Journal of Water Process Engineering 52 (2023) 103568

Contents lists available at ScienceDirect

Journal of Water Process Engineering

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

Deep *H2O*: Cyber attacks detection in water distribution systems using deep learning

Md Nazmul Kabir Sikder [a,b], Minh B.T. Nguyen [a], E. Donald Elliott [c], Feras A. Batarseh [b,d,*]

[a] Bradley Department of Electrical and Computer Engineering, Virginia Tech, Arlington, VA, USA
[b] Commonwealth Cyber Initiative, Virginia Tech, Arlington, VA, USA
[c] Yale Law School, New Haven, CT, USA
[d] Department of Biological Systems Engineering, Virginia Tech, Arlington, VA, USA

ARTICLE INFO

Keywords:
Water distribution systems
AI assurance
Anomaly detection
Concealed attacks
Deep learning

ABSTRACT

Water Distribution Systems (WDSs) leverage the recent technological advancements in sensor technologies and Cyber-Physical Systems (CPSs) for better processing, distribution, and delivery of clean water. Given the digital nature of CPSs, they can be vulnerable to different kinds of cyber threats, especially in cases where adversaries can conceal the state of the attack. If an adversary (state or non-state actor) successfully compromises a WDS, that could result in major destructive consequences to water quality, public health, and agricultural irrigation. This paper presents empirical Artificial Intelligence (AI)-based methods for detecting such concealed attacks in WDS. We present two Deep Learning (DL) models: Temporal Graph Convolutional Network (TGCN) with Attention, a supervised learning model, and High Confidence Auto-Encoder (HCAE), an unsupervised learning model. TGCN adopts Attention and Robust Mahalanobis Distance (RMD) metrics for robust and generalizable forecasting performance. HCAE uses customized hidden layers to improve classification performance compared to state-of-the-art approaches. Experiments are performed to evaluate the proposed models using the BATtle of the Attack Detection ALgorithms (BATADAL) dataset; founded on a Supervisory Control And Data Acquisition (SCADA) infrastructure. Additionally, we assess the performance of the two models against synthetically poisoned data generated from a Generative Adversarial Network (GAN). Both attack detection models show superior accuracy with attack detection, localization, and overall robustness against data poisoning. The results suggest that both the supervised and unsupervised models perform better attack detection with a ranking score of 0.845 and 0.933, respectively. Results also indicate that, among the two models, the unsupervised model performs better in detecting poisoned data (accuracy: 0.992) and has better generalizability. Experimental results are recorded, evaluated, and discussed.

1. Introduction

Recent unprecedented AI and sensor technology advancements are transforming all domains, including WDSs. With industrial revolution 4.0, WDS [1] is undergoing a significant digital transformation to enable data-driven monitoring and utility operations control. The addition of cyber elements aims to make these (CPSs) systems more effective. However, that comes with a trade-off, systems become increasingly vulnerable to security and safety threats [2,4]. For instance, the recent targeting of infrastructure in Ukraine reminds us of the risk of attacks on critical infrastructure, including cyberattacks on WDSs. According to UNICEF, as of May, 1.4 million people in eastern Ukraine were without access to safe water from public water systems.[1] Another example, in 2013, hackers seized control of a small Florida dam that released unprocessed water to nearby communities [4]. Besides that, on February 5th, 2021, a Florida water treatment plant (in Oldsmar, FL) was compromised. The hacker altered the levels of sodium hydroxide in the water - a chemical that would severely damage any human tissue it touches [5]. This event suggests that current WDS control utility operations are mainly exposed and continuously vulnerable to attackers,

* Corresponding author.
E-mail addresses: nazmulkabir@vt.edu (M.N.K. Sikder), mnguyen0226@vt.edu (M.B.T. Nguyen), e.donald.elliott@aya.yale.edu (E.D. Elliott), batarseh@vt.edu (F.A. Batarseh).

[1] www.peopleinneed.net/in-ukraine-we-are-supplying-water-to-tens-of-thousands-of-people-8921gp#we-are-supplying-water-to-tens-of-thousands-of-people-in-eastern-ukraine-.

https://doi.org/10.1016/j.jwpe.2023.103568
Received 14 October 2022; Received in revised form 3 February 2023; Accepted 7 February 2023

requiring sophisticated learning algorithms that can estimate the system's state, detect inconsistencies and outliers, and mitigate the harm done by such events.

AI has numerous advantages for detecting anomalies in WDSs, including those caused by cyberattacks which are harder for humans to detect than kinetic attacks. WDS human observers cannot detect all anomalies, and even when they become aware of an anomaly, they often misinterpret it. The frailties of human beings in interpreting data from complex systems have been cataloged by sociologist Charles Perrow in an important book [6] about the causes of nuclear accidents. The same kinds of human shortcomings in interpreting complex data are evident in after-action reports about the Deepwater Horizon oil spill, which, among other things, recommend increased use of AI to prevent future spills [7]. Humans have created systems that are too complex for them to monitor effectively, and they need help from AI. But the potential applications of the technologies that the authors describe are not limited to cyberattacks and WDSs. Many sewage spills result from leaks in antiquated underground systems that can go undetected. For example, in July 2020, an old pipe broke in New Haven, Connecticut. No one noticed until a citizen saw raw sewage on the street the next morning and called the authorities. Two million gallons of untreated sewage spilled into Long Island Sound over the next several days.[2]

1.1. Motivation

WDSs are a critical infrastructure that ought to be safe, secure, and reliable in delivering water for all intended purposes. Despite the stochastic nature of the WDSs operational processes, many Artificial Intelligence (AI) algorithms can help with early attack prediction or anomaly detection [8].

Attack detection models using AI require data representing the physical structure and the temporal behaviors of the WDSs. A WDS consists of a network of reservoirs, pumps, storage tanks, pipes, junctions, valves, and taps; usually spread across a geographical dimension.

In most cases, the models are developed using data streams from SCADA systems to classify if the system is running safely or not. SCADA collects real-time distributed field data measurements, including water flow rates, pump status, and pressure sensor readings, then transmits them (measurements) to a central server. Because of the complex interdependencies among different nodes, DL models are better suited to computationally represent the system [9]. Machine Learning (ML) models, including ensemble learning models, are a good choice for small and simple networks [10], however the increasing number of nodes in a network (such as in WDSs) creates non-linear relationships among them. According to Sikder et al. [10], DL models can capture non-linear relationships in a distributed network system more effectively when compared to ML models. Therefore, our work aims to address the security of WDSs by building robust and generalized DL models.

Despite the recent advancements in computing technology, WDS has security flaws because of its dependency on decade-old network devices. Therefore, an attacker can easily eavesdrop on the communication between the network and the central control system. Additionally, adversaries can send malicious attacks by spoofing sensor measurements, concealing the intended attacks from the operators' sight [11] - these attacks are also known as concealed attacks. Fortunately, such attacks have a digital and physical footprint in the network, such as sensor value deviation from the norm or water flow rate changes during high-demand hours. Statistically, these abnormal events can be considered anomalies, which a suitable learning algorithm can detect. For instance, what if the attacker conceals these anomalies or sensor measurements by sending replay attacks? A replay attack [12] occurs when a cyber criminal eavesdrops on secure network communication, intercept, and delay signals to misdirect the receiver. In stealthy attacks [13], since attackers conceal the physical layer, it becomes difficult to detect these attacks by using anomaly detection-based models. For example, Taormina et al. [14] presented difficulties associated with ML algorithms to detect concealed attacks in (WDSs) in comparison with DL algorithms. DL algorithms are key for estimations in such complex ecosystems. The BATADAL data that we use here consist of a WDS operator for C-Town Public Utility (CPU) with anomalous low or overflowing tank levels showing up during stealthy attacks [8].

1.2. Background of cybersecurity in WDS

Advanced WDSs constitute tens or hundreds of Internet of Things (IoTs) devices, including sensors and actuators; this complex connectivity makes the system vulnerable to malicious cyberattacks [15]. Cyberattacks are becoming increasingly sophisticated via minimally perturbed signals that go unnoticed to a human operator or a traditional expert system [15]. Therefore, intelligent algorithms (such as DL) are deemed necessary for detecting cyberattacks [16].

Cyber-attacks on water distribution systems can take many forms. According to a survey paper [3], one type is data poisoning, in which an attacker alters or corrupts data in a system to cause it to malfunction or present incorrect outcomes. Another type is ransomware, in which an attacker encrypts a system's data and demands a ransom payment in exchange for the decryption key. Other types of attacks on water distribution systems include denial-of-service attacks (i.e. an attacker floods a system with traffic to prevent legitimate users from accessing it), and unauthorized access (where an attacker gains access to a system without permission). These attacks can have serious consequences on water treatment and supply, including contamination, disruption of service, and financial loss to the WDS.

DL algorithms can be trained on large amounts of data to identify patterns and anomalies that may indicate a cyber-attack. For example, an AI algorithm can be trained to recognize network traffic patterns typical of a denial-of-service attack and then use this information to block similar traffic in the future [17]. AI-based intrusion detection systems (IDS) are also used to detect and prevent cyber-attacks on water distribution systems. These systems can use a combination of AI algorithms and rule-based systems to detect unusual activity in the network. Moreover, AI can be used to optimize the security of water distribution systems by automating many of the tasks that are currently performed manually. For example, AI-based systems can automatically update security configurations and patch vulnerabilities, reducing the risk of successful attacks [3].

It is important to note that the use of AI in cybersecurity research for water distribution systems is still in its early stages [3], and more research is needed to understand these technologies' capabilities and limitations fully. However, AI's potential to enhance these systems' security is significant, and it is expected that the use of AI in cybersecurity for water distribution systems will continue to grow.

