-means as an example, MapReduce tasks follow the procedure as:

1) The dataset is split into multiple chunks and they are fed to the mappers in the form of <*index, value*>.
2) The Map functions calculate the distance of each sample from centers, and then assign the samples to the closest cluster: <*index, center*>.
3) The Reduce functions compute the partial summation of the samples with the same center and binds them in the form of <*center, (sum, #samples)*>.
4) The synchronization phase sequentially calculates the new centers by dividing the *sum* by *#samples* and update centers: <*cluster, new center*>.
5) Repeat until convergence.

Over the past few years, variants of k-means clustering using MapReduce [96]–[98] have been introduced. Meanwhile, Scalable k-means++ [99] utilizes MapReduce at the initialization phase instead of the post-initialization phase.

B. CHALLENGING ISSUES

Although traditional approaches have made much progress in anomaly detection in time-series data, there is still room for improvement because of the following challenges.

1) LACK OF LABELS

Failure modes in most industrial circumstances are extremely rare, and therefore they are insufficient for use as labeled training data. The scarcity of failure modes makes collecting enough labeled training data time- and resource-intensive. Even when labeled data are obtained, the class imbalance between normal and abnormal data hampers model training.

2) COMPLEXITY OF DATA

Analyzing univariate time-series data is still a critical topic in applications that require less computation, such as edge computing. Nonetheless, as more industrial applications are automated and the complexity of control systems increases, separately monitoring individual univariate time-series data becomes impractical. With the large numbers of dimensions, traditional approaches generally experience a non-negligible drop in performance due to the *curse of dimensionality*. Moreover, correlations between variables that cannot be inferred by univariate time-series analysis can also be used to indicate anomalies.

V. DEEP LEARNING FOR ANOMALY DETECTION

In this paper, we focus on recent anomaly detection models that have been used to overcome the challenging issues mentioned in Section IV. Therefore, our survey works under the following assumptions.

- *Semi-Supervised/Unsupervised Learning:* All data are considered to be in the normal class for semi-supervised learning, whereas no explicit distinction between normal and abnormal classes is considered in unsupervised learning. Both strategies learn the data structure to overcome the shortage of labeled data.

TABLE 2. Inter-correlation between variables.

Category	Method	Publication
Dim. reduction	Independent univariate	Shahriar *et al.* [100], ODCA [101], R-PCA [102]
	Reduced multivariate	MAD-GAN [59], DAGMM [103], SPREAD [104]
2D matrix	Pairwise inner-product	MSCRED [51], RSM-GAN [74], Zhou *et al.* [105]
	Pairwise phase	Choi *et al.* [53]
Graph	Graph neural network	MTAD-GAT [73], AddGraph [106], MTAD-TF [107], GDN [108], GTA [109]
Others		THOC [60], OmniAnomaly [68], LSTM-NDT [110], LGMAD [111], USAD [72]

- *Multivariate Data:* The models should be capable of extracting and exploiting the information entangled in multivariate time-series data.
- *Deep Learning:* Deep-learning methods are explored to handle a complex and massive amount of data.

The overall taxonomy of recently published anomaly detection methods for time-series data is shown in Fig. 5. We classify the methods according to their choices of architectures and denote how they calculate the anomaly score from given data with colored circles. Two or more of the scores can be jointly considered.

In this section, we analyze these methods from three perspectives: how they define inter-correlation between variables; how they model the temporal context information; and how they define anomaly scores or thresholds.

A. INTER-CORRELATION BETWEEN VARIABLES

Most deep-learning models for multivariate time-series data establish relationships among multiple variables at every time step. This spatiotemporal information considers not only the temporal context but also the correlation between variables. Table 2 shows how the correlations of multivariate variables are established in the recent works.

1) DIMENSIONAL REDUCTION

A status of a large-scale system can be represented using a few significant factors. Thus, we can reduce the amount of computation by extracting the main features via dimensional reduction. Typically, a linear algebra-based method including principal component analysis and singular value decomposition, or a neural-network-based method including AE and VAE is used. Some previous works process the individual univariate time series, while the others treat the reduced representations as multivariate series. Dimension reduction also has a setback: detecting the cause of anomaly is difficult.

2) 2D MATRIX

A 2D matrix directly captures the morphological similarity and the relative scale among individual variables. Moreover, it considers multivariate variables jointly, making it robust to turbulence at specific points in time. Two representative definitions of the 2D matrix, $m^t \in \mathbb{R}^{n\times n}$ are formulated as follows:

$$m_{ij}^t = \frac{1}{w}\sum_{\delta=0}^{w} x_i^{t-\delta} x_j^{t-\delta}, \tag{11}$$

$$m_{ij}^t = \frac{1}{w}\sum_{\delta=0}^{w} \left\| x_i^{t-\delta} - x_j^{t-\delta} \right\|, \tag{12}$$

where $X = \{\mathbf{x}^1, \mathbf{x}^2, \cdots, \mathbf{x}^T\}$ are multivariate time-series data with n variables of length T, that is, $X \in \mathbb{R}^{n\times T}$, and $\mathbf{x}^t = (x_1^t, x_2^t, \cdots, x_n^t)$ is an n-dimensional vector. On one hand, if the phase of the entire variable suddenly rises or falls due to an unexpected event, (11) can detect anomalies, but (12) cannot. On the other hand, when the overall phase changes by a concept drift or a change point, (12) dismisses this event as normal, while (11) flags an unnecessary alarm.

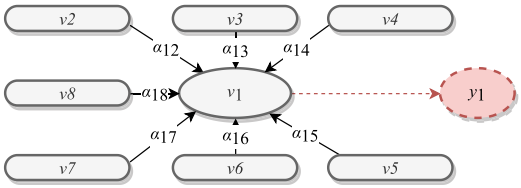


FIGURE 6. A mechanism of graph attention layer. Red circle is the final output.

3) GRAPH

A graph can define an explicit topological structure and learn the causal relationship among individual variables. Recently, several approaches [73], [108], [109] that applied an attention mechanism to GNN have been proposed to improve performance for identifying root causes. A directed graph is formulated as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, 2, \ldots, N\}$ is the set of N nodes, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is set of edges. Here, e_{ij} denotes the edge from node i to j. Generally, given a graph, the attention layer outputs representation for each node as follows:

$$y_i = \sigma\Big(\sum_{j=1}^{L} \alpha_{ij} v_j\Big), \tag{13}$$

where y_i denotes the feature representation of node i. σ corresponds to the sigmoid activation function, α_{ij} to the attention score which measures the influence of node j to

TABLE 3. Modeling temporal context.

Category	Method	Publication
RNN	Long Short Term Memory (LSTM)	LSTM-NDT [110], LGMAD [111], LSTM-AE [29], MAD-GAN [59], OmniAnomaly [68], SPREAD [104], LSTM-VAE [112]
	Gated Recurrent Unit (GRU)	THOC [60], GGM-VAE [113], S-RNNs [114]
CNN	Convolutional Neural Network (CNN)	Choi *et al.* [53], MU-Net [62], BeatGAN [115]
	Temporal Convolutional Network (TCN)	HS-TCN [116], TCN-GMM [117], TCN-ms [118]
Hybrid	Convolutional LSTM (ConvLSTM)	MSCRED [51], RSM-GAN [74]
Attention	Self-attention or Transformer	MTAD-GAT [73], SAnD [119], MTSM [120], GTA [109]
Others	Hierarchical Temporal Memory (HTM)	RADM [69], Wu *et al.* [121]

node i, where j is one of the L adjacent nodes of i, and v_j to the feature vector of node j. We can compute the attention score α_{ij} by the following equations:

$$e_{ij} = \text{LeakyReLU}(w^\top \cdot (v_i \oplus v_j)) \tag{14}$$

$$\alpha_{ij} = \frac{\exp e_i j}{\sum_{l=1}^{L} \exp e_i l}, \tag{15}$$

where $\oplus$ concatenates two node features. w denotes a set of learnable parameters, and LeakyReLU is a nonlinear activation function that has a gentle slope for negative values. Fig. 6 illustrates the intuition behind the graph attention.

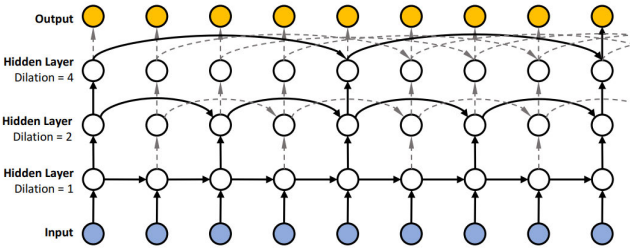


FIGURE 7. An example of a three-layer dilated RNN with dilation 1, 2, and 4. With its recurrent skip connection and its use of exponentially increasing dilation, it alleviates gradient problems and extend the range of temporal dependencies with fewer parameters.

4) OTHERS

Some methods [60], [68], [72], [110] that use the raw data can directly identify anomalies in the data. Meanwhile, Ding *et al.* [111] employs a multivariate Gaussian distribution to define correlations between data attributes.

B. MODELING TEMPORAL CONTEXT

The history of a sequence contains a great deal of knowledge about its behavior and can suggest future shifts. Hence, estimating the distribution alone is limited in detecting context and collective anomalies. In time-series applications, the temporal context should be considered when modeling the normal status. Table 3 shows the taxonomy of models in terms of modeling the temporal context.

1) RNN

Several deep learning-based approaches to model the temporal context. One of the most common benchmarks uses RNN to recognize pattern sequences and predict expected values. Thus, we can determine anomalies by identifying the differences between the predicted and actual signals. RNNs have been extended with other variants, such as LSTM [122] and GRU [123].