1.3. Our contribution

DL models make decisions based on either a black-box environment or a non-deterministic approach. Without assessing robustness and generalizability, DL models' deployment stage becomes unreliable. We address these two key components:, robustness and generalizability, and present the Deep H_2O framework that consists of two models: TGCN with attention, a supervised model, and HCAE, an unsupervised model. Key contributions are the following:

- The study uses TGCN with attention and HCAE models to identify cybersecurity threats in WDS, considering both time-sequential and non-sequential aspects.
- TGCN with attention captures spatial correlations in graph data and global temporal dynamics, understands complex interdependencies

[2] www.nhregister.com/news/article/Save-the-Sound-investigating-after-15396322.php.

among sensor readings, and leverages Graph Convolutional Network (GCN) [18] [19], Gated Recurrent Unit (GRU) [20], attention mechanism [21,22,19], and Robust Manalanobis Distance (RMD) [23] for anomaly detection. TGCN with attention provided better performance than the baseline TGCN model.

- HCAE uses a dimensionality reduction technique, ANN-based DL architecture, custom hidden layers, and constraints for a robust and well-generalized model that achieves an improved attack classification performance when compared to baseline AE models.
- The study assesses both models using original and synthetically generated samples, compares the performance of the supervised and unsupervised models, and justifies which model is better during a cyber-physical attack in a WDS.
- To check for generalizability and robustness, a synthetic test dataset is generated using GAN [24].

In the manuscript, we introduce the following three research hypotheses:

1. Research Question 1 (AI Assurance): How do AI assurance [25] constraints, such as layer customization, attention, and RMD, improve models' performance?
2. Research Question 2 (Data Poisoning): Can models' generalizability in WDSs be tested using poisoned data generated by GANs?
3. Research Question 3 (Feature Localization): How can the two models localize features based on embedded and learned representations in a given feature space (i.e., in a water system)?

Our study presents a novel approach for detecting cyber-physical attacks in WDS using TGCN with attention and HCAE models. We incorporate AI assurance methods [26] and constraints to improve the performance of these models. We also evaluate the models on both original and synthetic datasets and compare their performance using various evaluation metrics such as: F1-score, precision, recall, and time-to-detection. Our results show that both models can detect attacks in a WDS with high accuracy and efficiency. TGCN with attention and HCAE models can be promising approaches for detecting cyber-physical attacks in WDS. We also explain attack localization by identifying the attacked node during "ATTACK" windows in a time-efficient manner.

The rest of the paper is structured as follows: we present related works in Section 2, we introduce methods in Section 3, specifically TGCN with attention, and then HCAE. Then we describe our experimental design for the study in Section 4. Section 5 provides results and explanations. Finally, Section 6 presents our concluding remarks.

2. Related work

Given the constantly growing use of CPSs, the uses of AI in CPSs in various applications also grow [27,28,29]. This section discusses the related DL approaches for CPSs, especially multivariate time series data. We provide related works for supervised and unsupervised models in CPSs and present adversarial data generation approaches using GANs. Finally, we present WDS security works using AI methods.

2.1. Temporal graph convolutional network

TGCN architecture was first designed and developed by Zhao et al. [30] for traffic prediction in 2018. The model helps to capture spatial and temporal relationships in feature space. It consists of GCNs and GRUs, which allow it to learn complex topological structures to capture spatial dependencies and the dynamic changes in the data to capture temporal dependencies. In 2020, Zhu et al. [31] proposed architecture on top of TGCN with an attention mechanism. This attention mechanism improves prediction accuracy by adjusting the importance of different time points and assembling global temporal information. We transfer this knowledge from their study and adopt an attention mechanism in the WDSs to improve forecasting. To our best knowledge, neither Zhao et al.'s [30], nor Zhu et al.'s [31] architecture has been applied to WDS before.

In another study, Covert et al. [32] succeeded in creating a different architecture of TGCN in 2019 for automatic seizure detection. Unlike the TGCN architecture proposed by Zhao et al. [30], their approach consists of several Spatio-Temporal Convolutional (STC) layers. The STC layer does not contain any GRU but a 1-dimensional convolution to each of its input sequences [32]. STC layer operates on the input sequences in a way that allows the graph topology to be maintained at each layer. It then refers to the adjacency matrix of structural time series to aggregate neighboring features [32]. This architecture is then applied for cyber-physical attack detection on the BATADAL dataset [33] with good results compared to the benchmark values. Tsiami et al. [33] proposed a one-stage prediction-based algorithm that uses Covert et al.'s TGCN for sensor prediction and MD for anomaly detection. Although the results are promising, the model can still be potentially improved by replacing the MD metric with the RMD metric.

2.2. Autoencoder

The use of DL methods for anomaly detection has recently achieved improvements in learning high-dimensional datasets [34]. To eliminate outliers and noise without prior knowledge, a deep AE can be a helpful model [35]. A book [36] on outlier detection discusses how AEs are a natural choice for outlier detection since they are often used to reduce multidimensional datasets. The AE model presented at [35] discovers high-quality nonlinear features. This approach includes splitting the input data into two sets to increase the robustness of the model. The work results show good performance since they distinguish between random anomalies and other structured corruptions in CPS data. Sun et al. [37] proposed a novel sparse representation framework that learns dictionaries based on the latent space of Variational AutoEncoder (VAE). This framework addresses the limitation of most existing algorithms that can handle large-scale and high-dimensional data. Their proposed model can obtain hidden information and extract more high-level features by playing the role of dimensionality reduction.

Another work [38] addresses unsupervised anomaly detection on high-dimensional data. This work aims to address the limitation in existing unsupervised anomaly detection approaches that suffer from decoupled model learning with conflicting optimization goals. This paper presented a Deep Autoencoding Gaussian Model (DAGMM) for unsupervised anomaly detection. DAGMM optimizes the deep autoencoder and mixture model parameters jointly to help with the parameter learning of the mixture model. This joint optimization helps the autoencoder further reduce reconstruction errors. Our study focuses on AE reconstruction error reduction and attack detection performance maximization.

2.3. Generative adversarial networks

Generative models, including GANs, provide a way to learn deep representations without extensively annotating training data [24]. The inspiration for this idea comes from the two-player sum game between neural networks, where they balance each other out with gains and losses. GAN consists of a generator and discriminator. The generator captures the potential distribution of real samples to generate new samples, and the discriminator determines which samples are fake by discriminating which of the generated samples are real samples as accurately as possible [39]. GAN models are necessary for many DL applications, such as security, data augmentation, and privacy preservation. One work [39] stated that generative models understand data perspective, using real data to fit the distribution parameters and produce new data using the learned distribution. Another paper [40] explained the GAN framework by applying a range of benchmark datasets. They used noise merely on the bottom layer of the generator

network. They claimed the samples resulting from their estimation method have somewhat high variance and produce competitive samples compared to the generative models in the literature. Their work did not require interference during the learning, allowing them to incorporate various functions into the model. However, the model has disadvantages. The discriminator must be synchronized well with the generator to avoid the probability of placing the generator in a small area of data space.

Furthermore, Zhou [41] introduced GAN on the BATADAL datasets to create a virtual testbed for WDSs. Their approach computes the membership distance between the dimensions and then divides the dimensions with a small distance into a group. Then, they obtained a larger quantity of attack sample control data by expanding the attack sample. Another paper, [42] addresses the imbalanced and missing sample data used for Intrusion Detection Systems (IDSs) to defend against CPS attacks. They proposed to generate synthetic samples using GANs so the IDS gets trained using them as well as the originals. Their results showed improvements in attack detection and model stabilization; however, they did not provide any direction for balancing data classes. In our study, we generate synthetic balanced data using GAN for testing our models' generalizability.

2.4. Security for WDSs

The security aspects of water distribution have a wide variety of potential solutions. Kadosh et al. [43] presented a one-classifier approach to detect attacks in WDSs. Their approach uses a Support Vector Data Description (SVDD) algorithm to classify normal vs. anomalous behavior. Min et al. [44] proposed an ANN-based DL algorithm to detect cyber attacks. Taormina et al. [14] developed an approach that uses AEs to detect and localize intrusion attacks in a WD. Zou et al. [45] proposed an event detection model to detect and mitigate water contamination. In their approach, they proposed a hybrid model that comprises an ANN and a Support Vector Machine (SVM) to detect the contamination events. Bagherzadeh et al. [46] evaluated the effects of different feature selection methods on enhancing the model prediction performance of total nitrogen in wastewater treatment plants. Furthermore, they analyzed the importance of different characteristics, namely time, climate, hydraulic flow, and wastewater characteristics, in predicting energy consumption [47]. Their study suggested that the Gradient Boosting Machine algorithm exhibits a better performance in forecasting energy consumption when compared to other ML algorithms. Mehrani et al. [48] proposed a hybrid model that combines a mechanistic model and ML model to predict the liquid *N20* concentrations; their results suggest that a hybrid model that combines a mechanistic model and an Artificial Neural Network (ANN) model performs better with limited availability of data. Additionally, a significant amount of work has been reported on approaches to detect attacks in CPS used in water treatment plants [49].

Furthermore, Yoong et al. [50] designed an ML framework that can detect physical and software-generated anomalies in continuous water treatment plants without false alarms. Adepu et al. [15] designed and developed an expert system, Distributed Attack Detection (DAD), that detects physical anomalies of a plant in real-time operations. This study is a succession of a prior work of Adepu et al. [51] where they developed an anomaly detection framework based on physical invariants derived for each stage of the plant design. Macas et al. [52] claimed that present water treatment plants are complex, and their spatio-temporal relations need to be explored further. The authors presented an unsupervised framework for anomaly detection called Attention-based Convolutional LSTM Encoder-Decoder (ConvLSTM-ED) to capture temporal dependencies. In another study, Zizzo et al. [53] developed an adversarial attacker model that can compromise a subset of sensors and validate existing anomaly detection models. In their study, the attacker manipulates the detector by hiding its presence. Similarly, Anthi et al. [54] generated adversarial samples using the Jacobian-based Saliency Map attack and explored how adversarial learning can target the supervised models. Testing anomaly detection performance using an adversarial attacker model is a popular approach in WDSs; however, based on our search, applying GANs as adversaries for testing the generalizability of the attack detection models in WDSs is a novel study.