LSTM and GRU address the vanishing or exploding gradient problem, where the gradient becomes too small or too large as the network goes deeper. There are multiple gates in an LSTM and a GRU cell, and they can learn long-term dependencies by determining the number of previous states to keep or forget at every time step. Meanwhile, the dilated RNN, as illustrated in [124, Fig. 7], is proposed to extract multi-scale features while modeling long-term dependencies by using a skip connection between hidden states. Shen *et al.* [60] adopt a three-layer dilated RNN and extract features from each layer to jointly consider long- and short-term dependencies.

RNN-based approaches are generally used for anomaly detection in two ways. One is to predict future values and compare them to predefined thresholds or the observed values. This strategy is applied in [60], [110], [111], [114]. The other is to construct an AE or VAE to restore the observed values and evaluate the discrepancy between the reconstructed value and observed one. This strategy is used in [29], [59], [68], [104], [112], [113].

2) CNN

Although the RNN is the primary option for modeling time-series data, CNN sometimes shows better performance in several applications [53], [62], [115] that work with short-term data. By stacking convolutional layers, each layer learns a higher level of features from pixels to objects. In addition, the pooling layers introduce non-linearity to CNN, allowing them to capture the complex features in the sequences.

Instead of explicitly capturing the temporal context, the CNN models learn patterns in segmented time series. Hence, one of its drawbacks is that it is not easy to comprehend behaviors appearing over a long period. As an alternative, Temporal convolutional networks (TCN), a variant of CNN, has been proposed in [125]. There are three distinguishing properties of TCN. First, the convolutions in the

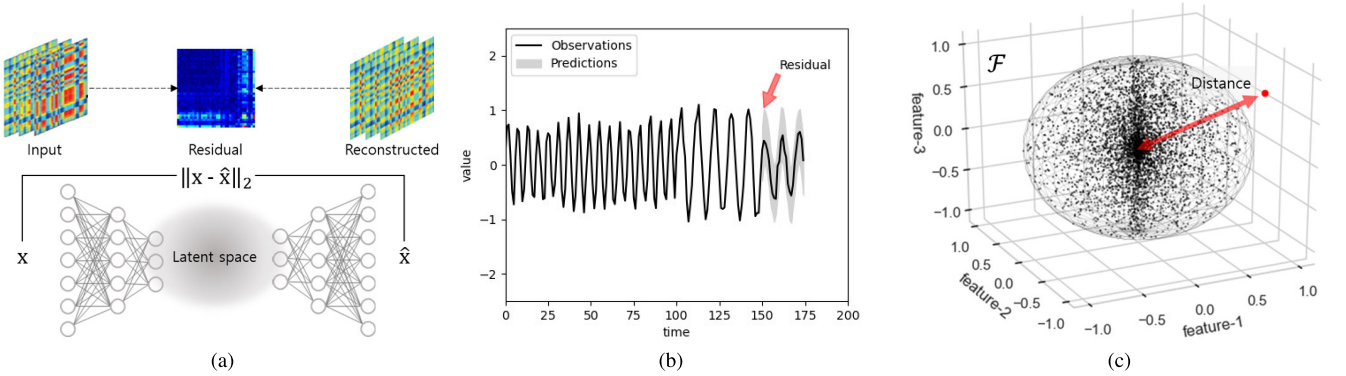


FIGURE 8. The examples of each type of anomaly criteria: (a) a reconstruction error; (b) a prediction error; and (c) a dissimilarity.

model are causal, meaning that they ensure no information leakage from the future to the past. Second, it can take a sequence of any length, just as with an RNN. Third, it can look quite far into the past to forecast futures using a combination of deep networks and dilated convolutions.

3) HYBRID

When monitoring time-series data with a sliding window, the detectable anomaly pattern varies according to the window size. For example, assume that we have three different windows for 30 sensor data and define a covariance matrix for each window. Then, the shape of the data becomes (30, 30, 3) at time t like an image. If we stack the covariance matrices from $t - 4$ to t to the time axis, the shape of the data becomes (5, 30, 30, 3) like a video, in which case, we should consider the spatial information and temporal dependencies simultaneously.

Shi *et al.* [126] first proposed a ConvLSTM model to solve the spatiotemporal sequence-forecasting problem. They replace the dot products in the LSTM cell with convolution operators, and consequently, all gates and states in the cell are reshaped into 3D tensors that can capture spatiotemporal information. Moreover, the model learns state transitions with fewer parameters. In [51], [74], the overall architectures were based on AE and GAN, respectively. In their encoders, ConvLSTMs capture the spatiotemporal context from the feature maps across the previous time steps. Additionally, a temporal attention mechanism [127] adjusts the contribution of the previous feature maps to update the current one.

4) ATTENTION

The attention mechanism was initially used as an auxiliary tool in models. However, novel approaches based on attention layers, such as Transformer [128] and bidirectional encoder representations from transformer (BERT) [129], have become mainstream in natural language processing (NLP). By paying attention to the input weights that contribute more to the output, the attention-based models can capture very long-range dependence with a relative importance to each data point. The remarkable achievements in NLP have led to a time-series anomaly detection domain. In this regard, several works [73], [109], [120] employing Transformer are presented recently.

5) OTHERS

Hierarchical temporal memory (HTM) is considered to be one of the most promising next-generation deep learning algorithms. It is designed to embody the structure and interaction of pyramidal neurons in the neocortex [69]. It comprises of stacked cells in a tree shape, and the columns of cells are activated by the input and the previous states of connected neighbors. HTM can capture and predict sequence patterns and thus is beneficial to anomaly detection in time-series data. what makes HTM more unique is that it continuously learns temporal patterns from streaming data without backpropagation. Hence, HTM requires minimal human intervention to be trained in an unsupervised manner.

C. ANOMALY CRITERIA

The models addressed above learn the representation of the given data in an unsupervised or semi-supervised manner by minimizing a defined objective (loss) function. The objective differs according to the model architecture and is generally related to the decision criteria for abnormality.

Once the models are trained, they are applied to the systems and machinery state diagnoses. In general, diagnostic results are expressed in numeric to help understand a given status. We call this numeric indicator an *anomaly score*. The greater it is, the more likely the state is to be abnormal. Specifically, when the score exceeds a certain threshold, the corresponding data point is determined as an anomaly. In the past, domain experts decided this threshold empirically, but now it is decided according to the model-training result. Some models [68], [69], [110]–[112], [120], [121] employ an adaptive threshold that continuously adjusts to the changes in data over time. The schemes for deriving an anomaly score can be classified into three types, as depicted in Fig. 8: a reconstruction error, a prediction error, and a dissimilarity.

1) RECONSTRUCTION ERROR

In general, AE, VAE, GAN, and Transformers use reconstruction errors as anomaly scores. AE-based models including [29], [51], [62], [72], [104], [114] reconstruct input data by extracting features from them. VAE-based models such as [68], [112], [113] estimate the data distribution and generate samples from it, which are very similar to the input data. GAN-based models explicitly generate samples that are as similar as possible to the input data with the generator, as in [53], [59], [74], [115]. Recently, Transformer with a stacked encoder-decoder structure, which consists only of attention mechanisms, is employed in several works [73], [109], [119], [120]. In particular, Zhao *et al.* [73] consider both prediction and reconstruction errors jointly in their model. Even though these models use different training schemes and objective functions, they calculate anomaly scores similarly. They reconstruct or generate data analogous to the input data and measure the residual between the input and generated data.

2) PREDICTION ERROR

There are two ways to derive anomaly scores from the prediction model. One applies a binary label based on the probability of the data point being classified as a normal, as proposed in [116], [119]. The prediction error indicates whether the expected label matches the ground truth. The other approach is to predict the expected value for the next time steps, as proposed in [69], [110], [111], [121]. In this case, the prediction error is the residual between the expected value and the observation. The second one is more practical than the first because the labels are insufficient in the real world.

3) DISSIMILARITY

Dissimilarity-based one measures how far the value derived by the model exists from the distribution or cluster of the accumulated data. There are various methods for measuring the similarity, such as the Euclidean distance, the Minkowski distance, the cosine similarity, and the Mahalanobis distance.

In the temporal hierarchical one-class (THOC) network [60] and TCN-Gaussian mixture model (GMM) [117], time-series features are extracted by a dilated RNN and TCN, respectively. Then, they are clustered using a similar deep support vector data description, or their distribution is estimated using a GMM. THOC measures the similarity between features and clusters using cosine similarity, and TCN-GMM uses the Mahalanobis distance. The similarity obtained from the models is subtracted from one to obtain an anomaly score. Conversely, multi-stage TCN [118] uses a multivariate Gaussian distribution to estimate the distribution of prediction errors rather than the features of training data. Then, the anomaly score is determined by measuring the Mahalanobis distance between the current prediction error and the pre-estimated error distribution.

VI. COMPARATIVE REVIEWS

In this section, we provide experimental performances of various methods on real-world datasets for time-series anomaly detection.

TABLE 4. Summary of datasets used in the experiments.

Dataset	#Dim.	#Training	#Testing	Anomaly rate (%)
SWaT	51	496,800	449,919	11.98
WADI	112	1,048,571	172,801	5.99
MSL	55	58,317	73,729	10.72

A. EXPERIMENTAL SETUP

To compare the performances of the presented methods, the following public time-series datasets are used:

- *Secure Water Treatment (SWaT) [57]:* Multi-variate time-series data collected over 11 days from water treatment test-bed, a small-scale cyber-physical system. The last 4 days of data contain 36 attacks. The objectives and the duration of these attacks are diverse. To get more information or request for the dataset, please refer to the SWaT website.[4]
- *Water Distribution (WADI) [58]:* Multi-variate time-series data from water distribution pipelines collected over 16 days. Each series includes various network traffic, sensor and actuator measurings. Out of 16 days, 14 days contain data under normal conditions, and two days under attack scenarios. Please refer to the WADI website[5] for more details.
- *Mars Science Laboratory Rover (MSL) [110]:* Multivariate time-series data recorded from Mars Science Laboratory rover. Training and testing testbeds are separated, and the anomalies in the testing testbed are all labelled. The data is available at the public storage.[6]

Several previous works of research have reported the performances of the anomaly detection methods on the datasets described in Table 4. The reported performances are used if available, and the other performances are obtained from our experiments. Detection results on SWaT [57] are available in [60], [72], and [109]; WADI [58] in [72], [109], and [108]; MSL [110] in [72], [108], and [68].

For performance evaluation, we adopt three standard evaluation metrics: Precision, Recall, and F1-score. They take the following form:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{16}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{17}$$

$$\text{F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{18}$$

[4]https://itrust.sutd.edu.sg/testbeds/secure-water-treatment-swat/
[5]https://itrust.sutd.edu.sg/testbeds/water-distribution-wadi/
[6]https://s3-us-west-2.amazonaws.com/telemanom/data.zip

TABLE 5. Hyper-parameters values used for each method. The methods marked with † indicate their papers also provided the performances of some other models measured under the same environment. The MSCRED [51] jointly uses three-sized sliding windows in the original work, and we have reflected this in our experiments.

Datasets	Methods	Down sampling	Window size	Epoch	Point adjust	Learning rate
SWaT	USAD [72]	0.2	12	70	✓	1e-3
	GTA [109]	0.1	60	50	✓	1e-4
	GDN [108]	0.1	5	50	✓	1e-3
	THOC [60]†, MSCRED [51], DAGMM [103], LSTM-VAE [112], OmniAnomaly [68]	0.1	100	100	✓	1e-3
WADI	MSCRED [51]	1	10, 30, 60	50	✓	1e-3
	THOC [60]	1	100	100	✓	1e-3
	GTA [109]	0.1	60	50	✓	1e-4
	GDN [108]	0.1	5	50	✓	1e-3
	MAD-GAN [59]	0.1	5	50	✗	1e-3
	USAD [72]†, DAGMM [103], LSTM-VAE [112], OmniAnomaly [68]	0.2	10	70	✓	1e-3
MSL	MSCRED [51]	1	10, 30, 60	50	✓	1e-3
	THOC [60]	0.1	100	100	✓	1e-3
	USAD [72]	0.2	5	250	✓	1e-3
	GDN [108]	0.1	5	50	✓	1e-3
	GTA [109]†, DAGMM [103], MAD-GAN [59], LSTM-VAE [112], OmniAnomaly [68]	1	60	50	✓	1e-4

TABLE 6. Anomaly detection accuracy in terms of precision (%), recall (%), and F1-score, on three datasets with ground-truth anomalies. * did not apply point adjustment on the WADI dataset, results in poor Recall and F1-score relatively.

Methods	SWaT			WADI			MSL		
	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
USAD	98.70	74.02	84.60	64.51	32.20	42.96	88.10	97.86	**92.72**
DAGMM	90.60	80.72	85.38	22.28	19.76	20.94	54.12	**99.34**	70.07
LSTM-VAE	98.39	77.01	86.40	46.32	32.20	37.99	52.57	95.46	67.80
OmniAnomaly	**99.01**	77.06	86.67	26.52	**97.99**	41.74	88.67	91.17	89.90
MSCRED	98.43	77.69	86.84	30.26	40.35	34.58	68.83	88.54	77.45
MAD-GAN*	98.72	77.60	86.90	41.44	33.92	37.3	85.17	89.91	87.47
THOC	98.08	79.94	88.09	42.12	63.34	50.59	82.70	96.48	89.06
GTA	94.83	88.10	91.34	83.91	83.61	83.76	**91.04**	91.17	91.10
GDN	95.85	**91.42**	**93.59**	**85.62**	85.41	**85.52**	82.92	99.19	90.33

Best performance in bold. Second-best with underlines.

where TP are the true positives that stand for the number of the detected true anomalies, FP are false positives that mean the incorrectly detected ones, and FN are false negatives that are undetected anomalies. Precision is the proportion of samples that are true anomalies among those predicted by the model as anomalies. Recall is the proportion of anomalies predicted by the model out of entire anomaly samples. Therefore, the higher Recall is, the more anomalies are caught without omission. At the same time, the higher Precision is, the fewer false alarms occur. Because Precision and Recall are inversely proportionate to each other in general, the threshold must be adjusted to evaluate model performance for different purposes. In many real-world scenarios, it is important for the system to detect as many actual attacks or anomalies as possible at the cost of few false alarms. Therefore, we focus more on Recall and F1-score than Precision in the experiments. Moreover, we report the best results of each model on all datasets for a fair comparison because different thresholds may result in different metric scores.

Anomaly detection methods for time-series data require various hyper-parameters tuned for the optimal performance. Since the optimal values of the hyper-parameters are not the same for each method, we report the used values in Table 5. Typical hyper-parameters include down sampling ratio, window size, point adjustment, and learning rate. In most case, time-series data used to be down-sampled prior to the experiments to model data of longer time frames under a fixed capacity of the model. According to [72], down-sampling speeds up learning by reducing the size of the data and also has a denoising effect. In addition, slicing each series using a

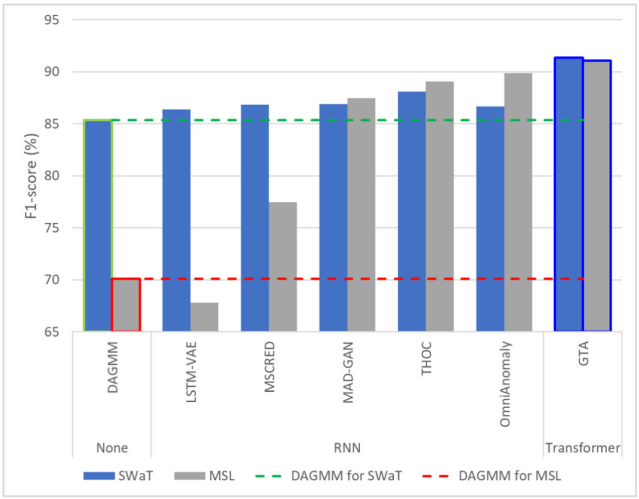


FIGURE 9. Experimental results on SWaT and MSL. The RNN-based and transformer-based models that capture temporal dependencies outperform DAGMM, the non-temporal modeling method.

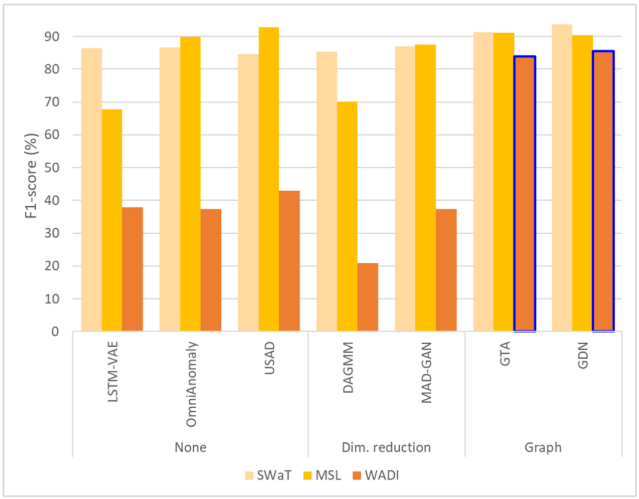


FIGURE 10. Experimental results on SWaT, MSL, and WADI. The dimension of the dataset affects the performance.

window of a fixed length is a common practice. *Point adjustment* is a technique to boost the recall of the detection model. Typical anomalies in datasets tend to be temporally adjacent. If the model successfully detects any of the anomalies within the segment when it makes decision for every time step, the evaluation process regards the whole contiguous segment of the anomalies as detected.

B. RESULTS AND ANALYSIS

We compare a wide range of state-of-the-arts in multivariate time series anomaly detection, categorized as follows:

- *AE:* DAGMM [103], MSCRED [51], OmniAnomaly [68]
- *VAE:* LSTM-VAE [112], USAD [72]
- *GAN:* MAD-GAN [59]
- *RNN:* THOC [60]
- *Transformer:* GTA [109]
- *GNN:* GTA [109], GDN [108]

Table 6 shows the anomaly detection accuracy in terms of Precision, Recall, and F1-score of the state-of-the arts on the benchmark datasets (SWaT, WADI, and MSL). Except for specific cases, we tried to employ the same experimental settings as much as possible to fairly compare the performance. If the comparison under the same settings is not plausible, we used the settings reported in the original paper. Each of these methods prioritizes a different metric as the authors choose specific thresholds depending on their goal. Therefore, we pick the F1-score as a baseline and sort the methods for SWaT correspondingly.

The result shows no clear one-size-fits-all method for all the datasets and no notable distinction in performance depending on their structure. Therefore, we interpret the results from several perspectives.

1) MODELING TEMPORAL DEPENDENCIES

Compared to DAGMM [103], designed to treat multivariate data without temporal information, RNN-based models show superiority (see Fig. 9). The average F1-scores of the RNN-based models on SWaT and MSL datasets are 1.87% and 14.90% higher than those of DAGMM, respectively. This is because they can take long sequences as input and capture the temporal dependencies.