3. Deep H_2O

Deep H_2O (Fig. 1) consists of two main parts, (1) TGCN with attention and (2) HCAE. The supervised model, TGCN with attention, performs well with time series samples and offers contextual anomaly detection; it is more aligned with the BATADAL case study, where sensor relations in the distribution system require comprehensive monitoring. Additionally, we expect that a WDS end-user (i.e., operator) might require AI models that consider the data samples as non-time series for specific application requirements (for example, missing data). Thus, we propose HCAE, an unsupervised model that works well with non-sequential data samples.

This section presents the design of TGCN with attention and HCAE models; discusses concealed attack detection in a WDS. We discuss the design choice of GAN to generate poisoned data. Fig. 1 presents the high-level overview of our proposed Deep H_2O framework that contains all the models together. We present all three models consecutively and their respective outcome selection processes.

3.1. C-town data description: BATADAL case study

The BATADAL competition simulates an intrusion attack in a fictional C-Town (Fig. 2a). The WDS in C-Town experiences anomalous behavior (concealed attacks) in its hydraulic components. The WDS in C-Town consists of 429 pipes, 388 junctions, 7 storage tanks, 11 pumps, 5 valves, and a reservoir. Participants from the competition developed algorithms that detect attacks in the shortest time possible. The algorithms aim to accurately detect true attacks (i.e., fewer false alarms) and locate nodes when the system has been altered during an attack [8]. In C-Town, the seven tanks guarantee water distribution and storage across the nodes (T1-T7). C-Town's WDS does not have abnormalities due to seasonal changes, resulting in a fixed rate of water consumption. A SCADA system collects data from the components of C-Town and work cohesively. Anomalous behaviors are shown using labeled physical abnormalities (Tables A.5 and B.6) that lead to harmful behaviors in the WDS (such as an overflow of a tank). Spotting these anomalies as quickly as possible will ensure the WDS does not suffer from preventable damage.

The water levels of the seven water tanks determine the operations of components in the five pumping stations (S1-S5), where one valve and eleven pumps are distributed. Also, nine Programmable Logic Controllers (PLCs) are located near their control components. These PLCs can send information about their statuses such as “ON” or “OFF”; the flow rate information passes through them as well as suctions and discharges pressures to the SCADA system since they are connected to the water level sensors, valves, and pumps. However, most PLCs receive information from other PLCs instead of being connected to the water level sensors and components directly involved with the control logic.

BATADAL simulation consists of three datasets. All the used data and model codes are available in the following GitHub repository[3].

- **Dataset 1:** Dataset 1 consists of time-series samples over a period of 12 months. The instances in this dataset are recorded during normal operation hours with no attacks. It has 44 features with 8,762 samples.
- **Dataset 2**: Table A.5 consists of samples recorded over a period of three months (April to June 2016). The samples in Dataset 2 have 44

[3] github.com/AI-VTRC/DeepH2O.

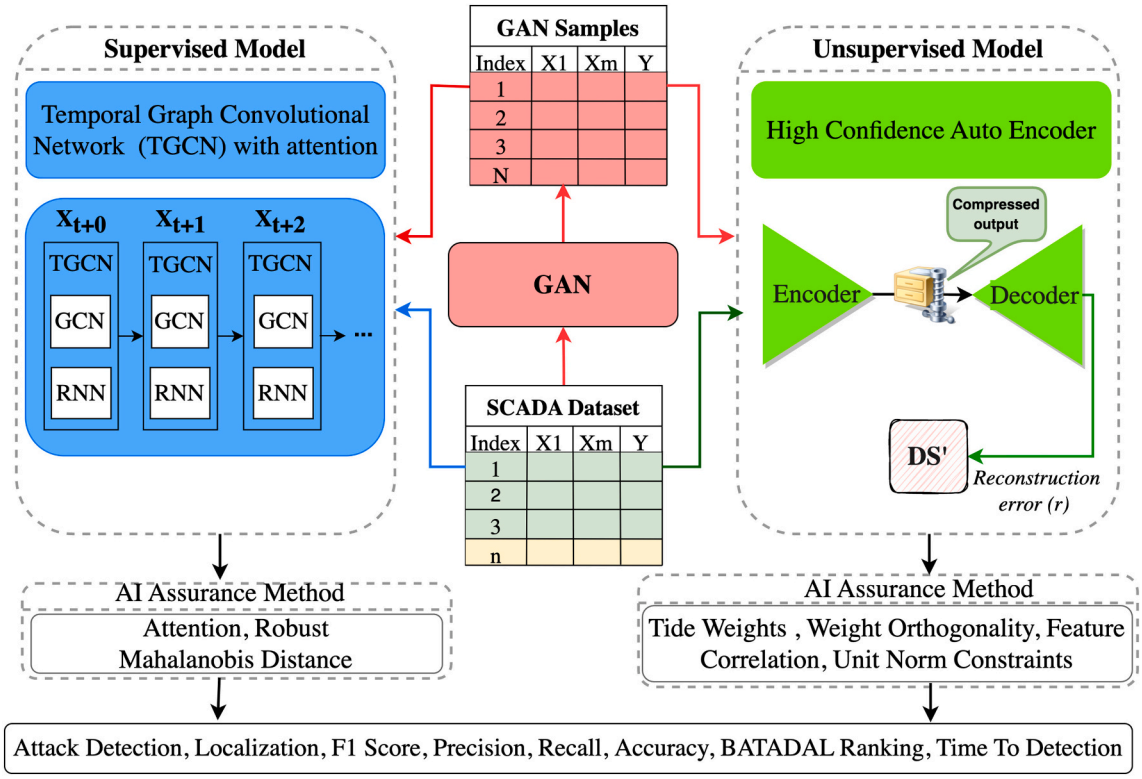
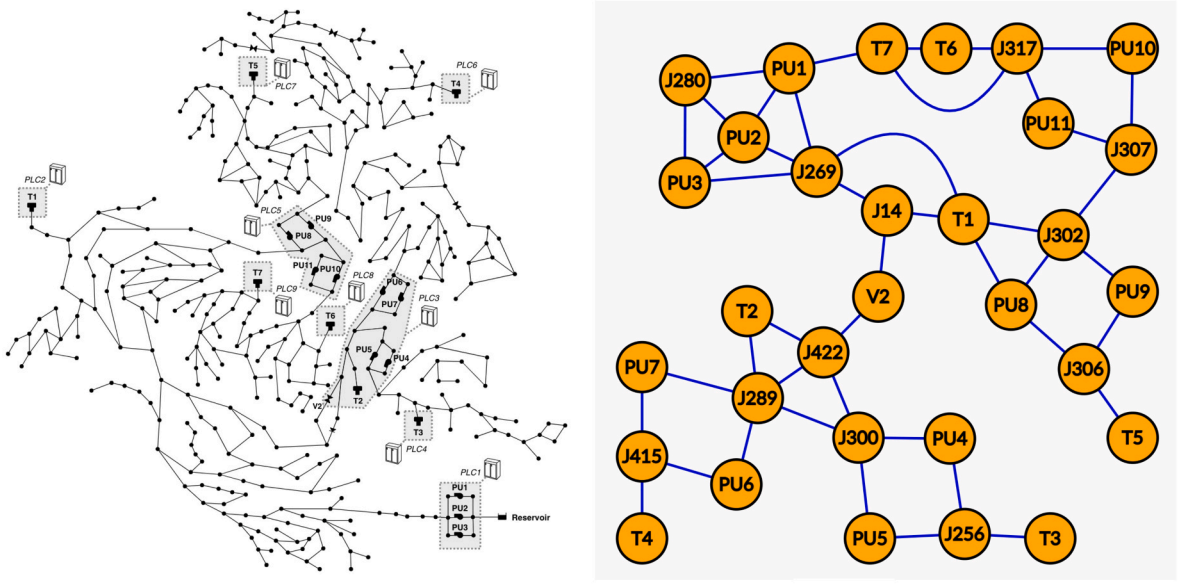


Fig. 1. Deep H_2O framework for cyber attack detection in WDSs.



(a) (b)

Fig. 2. WDS nodes representation [8] - Figure (a): Nodes layout of C-Town's distribution network; Figure (b): Graph data representation of reduced nodes (31 nodes) of C-Town.

features; the dataset consists of 4178 instances including the normal and attack samples.

- **Dataset 3**: This dataset also has 44 features, including the attack labels with 2090 samples. From Tables A.5 and B.6, it is evident that both Dataset 2 and Dataset 3 contains a different type of attack samples. Table B.6 contains three months of data from January 2017 to April 2017.

3.2. Supervised deep learning: TGCN with attention

This section explains the mechanism of TGCN with attention, RMD, and the attack detection topology. Tsiami et al. [33] proposed TGCN for the BATADAL competition. Since their model is not publicly available, we developed our baseline TGCN model, influenced by Tsiami's model architecture. We discuss the hyper-parameter choices in the design Section 4.2.1: Supervised Model Design. Next, we apply TGCN with

attention as a supervised prediction-based algorithm, where RMD is used for detecting outliers.