LSTM-VAE [112] replaces the feed-forward network in a VAE with LSTM. MSCRED [51] is a CNN-based AE that reconstructs a feature map that contains both the aggregated information of observations and the inter-correlation between variables within a fixed-size sliding window. Between the encoder and the decoder, it captures the spatiotemporal dependencies from the feature maps across the previous time steps using ConvLSTMs. MAD-GAN [59] employs LSTM-RNN as both generator and discriminator to learn the temporal context in a generative adversarial training fashion and reconstructs the original time series explicitly. THOC [60] adopts multi-layers of dilated RNNs to model temporal dependencies with a wide range of lengths.

Most RNN-based methods outperform DAGMM, but with the exception of LSTM-VAE on MSL. We argue that the main reason behind this phenomenon lies in the process over the latent variables; Although LSTM-VAE uses LSTM for sequence modeling, it ignores the temporal dependencies among latent variables. Meanwhile, OmniAnomaly [68] connects stochastic latent variables in the middle of encoder and decoder with a linear Gaussian state-space model to model the temporal dependencies with inherent stochasticity. As a result, the approaches without modeling temporal dependency are not suitable for time-series anomaly detection.

2) PARALLEL PROCESSING FOR LONG SEQUENCES

Despite the powerful capability of sequence modeling, one drawback of RNN is that it restricts parallelization because it computes its output sequentially. Meanwhile, Transformer takes a sequence at once, so parallel processing is possible. Furthermore, it can reflect contextual information at once

by computing contributions between all-time steps through a self-attention mechanism. This property is significant to sequence modeling because a longer sequence can provide more information. Consequently, compared to DAGMM, GTA that aims to adopt Transformer achieves overall 6.98% and 30.01% improvements in terms of the best F1-score on SWaT and MSL datasets, respectively. GTA also shows 5.11% and 10.64% improvements compared to the overall mean of the F1-score of the RNN-based model on SWaT and MSL datasets, respectively.

3) DIMENSION OF THE DATASETS

As shown in Fig. 10, we can see that the overall performances in terms of the best F1-score on the WADI dataset are significantly lower compared to the other datasets (SWaT and MSL), except for GNN-based methods. Recall that the dimension of the WADI dataset is 112, double that of SWaT and MSL, as described in Table 4. When we feed the 2D feature map that defines correlations between variables, such as a covariance matrix, to the deep-neural network-based models, the amount of feature expression and computation will be more than quadrupled compared to SWaT and MSL. In particular, in reconstruction-based models with deep layers, the amount of computation is overloaded for each layer. Undoubtedly, the poor results for WADI are expected.

4) INTER-CORRELATIONS BETWEEN ATTRIBUTES

Despite several factors affecting performance, we can see that there is no remarkable difference in the results on the WADI dataset when simply comparing models that undergo dimensionality reduction in the preprocessing stage with those that do not. We argue that the possible reason is that some important features are lost during dimensionality reduction.

Meanwhile, GNN-based models (GTA and GDN) achieve a relatively higher F1-score on the WADI dataset. While GTA greatly benefited from the sequence modeling ability of the Transformer, GDN, which does not consider temporal dependencies yielded notable results by simply learning the graph structure of the relationship between variables. We believe that the major factor lies in the dependencies between features. SWaT and WADI provide the network traffic, measurements from sensors and actuators under several control processes. These attributes are not entirely independent of each other, and thus there exist inter-correlations between the attributes within the associated equipment and control processes. Therefore, trivial variations in one sensor or actuator can affect other associated attributes within the same group. As a result, we observe that the graph structure learning with attention mechanism is more effective on datasets in which the elements are strongly related.

VII. GUIDELINES FOR PRACTITIONERS

Most current anomaly detection methods are highly specific to certain use cases. This means that there is no one-size-fits-all approach. In this respect, we provide guidelines for model selection according to the purpose and the circumstances of each application. Intuitive visualizations of our guidelines are provided in Fig. 11. We also discuss the training techniques that should be considered.

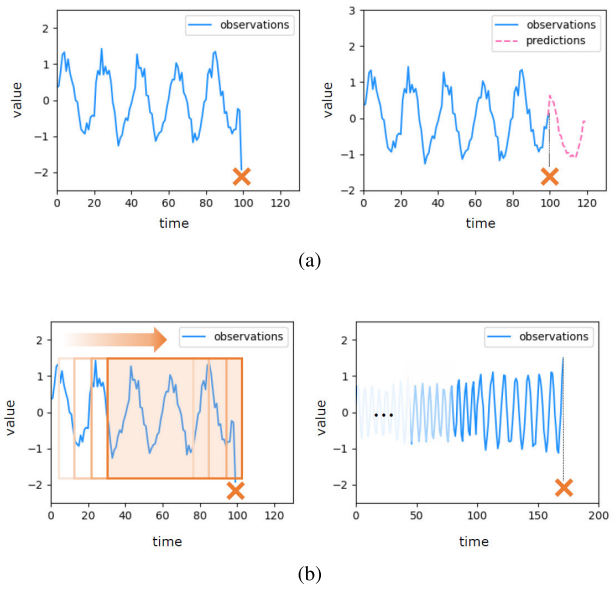


FIGURE 11. Strategies for anomaly detection in time-series data: (a) real-time vs. early-warning; (b) sliding windows vs. incremental update.

A. DETECTION STRATEGIES

Time-series data is not very different from data in other domains. However, there are unique properties of the time series structure, and there are several aspects of the environment in which the data are generated and analyzed that could affect the success of an anomaly detection algorithm.

1) REAL-TIME vs. EARLY WARNING

When an anomaly can incur severe damage, an early warning method can alert the removal of potential factors in advance. On the other hand, real-time detection methods are beneficial because only the actual anomalies are processed. Hence, it could reduce unnecessary costs caused by false positives.

- *Real-Time:* Online business and finance require real-time anomaly detection to respond quickly to incidents [130]. Also, monitoring manufacturing equipment in real-time is mandatory to reliably maintain a manufacturing capacity. Recently, cyber-physical systems (CPS) [131] have integrated physical and computational capabilities to remotely control substantial systems in real-time. They react immediately to dynamic changes and reduce human intervention. Generally, GRU- [60], [68], [113], [114] and CNN-based models [53], [62], [115] using reconstruction errors provide real-time anomaly detection capabilities. The inference time of each model can vary with computational complexity and computing resources. The models with high computational complexity take longer to make the

decision. Conversely, the models paired with extensive computation resources generally output the result faster. However, GRU is a type of the RNN that sequentially processes the observations. Thus, the inference time of the reconstruction-based models using GRU will be constant regardless of the computation resources unless data parallelism is not supported. Meanwhile, CNN-based reconstruction models handle given input data at once, and thus, they can process more features and longer sequences with sufficient computation resources.

- *Early Warning:* Maintenance costs in manufacturing plants constitute a substantial portion of the total production cost. Once a severe failure has occurred in facilities, the operators will lose vast amounts of time and costs due to an unscheduled downtime for repair. In this regard, a condition-driven preventive maintenance (PdM) [132] scheme has been introduced. Improved time-series anomaly detection algorithms that can predict future breakdowns are required to successfully perform PdM. Autoregressive algorithms that accumulate historical information in their model can predict possible faults. In particular, LSTM- [110], [111] and HTM-based models [121] have been widely used to predict faults in time-series data. The main challenges in anomaly prediction include false alarms and missed anomalies [133], [134]. Therefore, selection of an optimal threshold for anomaly detection is particularly important. A higher threshold value will suppress false alarms, but may miss the actual anomalies. On the contrary, a lower threshold will capture more anomalies but result in more false alarms.

2) SLIDING WINDOW vs. INCREMENTAL UPDATE

There are two propositions to infer context from time-series data. A time-series model either processes all of the historical data points or incrementally update the outputs for the newest items. These approaches are called sliding windows and incremental updates, respectively.

- *Sliding Window:* Some models can only feed-forward data of fixed sizes. TCN- [116]–[118] and CNN-based methods [53], [62], [115] fall into this category, and the size of the window affects the length of the temporal dependency modeled by the neural network. Therefore, practitioners should carefully choose an appropriate window size depending on the nature of the dataset (e.g., time lags between multivariate series and the frequency of subsequent anomalies). Excessive window sizes can cause anomalies to be overlooked, whereas insufficient window sizes can render the model incapable of capturing long-term dependencies. For example, Zhang *et al.* [51] compared the anomaly detection performance for varying window sizes, and chose the optimal value showing the maximum performance.
- *Incremental Update:* Incremental models update the predictions for new data via marginal computations. They are particularly beneficial in streaming environments in which data items are supplied one-by-one. Moreover, the computational benefits should not be underestimated. Methods based on sliding windows must maintain the entire data stream in memory for additional processing, which involves larger computations at each timestep. Autoregressive models, such as GRUs and LSTMs, are inherently incremental models because they maintain a compact summary of past data in their hidden states. For instance, some of the LSTM-based methods [110], [111] support incremental updates. However, many methods [29], [51], [69], [104] require references to past data for pre- or post-processing using AEs or other networks. For these methods, the incremental features are limited.

B. TRAINING AND PREPROCESSING TECHNIQUES

In addition to the detection phase, anomaly detection methods have a wide range of design choices for training.

1) LOSS FUNCTION

Time-series anomaly detection models are trained using different types of loss functions depending on how they model the normality of the data. The types of loss functions include an adversarial loss, a reconstruction loss, a prediction loss, and a negative log-likelihood.