3.2.1. Attention temporal graph convolutional network

A structural WDS multivariate time series dataset has a form (X, A) when $X \varepsilon \mathbb{R}^{N \times P}$ contains N-dimensional observation for P different sequences and X_i denotes the systems' values at time i. We use a static graph technique since the number of dimensions N (N is the number of nodes) does not change over time. $A \varepsilon \{0,1\}^{P \times P}$ is the adjacency matrix of the P sequences. In TGCN with attention, we apply GCN and GRU to capture spatial and temporal dependencies, respectively, presented in the form of graph data. Furthermore, we apply an attention mechanism to capture global variation in the structural data. Fig. 3 shows the high-level architecture of TGCN with the attention mechanism. Each method working principle is discussed in the following subsections.

Spatial Dependence Modeling: We consider WDS as structural graph data where we consider each pump as a node and water flow rates as edges between nodes. GCNs are semi-supervised models that can process graph structure and capture spatial dependencies in the WDS graph structure. Recent works have shown that GCNs have achieved significant progress in applications such as image classification [18], fraud detection [55], and social analysis [56]. GCNs have spectrum and spatial domain convolutions [18,19]. Details about GCN can be found in Tsiami's [33] paper.

Temporal Dependence Modeling: The BATADAL dataset is considered a set of time-series samples for supervised modeling. A DL technique is introduced, such as GRU [20] for temporal dependence modeling. There exist other popular techniques for capturing temporal data, such as Recurrent Neural Network (RNN) and Long Short Term Memory (LSTM) [57]. RNN is simple and effective for time series modeling, but it has limitations in long-term forecasting due to gradient vanishing or gradient explosion [58]. LSTM and GRU are both proved [57] to be more efficient than RNN in long-term memory modeling. They use similar gated mechanisms, which allows them to perform similar tasks [59]. However, GRU has a simpler architecture than LSTM; GRU is more light-weighted (fewer gates than LSTM) and trains faster than LSTM [19]. Further details about GRU can be found in Zhu's [19].

Global Variation Modeling: Attention mechanism has been proven to be successful in image capturing generation [21], and recommendation systems [22]. It is applied because the supervised model is expected to capture the global variation trends for better accurate forecasting. It can learn the importance of sensors' values information at every time-stamp [19]. Here, we use a context vector to express the global variation trends of sensors' values for future sensors' values prediction.

Given a time series $x_i (i = 1, 2, \ldots, n)$ where n is the length of the time series, the attention technique process the data in four steps. Firstly, the hidden states $h_i (i = 1, 2, \ldots, n)$ at different moment are calculated using GRUs and expressed as $H = h_1, h_2, \ldots, h_n$. Next, we introduce a multilayer perceptron as "*attention score model*" to weigh the importance of each hidden state [19]. Later, the attention function calculates the context vector $C(t)$, which can express the global sensors' variation information. Lastly, output results are obtained using the context vector. We present the Eqs. (1)–(3) of the attention mechanism as follows:

$$e_i = w_{(2)} \left(w_{(1)} H + b_1 \right) + b_{(2)} \tag{1}$$

$$\alpha_i = \frac{exp(e_i)}{\Sigma_{k=1}^{n} exp(e_k)} \tag{2}$$

$$C_t = \Sigma_{i=1}^{n} \alpha_i^{*} h_i \tag{3}$$

The attention mechanism feeds hidden state (h_i) at each timestamp (also known as weight calculation) and calculates outputs after two hidden layers. A *Softmax*() function calculates the logits (α_i) using Eq. (1), where ($w_{(1)}$ and $b_{(1)}$) and ($w_{(2)}$ and $b_{(2)}$) are the weight and bias pairs of the first layer the second layer, respectively [19]. Eqs. (2) and (3) show the calculation of the global sensor's variation values of the context vector.

Combining these three methods, we arrive at the TGCN with attention framework, which can be represented as follows, Eqs. (4)–(7):

$$u_t = \sigma \left(W_u^{*} \left[GC(A, X_t) \right] h_{t-1} \right] + b_u) \tag{4}$$

$$r_t = \sigma \left(W_r^{*} \left[GC(A, X_t) \right] h_{t-1} \right] + b_r) \tag{5}$$

$$c_t = tanh \left(W_c \left[GC(A, X_t) \right], \left(r_t^{*} h_{t-1} \right) \right] + b_c) \tag{6}$$

$$h_t = u_t^{*} h_{t-1} + (1 - u_t)^{*} c_t \tag{7}$$

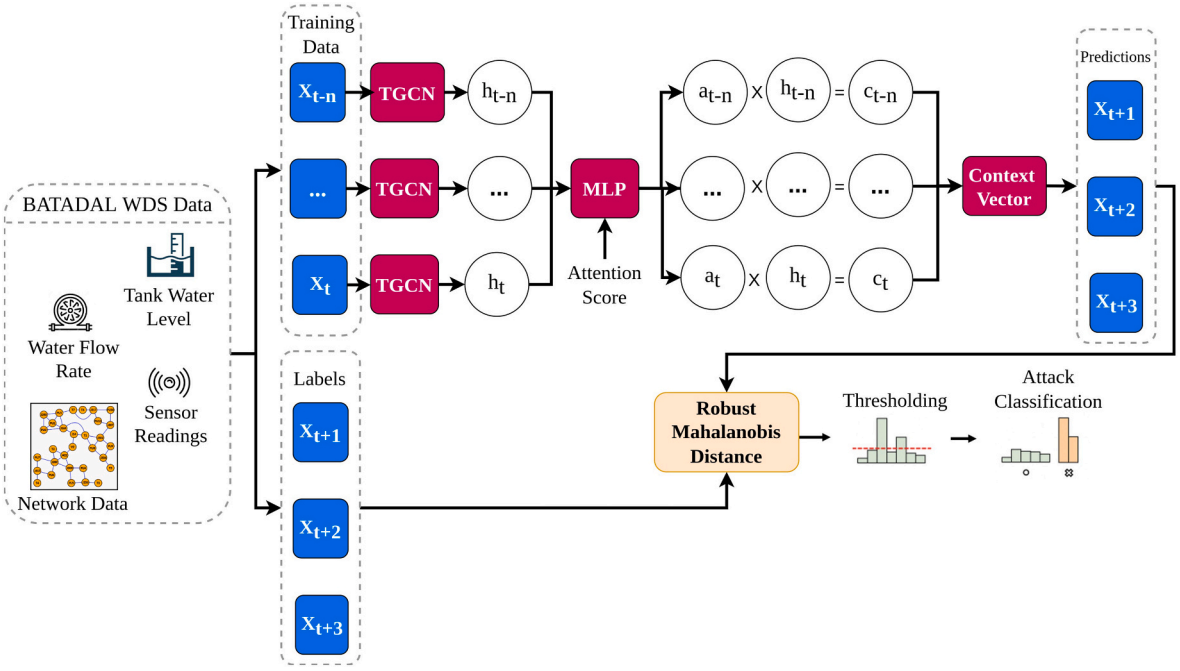


Fig. 3. TGCN with attention for WDSs cyber-physical attack detection.

Where, n sensors data are inputted into the TGCN with attention model to obtain n hidden state (h) that covered spatio-temporal characteristics: h_{t-n}, …, h_{t-1}, h_t; u_t and r_t represent update and reset gates at time t in Eqs. (4) and (5), respectively. c_t is the stored content at the current moment in Eq. (6). h_t is the output state at moment t, and W and b are the weight and deviation in the training process in Eq. (7). The hidden states are inputted into the attention mechanism to determine the context vector that represents the global distribution variation information for a given WDS. The weights of each hidden state h are calculated via *Softmax*() using a multilayer perceptron. Then the forecasting results are processed using a fully connected layer.

GCN has to ability to encode the topological structures, including pump and water flow information, by determining the relationship between a node sensor to its surrounding sensors [33]. Additionally, GRU determines sensors' values at the current time-stamp by using the hidden state at the previous time-stamp and the sensor's values at the current time-stamp as input. Finally, the attention mechanism re-weighs the influence of historical time series object states and captures the global variation trends of WDSs states for accurate forecasting.

3.2.2. Attack detection stage

TGCN with attention is capable of classifying whether a WDS is under "*ATTACK*" or "*NO ATTACK*." Our study presents the WDS network dataset as a graph with nodes (tanks, junctions, and pumps) linked with edges (pipes). Then the recorded readings are defined as structural time series that has the form (X,A) when $X \varepsilon \mathbb{R}^{N\times P}$ contains N-dimensional observation for P different sequences, and X_i denotes the systems' values at time i. $A \varepsilon \{0,1\}^{P\times P}$ is the adjacency matrix of the P sequences. By representing the input dataset in such a form, the model learns while capturing the spatio-temporal dependence among the feature space.

We develop the attack detection algorithm (Algorithm 1) in two stages. First, we train the model to predict the SCADA measurement under normal ("*NO ATTACK*") conditions. Then, we calculate the RMD distance metric and calibrate the model to classify all samples as "*NO ATTACK*" or "*ATTACK*."

3.2.3. AI assurance methods for supervised deep learning

For this stage, we explore AI assurance methods such as attention and RMD to improve the performance of the base model TGCN. TGCN can capture the spatio-temporal of the time series objects; however, the history of the data points is re-weighted with the addition of attention. Hence, this attention mechanism can capture the global variation trends of time series values for better forecasting accuracy.

Algorithm 1. TGCN/TGCN with attention Training and Testing.

Inputs: Dataset (X) and Model Arguments including *nI*, *nH*, *cf*, *activation*.

Execute: Robust Concealed Attack Detection and Localization.

1 Initialize the TGCN/TGCN with attention with pre-processed dataset X.

2 for k = 1,2,3,4,.... N_{Train} in Training set do.

3 Pass sample through Encoding, Latent representation, and Decoding layers (Eqs. (10), (11), (12)).

4 Use Adam Optimizer to minimize loss function (Eq. (13)).

5 return AE parameters θ.

6 Select threshold θ_{th}

7 for k = 1,2,3,4,.... N_{Test} in Testing set do.