- *Adversarial Loss:* Since the pioneering work of Goodfellow *et al.* [135], adversarial formulation has been widely used [136], [137] to improve the modeling capability of generative modules. This technique was also adopted in previous studies [59], [74], [115] for time-series anomaly detection. The discriminator primarily serves as a helper for the generative component. After training, it can also be used to generate anomaly score, as in [53], [59]. A typical adversarial formulation is given as the following two-player game:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x\sim p_{data}(X)}\left[\log D(x)\right] + \mathbb{E}_{z\sim p_z(Z)}\left[\log\left(1 - D(G(z))\right)\right], \tag{19}$$

  where D and G are the discriminator and generator modules, respectively. Although the results generated by the models trained with an adversarial loss can be remarkable, the most challenging issue is that the simultaneous dynamic training of two competing models is inherently unstable. Due to the unstable training process, the models may fall into failure modes instead of converging to the optima. A typical failure mode is a *mode collapsing* that the generator always outputs the same value from multiple inputs.
- *Reconstruction Loss:* AE is a preferred choice for anomaly detection, provided that AE trained with normal training data reconstructs normal data well. Several methods [29], [51], [104], [114] use AEs, optionally in conjunction with other modules. They use reconstruction losses as training loss functions, so that so that the

AE is trained to capture the normality of the training data. A typical reconstruction loss takes the following form:

$$\mathcal{L}(S_t, S_t') = \left\| S_t - S_t' \right\|, \tag{20}$$

where S_t is the observed data point at time step t, and S_t' is the reconstructed data point at timestep t.

- *Prediction Loss:* Prediction-based approaches detect anomalies by comparing the predictions with real observations [100], [111]. The prediction model is trained using a prediction loss so that the model is forced to produce an accurate prediction using past data or relationship among features. The prediction loss is similar to (20), except that S_t' indicates the prediction for the real observation S_t. This training scheme is applied in the inference time as is, and thus is beneficial for the early warning system.
- *Negative Log-Likelihood:* A group of generative models that can estimate the log-likelihood of input data commonly uses negative log-likelihood (NLL) as a training loss. Minimizing NLL maximizes the estimated likelihood of a dataset such that the model captures the notion of normality present in the dataset. GMMs are a type of generative model [103], [117] that includes NLL in their loss functions. Note that the NLL is optimized in different ways. For examples, TCN-GMM [117] maximizes the log-likelihood presented in (21) using the expectation-maximization algorithm.

$$J(\theta) = \sum_{k=1}^{K} w_k \frac{1}{(2\pi)^{D/2} |\Sigma_k|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mu_\mathbf{k})^T \Sigma_k^{-1} (\mathbf{x}-\mu_\mathbf{k})}, \tag{21}$$

where θ indicates the GMM parameters, $\{\Sigma_k, \mu_k, w_k\}_{k=1}^K$, and D is the number of dimensions in the feature vectors. In contrast, DAGMM [103] maximizes a similar loss term that uses gradient descent in an end-to-end fashion.

VAE, another class of generative models, is trained with an evidence lower bound (ELBO), as in (22), which is a lower bound of the log-likelihood. VAE-based methods [73], [112], [113] use ELBO for training.

$$J(\theta) = \mathbb{E}_q \left[\log p(x, z) \right] - \mathbb{E}_q \left[\log q(z) \right]. \tag{22}$$

They do not simply generate a data instance similar to the input but also approximate the unknown prior distribution using training data.

2) BATCH LEARNING vs. ONLINE UPDATE

A common challenge in time-series data is the nonstationary nature of data, as discussed in Section II-B. Following the changes in data distribution, we suggest two types of approaches to updating the model accordingly.

- *Batch Learning:* Deep learning typically assumes a stationary distribution of data, and deep neural network models are trained using a large batch of data sampled from the same distribution as the test distribution. Therefore, most deep learning-based methods should provide a new batch of training data to fine-tune the model. This training scheme may be problematic when the system administrator cannot re-collect data after each data update.
- *Online Update:* The above problem can be mitigated when the model supports online update. It enables fine-tuning of the model with newly appended data without the need to re-train the model from scratch. HTM-based methods have such capabilities [69], [121], but online updates are rarely found in deep-learning models because the nonstationary assumption of data distribution is rather unconventional in machine learning. Among deep learning-based approaches, some methods [110], [111] adjust their thresholds for binary decision-making.

  We can consider continual learning as an alternative. Continual learning, however, suffers from the plasticity-stability dilemma. Neural networks are known to do well on forward-transfer, and thus the parameters should be plastic to learn a new task. At the same time, they should be stable not to forget the important features. However, a fine-tuning to new tasks makes the parameters rapidly forget what they previously learned. We call this phenomenon catastrophic forgetting [138]. Common approaches to mitigating catastrophic forgetting include regularization-, dynamic network architectures-, and memory replay-based methods. Representative methods for each approach include elastic weight consolidation [139], dynamically expandable network [140], and deep generative replay [141].

3) DENOISING

Noise in time-series data is an inevitable factor induced by sensors. Noise, which is hardly distinguishable from anomalies, may degrade the performance of anomaly detection. Therefore, diverse techniques have been proposed to make the model effectively learn the normality of the data by removing the noise in advance.

- *Smoothing:* The exponentially weighted moving average is a recursive smoothing filter that performs a scheme in which weight is assigned to the current observation the most and decays exponentially as one traverses the past. Although this method is effective, it has a problem that we should determine the level of denoising.
- *Transformation:* Signals bear representation in both the time and frequency domains. Wavelet transform and fast Fourier transform decompose signals into multiple resolutions to extract frequency characteristics. The difference between the transformed data and the original data is regarded as a noise.
- *Estimation:* Kalman filter removes noisy data by representing them in a state-space model and applying probabilistic estimation [142].

- *Deep Learning:* If the training dataset is small compared with the model capacity, the deep-learning model can memorize the dataset. Hence, the model learns the noise. In this case, it is difficult to distinguish between noise and anomalies. A denoising autoencoder is a general deep learning-based method that addresses this problem. It trains the AE to restore the original input by adding random noise. Thus, it does not reconstruct the input as is, but instead, robustly learns the representation of the features to prevent overfitting.

VIII. CONCLUSION

For many years, data-driven decisions have been made across businesses and industry to provide better products and services to a global community. Analytical techniques for extracting beneficial information from large volumes of data collected from various sources offer many opportunities. Moreover, identifying and troubleshooting unexpected events from time-series data can help prevent accidents and financial losses. Deep learning-based approaches have been attracting a considerable amount attention because of their incredible capability to resolve these problems.

In this paper, we discussed the characteristics of time-series data and the anomalies detected therein. We also described various applications of anomaly detection in several industries, including manufacturing, energy management, cloud infrastructure, and structural health monitoring. Because there has been a historical interest in anomaly detection in time-series data, we briefly presented some traditional approaches and described challenging issues regarding this topic. As the complexity of the system increases while the refined data and labels for analysis remain insufficient, the demand for unsupervised deep learning-based time series anomaly detection continues to increase. In this regard, we provide a review of the latest deep learning-based anomaly detection methods for time-series data from several perspectives and report the evaluation results on three real-world benchmark datasets. Finally, we finish with guidelines for model selection and training techniques.