8 Pass sample through encoding, latent representation, and decoding layers (Eqs. (10), (11), (12))

9 return reconstructions errors E_R.

10 Apply threshold $\theta_{Threshold}$ on reconstructions errors E_R for anomaly detection and localization

Next, we apply the outlier detection metric for the WDS system. Many distance metrics are available for outlier detection in multivariate time-series datasets, including Euclidean distance, MD, and RMD. MD is more effective than Euclidean distance due to the use of a covariance matrix to calculate the distance between data points and the center while detecting outliers according to the distribution pattern of the data points [10,60]. In brief, the covariance matrix in MD indicates how variables vary together. On the other hand, Euclidean distance does consider the data points' distribution pattern, which may assign some abnormal points as outliers and vice versa. MD carries two desired characteristics: it incorporates the dependencies between the prediction error at each sensor, which is helpful for combined unusual prediction errors, and it allows developers to tune a single global anomaly threshold instead of an individual threshold [33]. Given a Gaussian distributed data, the squared MD between data x_i to the center of the distribution is, Eq. (8):

$$d_{(\mu,\Sigma)}(x_i)^2 = (x_i - \mu)^T \Sigma^{-1} (x_i - \mu) \tag{8}$$

where μ is the mean and Σ is the covariance matrix of the data points. One drawback of MD is that its covariance matrix is highly sensitive to outliers, which is not preferable if the data is noisy. The Minimum Covariance Determinant (MCD) estimator proposed by P.J.Rousseuw was introduced as a more robust covariance estimator [61]. The idea behind MCD is to find the data points in which empirical covariance has the smallest determinant, thus giving a "*pure*" subset of data points from which to compute standards estimates of the mean and the covariance matrix [61]. The application of MCD to MD generates the RMD, which assures an improved concise outlier detection. Hence, we calculate the RMD measure for multivariate time series outlier detection.

3.2.4. Prediction and calibration of supervised deep learning

We divide the supervised DL model's attack detection and calibration processes [33] into three stages (Fig. 4). In the first stage, we preprocess and normalize data and train the TGCN with attention model with the normal training dataset labeled as "*NO ATTACK*," given the recorded SCADA measurements of n prior time-steps. A window of a fixed size $n+1$ rolls over the time series dataset with a step size of 1. Each window's first n measurements are the input to TGCN with attention, while the final $n+1$ data points are the target outputs.

Next stage, after training TGCN with attention, we pass the validation set X_{val} through the model for prediction $\widehat{Y}_{val}$. Since we pick normal samples for the training and the validation set, we assume that the prediction errors ($E = Y - \widehat{Y}$) at each sensor are roughly Gaussian distributed. Hence, we apply the squared MD (Eq. (8)). Afterward, we estimate the robust covariance matrix to calculate the RMD values. The squared MD is essentially the sum of p independent standard normal variables; thus, it follows a chi-squared distribution with p degrees of freedom [23]. During this stage, the predicted errors from the normal training dataset have lower RMD values (when compared to the threshold as in Eq. (9)) than the anomalous ones. The anomalies are detected at timestep i when:

$$\overline{d_{(\mu,\Sigma)}(x_i)^2}_{i-l,i} \geq TH \tag{9}$$

Here, $\overline{d_{(\mu,\Sigma)}(x_i)^2}_{i-l,i}$ is the mean squared RMD in a window of length l; and TH is the cut-off threshold. The calibration process is required to obtain the concise threshold TH and l. We select the thresholds by testing different values for the two parameters in a holdout set with few "*ATTACK*" data points and calculate the algorithm's performance with the S function (Eq. (27)).

In the third and final stage, we select parameter values such that it maximizes the rank S function on an unseen testing dataset. Finally, we localize attacked features such as pumps, sensors, and valves after classifying the timestamps that contain "*ATTACK*" data points.

3.3. Unsupervised deep learning: HCAE

This section discusses the unsupervised WDSs attack detection

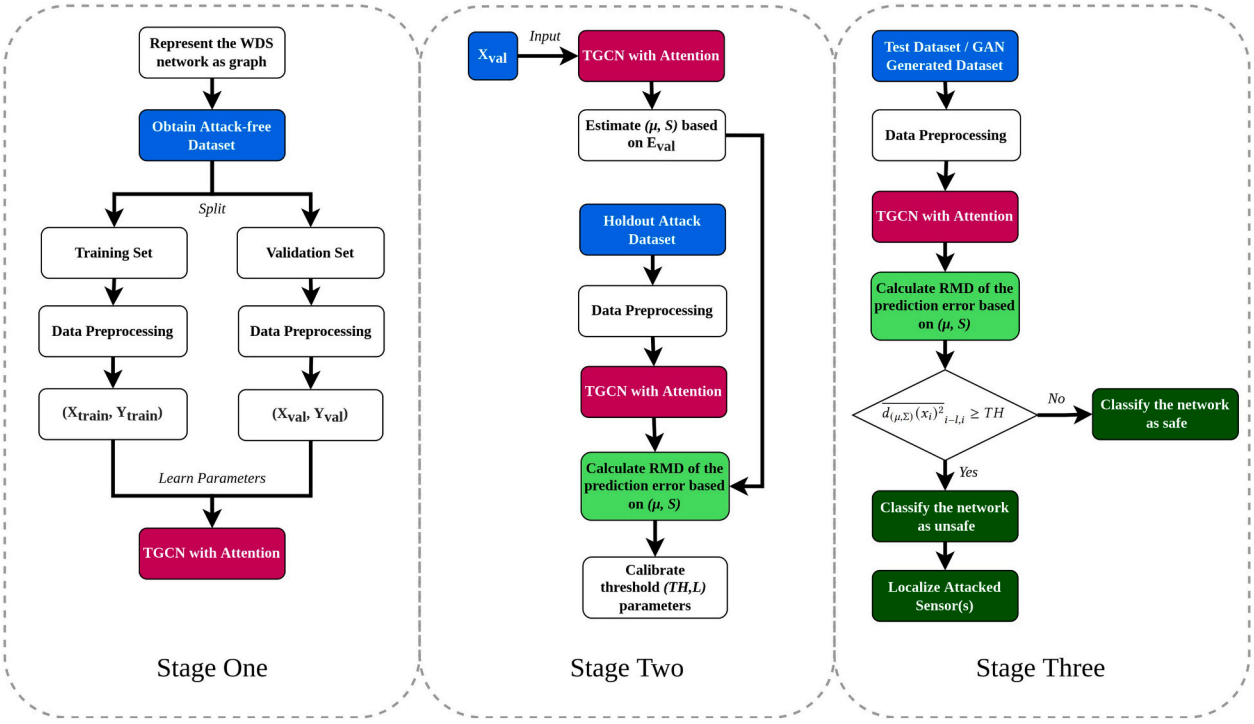


Fig. 4. TGCN with attention model development and attacks detection workflow [33].

framework, the mechanism of AE, and its revised version, HCAE. We apply AE as a reconstruction-based algorithm that performs dimensionality reduction and reconstructs the original input. The outcome from AE and HCAE are reconstruction errors (difference between output and input data), which identify physical anomalies from the feature space. Fig. 5 shows a fully connected ANN-based AE and its components. WDS data are fed to the AE and HCAE models. The models classify the inputs as either normal or anomalous samples based on a threshold.

3.3.1. Auto encoder

AEs have been a widely adopted DL method for the last couple of decades for both dimensionality reduction, and feature engineering [62]. We develop our baseline AE model by adopting Taormina's [14] AE model. Additional information about the design of our baseline AE model is discussed in the Unsupervised Model Design, Section 4.2.2.

The network is divided into two parts: an encoder function $h = f(X)$ and a decoder function $x' = g(f(x))$. AEs can be generalized as stochastic mappings of $P_{encoder} = (h|x)$ and $P_{decoder} = (x|h)$, where h is a hidden layer $h = f(x)$ that presents a code and is used to characterize the input. Multi-Layer Perceptrons(MLP) [63] form AEs with an input layer, an output layer, and multiple hidden layers. Mathematically an encoder and decoder can be written as Eqs. (10)–(12):

$$\phi : \mathscr{X} \rightarrow \mathscr{F} \tag{10}$$

$$\psi : \mathscr{F} \rightarrow \mathscr{X} \tag{11}$$

$$\phi, \psi = \underset{\phi,\psi}{argmin} \parallel \mathscr{X} - (\psi \circ \phi)\mathscr{X} \parallel^2 \tag{12}$$

where, Eqs. (10) and (11) represent encoder and decoder functionality respectively; Eq. (12) represents loss of the AE.

Input data $\mathscr{X}$ is transformed into a compressed representation $\mathscr{F}$ and reconstructed as $\mathscr{X}$ again. The objective of an AE is to minimize the reconstruction errors (Eqs. (13) and (14)), which yields a better reconstruction of the input set $\mathscr{X}$.

$$\mathscr{L}(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|^2 \tag{13}$$

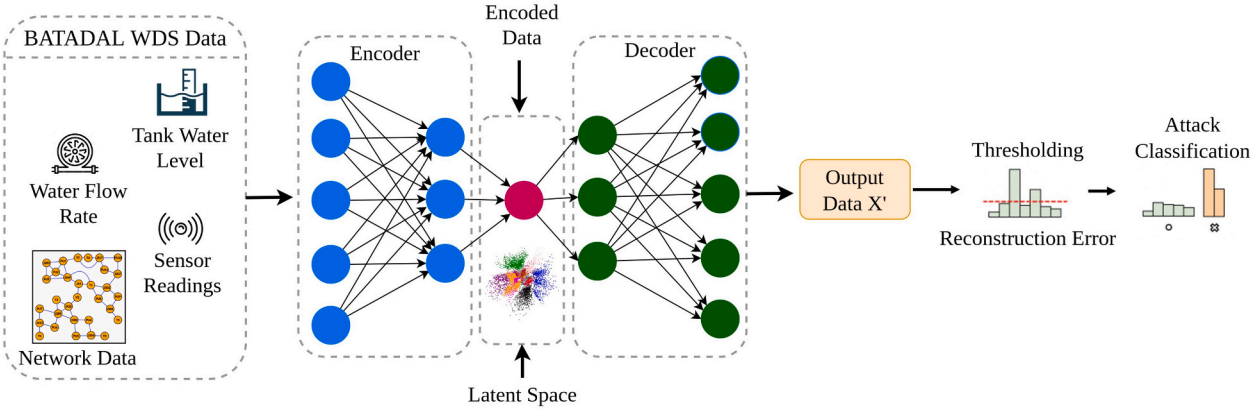


Fig. 5. Fully connected ANN-based Autoencoder for WDS cyber-physical attack detection.

$$\mathscr{L}(\mathbf{x}, \mathbf{x}^{'}) = \|\mathbf{x} - \sigma^{'}(\mathbf{W}^{'}(\sigma(\mathbf{W}\mathbf{x} + \mathbf{b}))) + \mathbf{b}^{'}\|^2 \quad (14)$$

Reconstruction errors are usually minimized using Stochastic Gradient Descent (SGD) [64], a potent optimization tool for many DL applications. However, our study applies the Adam optimizer, another powerful stochastic optimization method that outperforms SGD [65].

An anomaly detection system is expected to produce minimal false alarms, as false alarms are associated with expensive maintenance operations. Fig. 5 represents an ANN-based AE. Despite having fine-tuned hyper-parameters, AE suffers from non-determinism during training, resulting in a higher reconstruction error. A higher reconstruction error can result in an increased number of false positives, thus affecting the detector's performance [66]. As AE algorithms automatically learn features by performing feature engineering for dimensionality reduction, they tend to learn different features at each time [62]. This pattern of learning is suitable for systems where feature importance is unknown (for instance, thousands of sensor values in a water distribution system, complex and difficult feature space to human perception). However, such a non-deterministic learning pattern might not be suitable for a WDS. We expect a WDS model to provide, if possible, zero false alarms because of expensive maintenance operations. To address these issues, we revise AE architecture and form HCAE, thus solving the non-determinism problem of AE by further reducing reconstruction errors and improving attack detection performance by reducing false positives.

3.3.2. High confidence AutoEncoder (HCAE)

HCAE is a modified version of baseline AE; We developed HCAE by applying assurance methods (Eqs. (15)–(18)) to AE; improving the attack detection performance compared to the AE (baseline). We use HCAE to represent input features in a manifold space that generates minimum reconstruction errors while decoding and recreating the input features.

To minimize the reconstruction errors, the practitioners currently follow a trial-and-error based-method and optimize hyperparameters over multiple iterations. This approach is empirical; reducing the reconstruction errors is time-consuming and computationally expensive in a complex WDS. We apply a combination of neural network layer constraints for the model and achieve deterministic learning, which results in a manifold representation that yields minimum reconstruction errors. This strategy ensures learning a set of expected features from the training data each time. Resulted reconstruction errors yield better feature representation and attack detection performance [67]. We present our experimental results in the context of a WDS: how a set of constraints yields minimal reconstruction errors and robust attack detection. Algorithm 2 presents both algorithms, including baseline AE and HCAE training steps for attack detection in water systems.

3.3.3. AI assurance constraints for the auto encoder

Recent advancements in DL APIs, including Keras,[4] Tensorflow,[5] and Pythorch,[6] expedite AEs development more than ever. Nevertheless, a lack of a clear understanding of the fundamental properties of dimensionality reduction leads to a complex and inferior model. Thus, it is crucial to understand and adopt the basic properties of dimensionality reduction in AEs. We present multiple custom layer constraints and apply them to facilitate dimensionality reduction in a WDS. HCAE is effective for tuning and optimizing hyperparameters.

Algorithm 2. Baseline AE and HCAE Training and Testing.

Inputs: Dataset (*X*), Model Arguments including *nI*, *nH*, *cf*, *activation* and customized layers.

Execute: Robust Concealed Attack Detection and Localization.

1 Initialize AE/HCAE with pre-processed dataset *X*.

2 (This step is applicable for HCAE) Apply constraints on Encoder and Decoder layers using a combination of Tied Weights (Eq. (15)), Orthogonal Weights (Eq. (16)), Uncorrelated Features (Eq. (17)), Unit Norm (Eq. (18)).

3 For k = 1,2,3,4,.... N_{Train} in Training set do.

4 Pass sample through Encoding, Latent representation, and Decoding layers (Eqs. (10), (11), (12)).

5 Use Adam Optimizer to minimize loss function (Eq. (13)).

6 Return AE parameters θ_{params}.

7 Select threshold θ_{th}.

8 For k = 1,2,3,4,.... N_{Test} in Testing set do.

9 Pass sample through Encoding, Latent representation, and Decoding layers (Eqs. (10), (11), (12)).

10 Return reconstructions errors E_R.

11 Apply threshold θ_{th} on reconstructions errors E_R for anomaly detection and localization.

In order to improve the AEs detection performance, we apply the following set of constraints.

1. Tied Weights: Tied Weights [68] ensure equal weights for both encoder and decoder. This constraint also ensures easy learning, especially PCA-like dimensionality reduction and regularization. However, they do not always perform well on complex non-linear models. Again, tied weight constraint is not always necessary to continually improve the representation. If reconstruction errors are reasonable, the coding generates orthogonal latent features for given data. Such representation is helpful in dimensionality reduction and, eventually, for anomaly detection. In a multi-layer AE, weights vectors of layer l from an encoder and a decoder are transposed as Eq. (15).

$$W_l = W_{-l}^T \quad (15)$$

2. Orthogonal Weights: Each weight vector is independent; therefore, the weights of each encoding layer are orthogonal. The orthogonality constraints [69] act as regularization for the AE. Mathematically orthogonality condition for AE can be presented as,

$$W_{encoder}^T W_{encoder} = I \quad (16)$$

On applying, this constraint penalizes non-orthogonal weights. Depending on the dataset, the user can choose either orthogonal or non-orthogonal weights. Thus, the application of this constraint is conditioned on regularization.

3. Uncorrelated Features: If the output of the encoder is orthogonal, latent representations must be uncorrelated [70]. Hence, the output of the AE must have (Eq. (17)):

$$\text{correlation}\left(O_{\text{encoder}_i}, O_{\text{encoder}_j}\right) = 0 | i \neq j \quad (17)$$

4. Unit Norm: The weights of each layer must have unit norms [71]. This property helps to control exploding and vanishing gradients. Unit norm constraint (Eq. (18)) must be allied to all the layers of the AE.

$$\sum_{j=1}^{p} w_{ij}^2 = 1; i = 1, \ldots, k \quad (18)$$

These four constraints (Eqs. (15)–(18)), during model development, ensure the model does not create a sub-optimal decision boundary. They

[4] github.com/fchollet/keras
[5] github.com/tensorflow
[6] github.com/pytorch/pytorch

ensure the creation of a well-posed AE while constructing a high confident WDS model.

Unit norm and orthogonality solve regularization problems, especially for AEs, when AE learns from a training set but does not represent a test set well. Also, tide weight can reduce the number of parameters as a regularization technique. Here, the unit norm constraint addresses exploding gradients issue by bounding gradients into a finite value. Additionally, orthogonality resolves the vanishing gradients problem by assigning fewer nonzero weights, so only informative weights stand out. Thus, only these nonzero weights flow information during back-propagation and resolve the vanishing gradient issue [68], [69], [70], [71].

When we apply these four constraints (Eqs. (15)–(18)) while designing HCAE, we hypothesize the model will not converge to a suboptimal point. To test our hypothesis, we compare the results before and after using these four constraints; observe if attack detection performance (F1 score) is improved from the baseline AE [14]. We use the BATADAL dataset for training and testing our models.

Later in the manuscript, we present adversarial testing using a GAN to observe if the model can detect attacks from synthetically generated poisoned datasets (previously unseen data with a different distribution).

3.3.4. Attack detection and calibration stages of HCAE

HCAE is capable of classifying whether the WDSs are under "*ATTACK*" or "*NO ATTACK*" by investigating each sample. Nonetheless, the direct classification technique becomes erroneous with a small and imbalanced dataset because AE requires a large dataset to learn the representation. However, with HCAE's deterministic learning, we hypothesize the HCAE can detect attacks with minimum false positives. We test our hypothesis by training both AE and HCAE with the same imbalanced data and evaluate with total false positives for each model.

Unlike the supervised algorithm (TGCN with attention), we present data streams with non-sequential representation. Next, (X) is defined as $X \varepsilon \mathbb{R}^{N \times m}$ which contains N-dimensional observation for m different features; and X_i denotes the systems' values at time i.

The attack detection process is divided into two stages (Fig. 6). In stage one, we create custom HCAE layers by following Eqs. (15)–(18). We ensure the model always yields minimum reconstruction errors and maximum binary classification performance (F1 score) by observing the model performance on multiple hyperparameter sets. Later in stage two, data are preprocessed and normalized to have the maximal absolute value of each attribute, applied to all three provided datasets. After that, we train the HCAE model with the normal training dataset labeled "*NO ATTACK*." We split the training dataset into training (X_{train}, Y_{train}) and validation (X_{val}, Y_{val}) set. We apply early stopping, a regularization scheme when the model converges and starts to over-fit on the training dataset.