REFERENCES

[1] J. P. Assendorp, "Deep learning for anomaly detection in multivariate time series data," Ph.D. dissertation, Dept. Comput. Sci., Hochschule für angewandte Wissenschaften Hamburg, Hamburg, Germany, 2017.
[2] A. Blázquez-García, A. Conde, U. Mori, and J. A. Lozano, "A review on outlier/anomaly detection in time series data," 2020, *arXiv:2002.04236*. [Online]. Available: http://arxiv.org/abs/2002.04236
[3] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang, "A novel anomaly detection scheme based on principal component classifier," Dept. Elect. Comput. Eng., Univ. Miami, Coral Gables, FL, USA, Tech. Rep., Jan. 2003.
[4] F. Angiulli and C. Pizzuti, "Fast outlier detection in high dimensional spaces," in *Proc. Eur. Conf. Princ. Data Mining Knowl. Discovery (PKDD)*. Berlin, Germany: Springer, 2002, pp. 15–27.
[5] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2000, pp. 93–104.
[6] B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Comput.*, vol. 13, no. 7, pp. 1443–1471, 2001.
[7] M. Hu, X. Feng, Z. Ji, K. Yan, and S. Zhou, "A novel computational approach for discord search with local recurrence rates in multivariate time series," *Inf. Sci.*, vol. 477, pp. 220–233, Mar. 2019.
[8] K. Yan, W. Li, Z. Ji, M. Qi, and Y. Du, "A hybrid LSTM neural network for energy consumption forecasting of individual households," *IEEE Access*, vol. 7, pp. 157633–157642, 2019.
[9] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," 2019, *arXiv:1901.03407*. [Online]. Available: http://arxiv.org/abs/1901.03407
[10] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Müller, "Deep learning for time series classification: A review," *Data Mining Knowl. Discovery*, vol. 33, no. 4, pp. 917–963, Jul. 2019.
[11] A. A. Cook, G. Mısırlı, and Z. Fan, "Anomaly detection for IoT time-series data: A survey," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6481–6494, Jul. 2020.
[12] P. D. Talagala, R. J. Hyndman, K. Smith-Miles, S. Kandanaarachchi, and M. A. Muñoz, "Anomaly detection in streaming nonstationary temporal data," *J. Comput. Graph. Statist.*, vol. 29, no. 1, pp. 13–27, Jan. 2020.
[13] M. Goldstein and S. Uchida, "A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data," *PLoS ONE*, vol. 11, no. 4, Apr. 2016, Art. no. e0152173.
[14] C. C. Aggarwal, "Outlier analysis," in *Data Mining*. New York, NY, USA: Springer, 2015, pp. 237–263.
[15] D. M. Hawkins, *Identification of Outliers*, vol. 11. London, U.K.: Chapman & Hall, 1980.
[16] Y. Bao, Z. Tang, H. Li, and Y. Zhang, "Computer vision and deep learning–based data anomaly detection method for structural health monitoring," *Struct. Health Monit.*, vol. 18, no. 2, pp. 401–421, 2019.
[17] Z. Tang, Z. Chen, Y. Bao, and H. Li, "Convolutional neural network-based data anomaly detection method using multiple information for structural health monitoring," *Struct. Control Health Monitor.*, vol. 26, no. 1, p. e2296, Jan. 2019.
[18] J. D. Hamilton, *Time Series Analysis*. Princeton, NJ, USA: Princeton Univ. Press, 2020.
[19] J. Kusuma, L. Doherty, and K. Ramchandran, "Distributed compression for sensor networks," in *Proc. Int. Conf. Image Process.*, vol. 1, 2001, pp. 82–85.
[20] I. Sluiter and R. M. Rosen, "General introduction," in *Aesthetic Value in Classical Antiquity*. Leiden, The Netherlands: Brill, 2012, pp. 1–14.
[21] G. Widmer and M. Kubat, "Learning in the presence of concept drift and hidden contexts," *Mach. Learn.*, vol. 23, no. 1, pp. 69–101, 1996.
[22] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
[23] V. Tuzlukov, *Signal Processing Noise*. Boca Raton, FL, USA: CRC Press, 2018.
[24] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014, *arXiv:1412.6572*. [Online]. Available: http://arxiv.org/abs/1412.6572
[25] A. Ajami and M. Daneshvar, "Data driven approach for fault detection and diagnosis of turbine in thermal power plant using independent component analysis (ICA)," *Int. J. Electr. Power Energy Syst.*, vol. 43, no. 1, pp. 728–735, Dec. 2012.
[26] D. Kim, H. Yang, M. Chung, S. Cho, H. Kim, M. Kim, K. Kim, and E. Kim, "Squeezed convolutional variational AutoEncoder for unsupervised anomaly detection in edge device industrial Internet of Things," in *Proc. Int. Conf. Inf. Comput. Technol. (ICICT)*, Mar. 2018, pp. 67–71.
[27] K. Leahy, R. L. Hu, I. C. Konstantakopoulos, C. J. Spanos, and A. M. Agogino, "Diagnosing wind turbine faults using machine learning techniques applied to operational data," in *Proc. IEEE Int. Conf. Prognostics Health Manage. (ICPHM)*, Jun. 2016, pp. 1–8.
[28] X. Jin, Y. Sun, Z. Que, Y. Wang, and T. W. S. Chow, "Anomaly detection and fault prognosis for bearings," *IEEE Trans. Instrum. Meas.*, vol. 65, no. 9, pp. 2046–2054, Sep. 2016.
[29] R.-J. Hsieh, J. Chou, and C.-H. Ho, "Unsupervised online anomaly detection on multivariate sensing time series data for smart manufacturing," in *Proc. IEEE 12th Conf. Service-Oriented Comput. Appl. (SOCA)*, Nov. 2019, pp. 90–97.
[30] B. Luo, H. Wang, H. Liu, B. Li, and F. Peng, "Early fault detection of machine tools based on deep learning and dynamic identification," *IEEE Trans. Ind. Electron.*, vol. 66, no. 1, pp. 509–518, Jan. 2018.
[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 2, pp. 84–90, Jun. 2017.

[32] J. Cristian Borges Gamboa, "Deep learning for time-series analysis," 2017, *arXiv:1701.01887*. [Online]. Available: http://arxiv.org/abs/1701.01887

[33] C.-Y. Hsu and W. C. Liu, "Multiple time-series convolutional neural network for fault detection and diagnosis and empirical study in semiconductor manufacturing," *J. Intell. Manuf.*, vol. 32, pp. 1–14, Mar. 2020.

[34] E. Kim, S. Cho, B. Lee, and M. Cho, "Fault detection and diagnosis using self-attentive convolutional neural networks for variable-length sensor data in semiconductor manufacturing," *IEEE Trans. Semicond. Manuf.*, vol. 32, no. 3, pp. 302–309, Aug. 2019.

[35] K. B. Lee, S. Cheon, and C. O. Kim, "A convolutional neural network for fault classification and diagnosis in semiconductor manufacturing processes," *IEEE Trans. Semicond. Manuf.*, vol. 30, no. 2, pp. 135–142, May 2017.

[36] B. Lindemann, N. Jazdi, and M. Weyrich, "Anomaly detection and prediction in discrete manufacturing based on cooperative LSTM networks," in *Proc. IEEE 16th Int. Conf. Autom. Sci. Eng. (CASE)*, Aug. 2020, pp. 1003–1010.

[37] H. Wang, S. Li, L. Song, and L. Cui, "A novel convolutional neural network based fault recognition method via image fusion of multi-vibration-signals," *Comput. Ind.*, vol. 105, pp. 182–190, Feb. 2019.

[38] K.-P. Lee, B.-H. Wu, and S.-L. Peng, "Deep-learning-based fault detection and diagnosis of air-handling units," *Building Environ.*, vol. 157, pp. 24–33, Jun. 2019.

[39] H. Shahnazari, P. Mhaskar, J. M. House, and T. I. Salsbury, "Modeling and fault diagnosis design for HVAC systems using recurrent neural networks," *Comput. Chem. Eng.*, vol. 126, pp. 189–203, Jul. 2019.

[40] H. Wu and J. Zhao, "Fault detection and diagnosis based on transfer learning for multimode chemical processes," *Comput. Chem. Eng.*, vol. 135, Apr. 2020, Art. no. 106731.

[41] J. Pan, Y. Pottimurthy, D. Wang, S. Hwang, S. Patil, and L.-S. Fan, "Recurrent neural network based detection of faults caused by particle attrition in chemical looping systems," *Powder Technol.*, vol. 367, pp. 266–276, May 2020.

[42] B. Mamandipoor, M. Majd, S. Sheikhalishahi, C. Modena, and V. Osmani, "Monitoring and detecting faults in wastewater treatment plants using deep learning," *Environ. Monitor. Assessment*, vol. 192, no. 2, p. 148, Feb. 2020.

[43] M. Witczak, M. Mrugalski, M. Pazera, and N. Kukurowski, "Fault diagnosis of an automated guided vehicle with torque and motion forces estimation: A case study," *ISA Trans.*, vol. 104, pp. 370–381, Sep. 2020.

[44] S. C. Srivastava, A. K. Choudhary, S. Kumar, and M. Tiwari, "Development of an intelligent agent-based AGV controller for a flexible manufacturing system," *Int. J. Adv. Manuf. Technol.*, vol. 36, nos. 7–8, p. 780, 2008.

[45] M. Acosta and S. Kanarachos, "Tire lateral force estimation and grip potential identification using neural networks, extended Kalman filter, and recursive least squares," *Neural Comput. Appl.*, vol. 30, no. 11, pp. 3445–3465, Dec. 2018.

[46] T. Graber, S. Lupberger, M. Unterreiner, and D. Schramm, "A hybrid approach to side-slip angle estimation with recurrent neural networks and kinematic vehicle models," *IEEE Trans. Intell. Vehicles*, vol. 4, no. 1, pp. 39–47, Mar. 2018.

[47] A. Zhakov, H. Zhu, A. Siegel, S. Rank, T. Schmidt, L. Fienhold, and S. Hummel, "Automatic fault detection in rails of overhead transport systems for semiconductor fabs," in *Proc. 30th Annu. SEMI Adv. Semiconductor Manuf. Conf. (ASMC)*, May 2019, pp. 1–6.

[48] A. Zhakov, H. Zhu, A. Siegel, S. Rank, T. Schmidt, L. Fienhold, and S. Hummel, "Application of ANN for fault detection in overhead transport systems for semiconductor fab," *IEEE Trans. Semicond. Manuf.*, vol. 33, no. 3, pp. 337–345, Aug. 2020.

[49] O. Örnek, S. Vatan, S. Sarıoğlu, and A. Yazıcı, "Anomaly detection for autonomous transfer vehicles in smart factories," in *Proc. 6th Int. Conf. Control Eng. Inf. Technol. (CEIT)*, Oct. 2018, pp. 1–5.

[50] M. Fagiani, S. Squartini, M. Severini, and F. Piazza, "A novelty detection approach to identify the occurrence of leakage in smart gas and water grids," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2015, pp. 1–8.

[51] C. Zhang, D. Song, Y. Chen, X. Feng, C. Lumezanu, W. Cheng, J. Ni, B. Zong, H. Chen, and N. V. Chawla, "A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 1409–1416.

[52] S. Basumallik, R. Ma, and S. Eftekharnejad, "Packet-data anomaly detection in PMU-based state estimator using convolutional neural network," *Int. J. Electr. Power Energy Syst.*, vol. 107, pp. 690–702, May 2019.

[53] Y. Choi, H. Lim, H. Choi, and I.-J. Kim, "GAN-based anomaly detection and localization of multivariate time series data for power plant," in *Proc. IEEE Int. Conf. Big Data Smart Comput. (BigComp)*, Feb. 2020, pp. 71–74.

[54] A. Capozzoli, M. S. Piscitelli, S. Brandi, D. Grassi, and G. Chicco, "Automated load pattern learning and anomaly detection for enhancing energy management in smart buildings," *Energy*, vol. 157, pp. 336–352, Aug. 2018.

[55] J. Sipple, "Interpretable, multidimensional, multimodal anomaly detection with negative sampling for detection of device failure," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 9016–9025.

[56] C. Fan, F. Xiao, Y. Zhao, and J. Wang, "Analytical investigation of autoencoder-based methods for unsupervised anomaly detection in building energy data," *Appl. Energy*, vol. 211, pp. 1123–1135, Feb. 2018.