During each epoch, we compute loss (Eq. (15)), the squared of the reconstruction errors $r_e = |x - x'|$; minimize them using Adam optimizer. After both AE and HCAE models are well-trained, we select threshold θ_{th} in an empirical fashion. For that, the range of average reconstruction errors among all features in a sample is observed and summed up. Then we set the threshold based on the final range estimation, as shown in the following Eq. (19).

$$\theta_{th} = max\left\{ f(x) : \sum_{m} \frac{|x - x'|}{m} \text{ for} N_{training-samples} \right\} \tag{19}$$

The calibration process is crucial to derive a concise threshold θ_{th} for testing the model on new samples. If a test object is classified as "ATTACK," we localize the features associated with attacked attributes, such as pumps, sensors, and valves, using reconstruction errors.

For the HCAE model, we select hyperparameters that result in the best model's performance (F1-score). Our objective is to compare the performance of HCAE (AE with constraints) with the AE model (without constraints). To facilitate a fair comparison between HCAE and AE models, we retrain Taormina's AE model using the same hyperparameters of the HCAE model. We refer to the retrained AE model as the baseline AE model.

3.4. Synthetic WDSs attack data generation

Unlike other DL-based attack detection approaches that require significant domain knowledge and passive awareness of the attacked model [72], GANs are proven to be effective in generating realistic attack

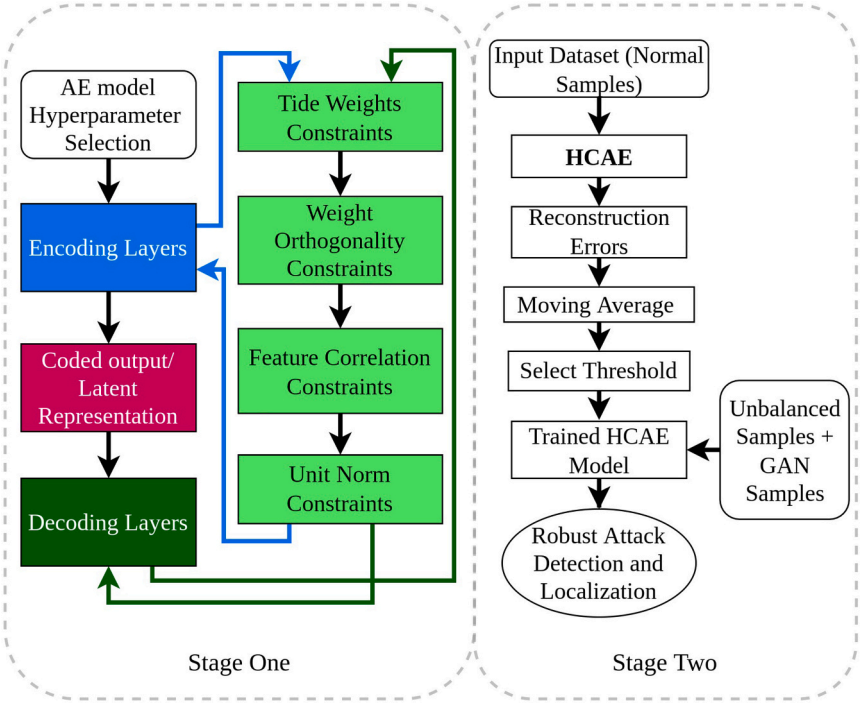


Fig. 6. HCAE model development and attack detection workflow.

samples (poisoned data) [73] with minimal information about either the domain or the DL model. We use GAN [40] for synthetic data generation. GAN network learns feature statistics of a given dataset for generating a new set of synthetic data. A generator produces the synthetic samples in the GAN network, and a discriminator evaluates them. The generator learns by mapping latent feature space to a data distribution of particular interest. Discriminator maximizes its objective by learning how to distinguish original samples from the generated fake samples. The generator aims to minimize the discriminator's objective by fooling it into thinking otherwise (fake samples as real ones). For instance, while generating a synthetic image set, GAN keeps the similar statistics of generated images set from the training set; hence, those generated images look superficially similar to human perception. During the training phase, both generator and discriminator play a minimax game where a bi-level optimization is performed to train the GAN network.

In our study, the generator learns the distribution of training samples X and maps data space as $G(z;\theta_g)$, where G is differentiable with respect to parameters θ_g. Then discriminator investigates if the data comes from the training samples and not from the generator itself. We train the discriminator to specify between original and generated samples from the generator correctly. Both the discriminator and the generator take participation in a minimax game which is represented as a value function as $V(G,D)$, as Eq. (20):

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(x)}[log D(\boldsymbol{x})] + \mathbb{E}_{z\sim p_z(z)}[log(1 - D(G(z)))] \quad (20)$$

Here, the generator is trained to maximize $\log D(\boldsymbol{x})$ and minimize $\log(1 - D(G(\boldsymbol{z})))$ in a numerical and iterable fashion. This GAN approach generates synthetic poisoned data for the WDSs system. We assess both supervised and unsupervised models on these synthetic data for generalization study.

4. Experimental design

This section presents the three research questions, and discusses the selection of hyperparameters and metrics used in our experiments.

In our experiments, supervised learning models are trained using two datasets: Dataset 1 and Dataset 2. As unsupervised learning requires "NO ATTACK" samples for training, the unsupervised models are trained with Dataset 1("NO ATTACK" samples). Both the supervised and the unsupervised models are evaluated using Dataset 3. We use total of 492 "ATTACK" samples from Dataset 3 to train the GAN model for synthetic data generation. Experiments are conducted to answer the following three research questions presented in Section 1.2.

4.1. Experimental models design

This section discusses the hyperparameter settings for TGCN, TGCN with attention, HCAE, AE, and GAN. This section presents the optimized hyperparameters set to train and evaluate the models. We also list all alternative hyperparameter values that are used to randomly search for the best models in appendix C, Table C.7.

4.1.1. Supervised model design

We pre-process the time series data as graph data and normalize them; during normalization, we perform standard minimum maximum scaling ranges from 0 to 1. The graph data can be represented as an adjacency matrix along with node data. Fig. 2b illustrates the node's information and locations in C-Town's WDS network. The adjacency matrix is constructed of the provided 31 nodes. Note that only one type of attribute is reported for each node in the network, except the pump nodes that have two different measurements: status and flow of water. We keep a single attribute for each node and exclude the binary pump status, where the numerical flow rate is preserved.

The supervised models: TGCN (baseline) and TGCN with attention, are implemented using PyTorch and Scikit-learn API.[7] The models are trained on a CPU with Intel core i5 10th gen.

We have tested both models with multiple iterations with different hyperparameter sets to obtain one. For TGCN, we choose an Adam optimizer algorithm with a learning rate of 0.01; for TGCN with attention, we choose an Adam optimizer learning rate of 0.005. Then, we utilize Rectified Linear Unit (ReLu) as an activation function for both models. The batch size selected for both models is 16 and 128, respectively. From each batch, we take sequences of length n ($n = 8$ *hours*) from the training inputs X_{train} and targets Y_{train}. We use the optimizer to minimize the mean squared error (MSE) loss. While selecting optimal hyperparameters, we focus on maximizing performance metrics, including precision, recall, F1 score, accuracy, and specificity.

To train both TGCN and TGCN with attention models, we use the "NO ATTACK" samples (Dataset 1) and split them into training and validation sets with a 75:25 ratio. Then, we train both models for 10 times with 1000 epochs each time. A threshold (TH) is fine-tuned using the holdout set, Dataset 2, to maximize the ranking S score on the holdout set.

4.1.2. Unsupervised model design

We pre-process the data before inputting them to the AE and the HCAE. The pre-processing step includes normalization and removing null samples from the data. For normalization, we perform standard minimum maximum scaling ranges from 0 to 1. Both AE (baseline) and HCAE models are implemented using Scikit-learn API and are trained on a CPU with Intel core i5 10th gen. We use Adam Optimizer with learning rate = 0.0001, a decay factor of 0.5, and $(\beta_1, \beta_2) = (0.9, 0.99)$. Additionally, 500 epochs are selected with a minibatch size of 32. The design of HCAE differs from baseline AE in the design of the hidden layer definition. Early stopping is applied with patience = 3 for better regularization. Here, the patience parameter ensures convergence, when the training loss and validation loss don't change for three consecutive epochs and the training, is marked complete. We are compressing input features using an under-complete autoencoder architecture and both models' compression factor is selected as 2.5. Thus, we get the number of neurons in each layer as following: encoder layers: $[l_0, l_1, l_2] = [43, 34, 25]$; bottleneck layer: $[l_3] = [17]$; and decoder layer as: $[l_4, l_5, l_6] = [25, 34, 43]$.

Eqs. (15)–(18) represent AI assurance constraints, including Tide Weights, Orthogonal Weights, Uncorrelated Features, and Unit Norms constraints, which are applied to the AE. We pick a combination of these constraints and apply them to the hidden layers. Our goal is to obtain a meaningful and uncorrelated latent representation, a prerequisite for dimensionality reduction. We empirically select optimal hyperparameters for the AE and the HCAE models and maximize binary classification performance scores, including precision, recall, F1 score, accuracy, and specificity. Dataset 1 is used during model training, and Dataset 3 is used for model testing. Finally, we select threshold θ_{TH} by following an empirical approach. We plot the F1 scores for baseline AE and HCAE models for Dataset 3 against a threshold range from 96 % percentile to 100 % percentile of their average reconstruction errors (Fig. 7). We observe that both models reach a maximum F1 score at 98.5 % percentile. Hence, we choose $\theta_{TH} = 0.985$ as the model's threshold.

4.1.3. GAN model settings

A GAN [24] is used to generate poisoned data in our experiments. All 492 "ATTACK" samples from the test dataset (Dataset 3) are provided as input to the GAN network. For training the generator, the prior noise dimension is set as 32. The prior noise dimension and class labels are mapped into hidden layers (layer dimension 128) with a rectified linear

[7] github.com/scikit-learn/scikit-learn.