[57] A. P. Mathur and N. O. Tippenhauer, "SWaT: A water treatment testbed for research and training on ICS security," in *Proc. Int. Workshop Cyber-Phys. Syst. Smart Water Netw. (CySWater)*, Apr. 2016, pp. 31–36.

[58] C. M. Ahmed, V. R. Palleti, and A. P. Mathur, "WADI: A water distribution testbed for research in the design of secure cyber physical systems," in *Proc. 3rd Int. Workshop Cyber-Phys. Syst. Smart Water Netw.*, Apr. 2017, pp. 25–28.

[59] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, and S.-K. Ng, "MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks," in *Proc. Int. Conf. Artif. Neural Netw. (ICANN)*. Munich, Germany: Springer, 2019, pp. 703–716.

[60] L. Shen, Z. Li, and J. Kwok, "Timeseries anomaly detection using temporal hierarchical one-class network," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 13016–13026.

[61] V. Vercruyssen, W. Meert, G. Verbruggen, K. Maes, R. Baumer, and J. Davis, "Semi-supervised anomaly detection with an application to water analytics," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2018, pp. 527–536.

[62] T. Wen and R. Keyes, "Time series anomaly detection using convolutional neural networks and transfer learning," 2019, *arXiv:1905.13628*. [Online]. Available: http://arxiv.org/abs/1905.13628

[63] P. Filonov, A. Lavrentyev, and A. Vorontsov, "Multivariate industrial time series with cyber-attack simulation: Fault detection using an LSTM-based predictive data model," 2016, *arXiv:1612.06676*. [Online]. Available: http://arxiv.org/abs/1612.06676

[64] Y. Song and S. Li, "Gas leak detection in galvanised steel pipe with internal flow noise using convolutional neural network," *Process Saf. Environ. Protection*, vol. 146, pp. 736–744, Feb. 2020.

[65] A. Sari, "A review of anomaly detection systems in cloud networks and survey of cloud security measures in cloud storage applications," *J. Inf. Secur.*, vol. 6, no. 2, pp. 142–154, Apr. 2015.

[66] X. Zhang, J. Kim, Q. Lin, K. Lim, S. O. Kanaujia, Y. Xu, K. Jamieson, A. Albarghouthi, S. Qin, M. J. Freedman, and Y. Xiong, "Cross-dataset time series anomaly detection for cloud systems," in *Proc. USENIX Annu. Tech. Conf. (USENIX ATC)*, 2019, pp. 1063–1076.

[67] C. Huang, G. Min, Y. Wu, Y. Ying, K. Pei, and Z. Xiang, "Time series anomaly detection for trustworthy services in cloud computing systems," *IEEE Trans. Big Data*, early access, Jun. 1, 2017, doi: 10.1109/TBDATA.2017.2711039.

[68] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 2828–2837.

[69] N. Ding, H. Gao, H. Bu, and H. Ma, "RADM: Real-time anomaly detection in multivariate time series based on Bayesian network," in *Proc. IEEE Int. Conf. Smart Internet Things (SmartIoT)*, Aug. 2018, pp. 129–134.

[70] M. Munir, S. A. Siddiqui, A. Dengel, and S. Ahmed, "DeepAnT: A deep learning approach for unsupervised anomaly detection in time series," *IEEE Access*, vol. 7, pp. 1991–2005, 2018.

[71] T. Kieu, B. Yang, and C. S. Jensen, "Outlier detection for multidimensional time series using deep neural networks," in *Proc. 19th IEEE Int. Conf. Mobile Data Manage. (MDM)*, Jun. 2018, pp. 125–134.

[72] J. Audibert, P. Michiardi, F. Guyard, S. Marti, and M. A. Zuluaga, "USAD: Unsupervised anomaly detection on multivariate time series," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2020, pp. 3395–3404.

[73] H. Zhao, Y. Wang, J. Duan, C. Huang, D. Cao, Y. Tong, B. Xu, J. Bai, J. Tong, and Q. Zhang, "Multivariate time-series anomaly detection via graph attention network," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2020, pp. 841–850.
[74] F. Khoshnevisan, Z. Fan, and V. R. Carvalho, "Improving robustness on seasonality-heavy multivariate time series anomaly detection," 2020, *arXiv:2007.14254*. [Online]. Available: http://arxiv.org/abs/2007.14254
[75] S. Bhatia, B. Hooi, M. Yoon, K. Shin, and C. Faloutsos, "MIDAS: Microcluster-based detector of anomalies in edge streams," in *Proc. AAAI*, 2020, pp. 3242–3249.
[76] Y. Weng and L. Liu, "A collective anomaly detection approach for multidimensional streams in mobile service security," *IEEE Access*, vol. 7, pp. 49157–49168, 2019.
[77] J. P. Lynch, C. R. Farrar, and J. E. Michaels, "Structural health monitoring: Technological advances to practical implementations [scanning the issue]," *Proc. IEEE*, vol. 104, no. 8, pp. 1508–1512, Aug. 2016.
[78] M. Markou and S. Singh, "Novelty detection: A review-part 1: Statistical approaches," *Signal Process.*, vol. 83, no. 12, pp. 2481–2497, 2003.
[79] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," in *Proc. KDD Workshop*, Seattle, WA, USA, vol. 10, 1994, pp. 359–370.
[80] G. E. P. Box and D. A. Pierce, "Distribution of residual autocorrelations in autoregressive-integrated moving average time series models," *J. Amer. Statist. Assoc.*, vol. 65, no. 332, pp. 1509–1526, Apr. 1970.
[81] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results," *J. Transp. Eng.*, vol. 129, no. 6, pp. 664–672, Nov. 2003.
[82] G. E. P. Box and G. C. Tiao, "Intervention analysis with applications to economic and environmental problems," *J. Amer. Stat. Assoc.*, vol. 70, no. 349, pp. 70–79, Mar. 1975.
[83] C. A. Sims, "Macroeconomics and reality," *Econometrica, J. Econ. Soc.*, vol. 48, no. 1, pp. 1–48, Jan. 1980.
[84] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probab.*, Oakland, CA, USA, vol. 1, 1967, pp. 281–297.
[85] L. M. Manevitz and M. Yousef, "One-class SVMs for document classification," *J. Mach. Learn. Res.*, vol. 2, pp. 139–154, Dec. 2001.
[86] G. J. McLachlan and K. E. Basford, *Mixture Models: Inference and Applications to Clustering*, vol. 38. New York, NY, USA: Marcel Dekker, 1988.
[87] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. Kdd*, vol. 96, 1996, pp. 226–231.
[88] Z. Huang, "Extensions to the K-means algorithm for clustering large data sets with categorical values," *Data Mining Knowl. Discovery*, vol. 2, no. 3, pp. 283–304, Sep. 1998.
[89] X. Xu, N. Yuruk, Z. Feng, and T. A. J. Schweiger, "SCAN: A structural clustering algorithm for networks," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2007, pp. 824–833.
[90] H. Shiokawa, Y. Fujiwara, and M. Onizuka, "SCAN++: Efficient algorithm for finding clusters, hubs and outliers on large-scale graphs," *Proc. VLDB Endowment*, vol. 8, no. 11, pp. 1178–1189, Jul. 2015.
[91] L. Chang, W. Li, L. Qin, W. Zhang, and S. Yang, "pSCAN: Fast and exact structural graph clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 2, pp. 387–401, Feb. 2017.
[92] H. Shiokawa, T. Takahashi, and H. Kitagawa, "ScaleSCAN: Scalable density-based graph clustering," in *Proc. Int. Conf. Database Expert Syst. Appl. (DEXA)*. Regensburg, Germany: Springer, 2018, pp. 18–34.
[93] S.-S. Li, "An improved DBSCAN algorithm based on the neighbor similarity and fast nearest neighbor query," *IEEE Access*, vol. 8, pp. 47468–47476, 2020.
[94] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, 2008.
[95] T. White, *Hadoop: The Definitive Guide*. Sebastopol, CA, USA: O'Reilly Media, 2012.
[96] W. Zhao, H. Ma, and Q. He, "Parallel *K*-means clustering based on MapReduce," in *Proc. IEEE Int. Conf. Cloud Comput.* Berlin, Germany: Springer, Dec. 2009, pp. 674–679.
[97] X. Cui, P. Zhu, X. Yang, K. Li, and C. Ji, "Optimized big data K-means clustering using MapReduce," *J. Supercomput.*, vol. 70, no. 3, pp. 1249–1259, 2014.
[98] S. S. Bandyopadhyay, A. K. Halder, P. Chatterjee, M. Nasipuri, and S. Basu, "HdK-means: Hadoop based parallel K-means clustering for big data," in *Proc. IEEE Calcutta Conf. (CALCON)*, Dec. 2017, pp. 452–456.
[99] B. Bahmani, B. Moseley, A. Vattani, R. Kumar, and S. Vassilvitskii, "Scalable K-means++," 2012, *arXiv:1203.6402*. [Online]. Available: http://arxiv.org/abs/1203.6402
[100] M. S. Shahriar, D. Smith, A. Rahman, M. Freeman, J. Hills, R. Rawnsley, D. Henry, and G. Bishop-Hurley, "Detecting heat events in dairy cows using accelerometers and unsupervised learning," *Comput. Electron. Agricult.*, vol. 128, pp. 20–26, Oct. 2016.
[101] H. Lu, Y. Liu, Z. Fei, and C. Guan, "An outlier detection algorithm based on cross-correlation analysis for time series dataset," *IEEE Access*, vol. 6, pp. 53593–53610, 2018.
[102] T. Yu, X. Wang, and A. Shami, "Recursive principal component analysis-based data outlier detection and sensor data aggregation in IoT systems," *IEEE Internet Things J.*, vol. 4, no. 6, pp. 2207–2216, Dec. 2017.
[103] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, "Deep autoencoding Gaussian mixture model for unsupervised anomaly detection," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–19.
[104] N. Gugulothu, P. Malhotra, L. Vig, and G. Shroff, "Sparse neural networks for anomaly detection in high-dimensional time series," in *Proc. AI IOT Workshop Conjunct (ICML, IJCAI ECAI)*, 2018, pp. 1–8.
[105] Y. Zhou, R. Arghandeh, H. Zou, and C. J. Spanos, "Nonparametric event detection in multiple time series for power distribution networks," *IEEE Trans. Ind. Electron.*, vol. 66, no. 2, pp. 1619–1628, Feb. 2018.
[106] L. Zheng, Z. Li, J. Li, Z. Li, and J. Gao, "AddGraph: Anomaly detection in dynamic graph using attention-based temporal GCN," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 4419–4425.
[107] Q. He, Y. J. Zheng, C. L. Zhang, and H. Y. Wang, "MTAD-TF: Multivariate time series anomaly detection using the combination of temporal pattern and feature pattern," *Complexity*, vol. 2020, pp. 1–9, Oct. 2020.
[108] A. Deng and B. Hooi, "Graph neural network-based anomaly detection in multivariate time series," in *Proc. Conf. Artif. Intell. (AAAI)*, vol. 35, 2021, pp. 4027–4035.
[109] Z. Chen, D. Chen, X. Zhang, Z. Yuan, and X. Cheng, "Learning graph structures with transformer for multivariate time series anomaly detection in IoT," 2021, *arXiv:2104.03466*. [Online]. Available: http://arxiv.org/abs/2104.03466
[110] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom, "Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding," in *Proc. 24th Int. Conf. Knowl. Discovery Data Mining (ACM SIGKDD)*, Jul. 2018, pp. 387–395.
[111] N. Ding, H. Ma, H. Gao, Y. Ma, and G. Tan, "Real-time anomaly detection based on long short-term memory and Gaussian mixture model," *Comput. Electr. Eng.*, vol. 79, Oct. 2019, Art. no. 106458.
[112] D. Park, Y. Hoshi, and C. C. Kemp, "A multimodal anomaly detector for robot-assisted feeding using an LSTM-based variational autoencoder," *IEEE Robot. Automat. Lett.*, vol. 3, no. 3, pp. 1544–1551, Jul. 2018.
[113] Y. Guo, W. Liao, Q. Wang, L. Yu, T. Ji, and P. Li, "Multidimensional time series anomaly detection: A GRU-based Gaussian mixture variational autoencoder approach," in *Proc. Asian Conf. Mach. Learn.*, 2018, pp. 97–112.
[114] T. Kieu, B. Yang, C. Guo, and C. S. Jensen, "Outlier detection for time series with recurrent autoencoder ensembles," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 2725–2732.
[115] B. Zhou, S. Liu, B. Hooi, X. Cheng, and J. Ye, "BeatGAN: Anomalous rhythm detection using adversarially generated time series," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 4433–4439.
[116] Y. Cheng, Y. Xu, H. Zhong, and Y. Liu, "HS-TCN: A semi-supervised hierarchical stacking temporal convolutional network for anomaly detection in IoT," in *Proc. IEEE 38th Int. Perform. Comput. Commun. Conf. (IPCCC)*, Oct. 2019, pp. 1–7.
[117] J. Liu, H. Zhu, Y. Liu, H. Wu, Y. Lan, and X. Zhang, "Anomaly detection for time series using temporal convolutional networks and Gaussian mixture model," *J. Phys., Conf. Ser.*, vol. 1187, no. 4, Apr. 2019, Art. no. 042111.
[118] Y. He and J. Zhao, "Temporal convolutional networks for anomaly detection in time series," *J. Phys., Conf. Ser.*, vol. 1213, Jun. 2019, Art. no. 042050.
[119] H. Song, D. Rajan, J. Thiagarajan, and A. Spanias, "Attend and diagnose: Clinical time series analysis using attention models," in *Proc. Conf. Artif. Intell. (AAAI)*, vol. 32, 2018, pp. 1–8.
[120] H. Meng, Y. Zhang, Y. Li, and H. Zhao, "Spacecraft anomaly detection via transformer reconstruction error," in *Proc. Int. Conf. Aerosp. Syst. Sci. Eng.* Singapore: Springer, 2020, pp. 351–362.

[121] J. Wu, W. Zeng, and F. Yan, "Hierarchical temporal memory method for time-series-based anomaly detection," *Neurocomputing*, vol. 273, pp. 535–546, Jan. 2018.
[122] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[123] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014, *arXiv:1406.1078*. [Online]. Available: http://arxiv.org/abs/1406.1078
[124] S. Chang, Y. Zhang, W. Han, M. Yu, X. Guo, W. Tan, X. Cui, M. Witbrock, M. Hasegawa-Johnson, and T. S. Huang, "Dilated recurrent neural networks," 2017, *arXiv:1710.02224*. [Online]. Available: http://arxiv.org/abs/1710.02224
[125] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018, *arXiv:1803.01271*. [Online]. Available: http://arxiv.org/abs/1803.01271
[126] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," 2015, *arXiv:1506.04214*. [Online]. Available: http://arxiv.org/abs/1506.04214
[127] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*. [Online]. Available: http://arxiv.org/abs/1409.0473
[128] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, Long Beach, CA, USA, 2017, pp. 5998–6008.
[129] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," Tech. Rep., 2019.
[130] M. Toledano, I. Cohen, Y. Ben-Simhon, and I. Tadeski, "Real-time anomaly detection system for time series at scale," in *Proc. Workshop Anomaly Detection Finance (KDD)*, 2018, pp. 56–65.
[131] R. Baheti and H. Gill, "Cyber-physical systems," *Impact Control Technol.*, vol. 12, pp. 161–166, Mar. 2011.
[132] R. K. Mobley, *An Introduction to Predictive Maintenance*. Amsterdam, The Netherlands: Elsevier, 2002.
[133] V. Flovik, "How to use machine learning for anomaly detection and condition monitoring," in *Concrete Use Case for Machine Learning and Statistical Analysis*. Canada: Towards Data Science Inc., 2018.
[134] P. Kamat and R. Sugandhi, "Anomaly detection for predictive maintenance in industry 4.0—A survey," in *Proc. E S Web Conf.*, vol. 170, 2020, p. 02007.
[135] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014, *arXiv:1406.2661*. [Online]. Available: http://arxiv.org/abs/1406.2661
[136] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, *arXiv:1511.06434*. [Online]. Available: http://arxiv.org/abs/1511.06434
[137] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1125–1134.
[138] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," *Psychol. Learn. Motivat.*, vol. 24, pp. 109–165, Dec. 1989.
[139] K. James, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, and D. Hassabis, "Overcoming catastrophic forgetting in neural networks," *Proc. Nat. Acad. Sci. USA*, vol. 114, no. 13, pp. 3521–3526, Mar. 2017.
[140] J. Yoon, E. Yang, J. Lee, and S. Ju Hwang, "Lifelong learning with dynamically expandable networks," 2017, *arXiv:1708.01547*. [Online]. Available: http://arxiv.org/abs/1708.01547
[141] H. Shin, J. Kwon Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," 2017, *arXiv:1705.08690*. [Online]. Available: http://arxiv.org/abs/1705.08690
[142] S. Park, M.-S. Gil, H. Im, and Y-S. Moon, "Measurement noise recommendation for efficient Kalman filtering over a large amount of sensor data," *Sensors*, vol. 19, p. 1168, Mar. 2019.

KUKJIN CHOI received the B.S. degree in computer science and engineering from Sogang University, Seoul, South Korea, in 2013. He is currently pursuing the M.S. degree in electrical and computer engineering with Seoul National University, Seoul. He is also currently a Staff Software Engineer with the DIT Center, Samsung Electronics. His research interests include deep learning, anomaly detection, and time-series analysis.

JIHUN YI received the B.S. degree in electrical and computer engineering from Seoul National University, Seoul, South Korea, in 2017, where he is currently pursuing the Ph.D. degree in electrical and computer engineering. His research interests include deep learning, anomaly detection, and explainable AI.

CHANGHWA PARK received the B.S. and M.S. degrees in electrical and computer engineering from Seoul National University, Seoul, South Korea, in 2019 and 2021, respectively. He is currently a Research Engineer with AIRS Company, Hyundai Motor Group. His research interests include deep learning, domain adaptation, and self-supervised learning.

SUNGROH YOON (Senior Member, IEEE) received the B.S. degree in electrical engineering from Seoul National University, South Korea, in 1996, and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, CA, USA, in 2002 and 2006, respectively. From 2016 to 2017, he was a Visiting Scholar at the Department of Neurology and Neurological Sciences, Stanford University. He held research positions at Stanford University and Synopsys, Inc., Mountain View, CA. From 2006 to 2007, he was with Intel Corporation, Santa Clara, CA. He was an Assistant Professor with the School of Electrical Engineering, Korea University, from 2007 to 2012. He is currently a Professor with the Department of Electrical and Computer Engineering, Seoul National University. His current research interests include machine learning and artificial intelligence. He was a recipient of the SNU Education Award, in 2018; the IBM Faculty Award, in 2018; the Korean Government Researcher of the Month Award, in 2018; the BRIC Best Research of the Year, in 2018; the IMIA Best Paper Award, in 2017; the Microsoft Collaborative Research Grant, in 2017; the SBS Foundation Award, in 2016; the IEEE Young IT Engineer Award, in 2013; and many other prestigious awards.