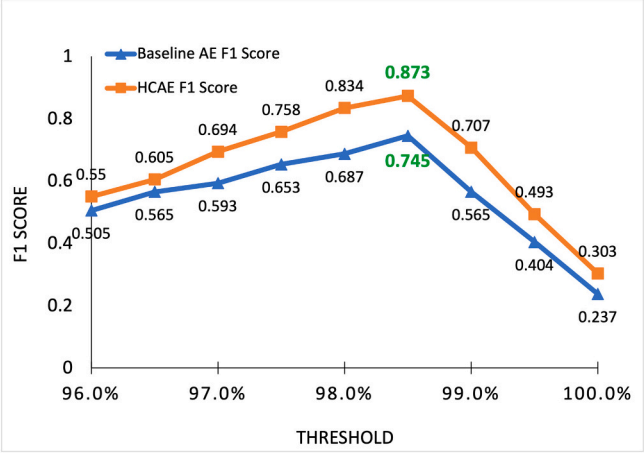


Fig. 7. F1 Score obtained on Dataset 3 for different Thresholds θ.

unit (ReLU) activation function. For output, a sigmoid function is selected as the last activation layer for the discriminator unit. We choose an Adam optimizer to minimize the loss function of the generator with a learning rate of 1e-5. Additionally, we pick a minibatch of size 128 and 200 epochs and set $(\beta_1, \beta_2) = (0.5, 0.9)$ for the optimizer. The poisoned dataset is generated by synthesizing only the "*ATTACK*" samples from the test set (Dataset 3). Then the generated samples are randomly placed in the same test set (Dataset 3) again. The synthetic dataset is conditioned to balance the number of samples in both classes. Testing a model with a balanced dataset helps us to evaluate the DL model's generalization ability. Hence, the poisoned dataset used in our experiments consists of 984 samples with 492 "ATTACK" samples and 492 "NO ATTACK" samples.

4.2. Model performance metrics

We use multiple performance metrics from the BATADAL competition to evaluate the model's ability to detect a threat in the shortest possible amount of time. In addition to this, we also use additional five metrics namely accuracy, precision, recall, specificity, and F1-score to measure the performance of a binary classifier.

4.2.1. Time-to-detection score: S_{TTD}

Time-to-detection is the difference between ground truth attack start time t_o (Eq. (21)) and algorithm detection start time t_d.

$$0 \leq \mathrm{S_{TTD}} = t_d - t_0 \leq \Delta t \tag{21}$$

The attack is indicated by Δt. A smaller TTD indicates that an algorithm has an improved detection performance during an ongoing attack. Additionally, the detection rate is associated with recall (%) or sensitivity which is otherwise referred as the True Positive Rate (TPR) and it is represented in Eq. (22). Additionally, precision, what proportion of positive identifications was actually correct is represented in Eq. (23).

$$\text{Sensitivity} = \text{Recall} = \text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{22}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{23}$$

Here, FN is the number of false negatives, and TP is the number of true positives. TPR is determined by the ratio of the correct attack classifications and the total number of attacks detected by the algorithm (including TP and FN). Additionally, we leverage True Negative Rate (TNR) or specificity metric to check false alarms by the models, and it is defined as (Eq. (24)),

$$\text{Specificity} = \text{TNR} = \frac{\text{TN}}{\text{FP} + \text{TN}} \tag{24}$$

TN is the number of True Negatives, and FP is the number of False Positives. TNR is determined by the ratio of the number of correct classifications for safe conditions (without attack) and the number of total classifications for safe conditions (including FP and TN).

4.2.2. Binary classification metric: F1-score

Eqs. (22) and (23) are also known as recall and precision respectively. In addition to accuracy and ranking, we calculate the F1-score using Eq. (25) that accounts for both precision and recall,

$$F1\ \ Score = 2^{*}\frac{\text{Precision}^{*}\text{Recall}}{\text{Precision} + \text{Recall}} \tag{25}$$

Training a DL model with an imbalanced dataset and evaluating its performance using the accuracy metric can be misleading [74]. In such cases, an F1-score is preferred over accuracy as the F1-score represents a harmonic mean of precision and recall.

4.2.3. Classification performance score: S_{CLF}

To compare with the other state-of-the-art detection algorithms, Eqs. (22) and (24) are merged as classification performance score S_{CLF} (Eq. (26)), the mean of Eqs. (22) and (24).

$$S_{\mathrm{CLF}} = \frac{\text{TPR} + \text{TNR}}{2} \tag{26}$$

This score (S_{CLF}) represents detection as well as false-negative alarms. Additionally, this score is relevant to the F1 score, which is appropriate for problems with binary classification. The score can result in a 0 or 1 (where 1 indicates a perfect classification).

4.2.4. Ranking score: S

Time-to-detection S_{TTD} and classification performance score S_{CLF} metrics can be merged further into a single ranking score as, Eq. (27):

$$S = \gamma \cdot S_{\mathrm{TTD}} + (1 - \gamma) \cdot S_{\mathrm{CLF}} \tag{27}$$

According to the BATADAL competition, γ is set to 0.5 to ensure the weight of the early detection and the accuracy are equally adjusted [8].

5. Deep H_2O results

This section presents the results of the three research questions presented prior (RQ1 - RQ3).

5.1. RQ1: AI assurance

5.1.1. Supervised detection results

Table 1 presents the attack detection performance of both supervised and unsupervised models. Among the two supervised models, results suggest that the TGCN with attention model performs better in attack detection in WDS. With the introduction of Attention and RMD assurance methods, the TGCN with attention model results in a significant

Table 1
Attack detection performance comparison between baseline and improved models on BATADAL Dataset 3.

Performance metrics	Supervised model			Unsupervised model		
(Dataset 3 2)	Tsiami	TGCN	TGCN with attention	Taormina	AE	HCAE
Precision	0.843	0.645	0.721	0.881	0.882	0.972
Recall	0.906	0.553	0.774	0.602	0.604	0.865
F1 Score	0.873	0.591	0.746	0.715	0.745	0.873
Accuracy	N/A	0.850	0.897	N/A	0.919	0.951
Specificity	N/A	0.922	0.927	N/A	0.972	0.983

improvement in recall and F1 score performance metrics. Out of the five metrics presented in Table 1, we observe an improvement in precision, recall, F1 score, and accuracy by 7.6 %, 22.1 %, 15.5 %, and 4.7 %, respectively. Precision, recall, and specificity metrics improve from baseline TGCN to TGCN with attention.

In terms of detecting attacks in WDS, results indicate that both the TGCN model (baseline) and TGCN with attention model successfully detect all seven attacks. Nevertheless, we notice (Fig. 8a, b, c, d) that the baseline model (TGCN) has a higher number of false positives compared to TGCN with attention model. We believe that the introduction of assurance methods (Attention and RMD) improves the TGCN with attention model and minimizes the number of false positives; This is also reflected in the model's performance with an improved F1 and precision score compared to its baseline. Overall, our results suggest that TGCN with attention model performs better than TGCN (baseline).

5.1.2. Unsupervised detection results

To study the impact of the assurance constraints (Eqs. (15)–(18)) on HCAE, we present the model performance with and without assurance constraints in Table 1. Not all four constraints bring optimal performance, but a combination of these constraints achieves better classification and dimensionality reduction performance than the baseline AE model. From Table 1, we observe that sensitivity, specificity, accuracy, and F1 score improve significantly with assurance constraints applied to the AE. Also, we observe that precision is increased, because HCAE learns the imbalanced data (BATADAL dataset is unbalanced) better than AE.

Fig. 9 presents the attack detection performance of the unsupervised models. The test dataset (Dataset 3) consists of seven different attacks. All seven attacks are classified as "ATTACK" by both AE and HCAE models. Fig. 9a and b present all seven attacks detected by the AE model and the HCAE model respectively. As Fig. 9a illustrates, in addition to detecting all SEVEN attacks, the AE model results in 21 sets of false alarms. On the contrary, the HCAE model results in a single false alarm (Fig. 9b). The result suggests that HCAE learns the complex interdependencies between the features during concealed attacks, hence performing better than AE in detecting the attacks.

Table 4 presents the performance metrics, including the ranking score (S). Although both HCAE and AE time-to-detection performance is identical (94.7 %), the classification performance (Ranking Score S) of the HCAE has significantly improved than the baseline AE. From the table, we observe that classification performance has been improved from 80 % to 92 %. Similarly, TPR also improved from 60.4 % to 86.5 %, a significant increase. This performance improvement is expected as HCAE learns the complex relationships between features in a deterministic scheme, whereas AE learns them in a non-deterministic approach.

5.2. RQ2: Data poisoning

In this sub-section, we present the performance of both supervised and unsupervised models on poisoned data. Table 2 presents the attack detection performance on synthetic poisoned data generated using GAN.

Results (Fig. 10a, b, c, d) suggest the supervised models perform poorly with poisoned data. More specifically for TGCN, we observe, by comparing Tables 1 and 2, that the attack detection performance of the model is decreased by more than 50 % across all five metrics. Similar to the baseline model, TGCN with attention model behaves poorly; results suggest, on average, a 65 % reduction in performance across all five metrics. The poor performance of supervised models on the poisoned data can be explained as follows. The TGCN and TGCN with attention models learn the behavior of a WDS by embedding the spatio-temporal structure of the WDS. In other words, they learn to detect attacks based on the sequential information inferred from the dataset during the training process. As the attacks are randomly distributed across the poisoned dataset (GAN data), both the supervised models fail to detect the attacks, resulting in poor performance.

On the contrary to supervised models, we observe that the AE and HCAE (unsupervised models) perform well (Fig. 11a, b, c, d) on the poisoned data. In some cases, results suggest the performance of both

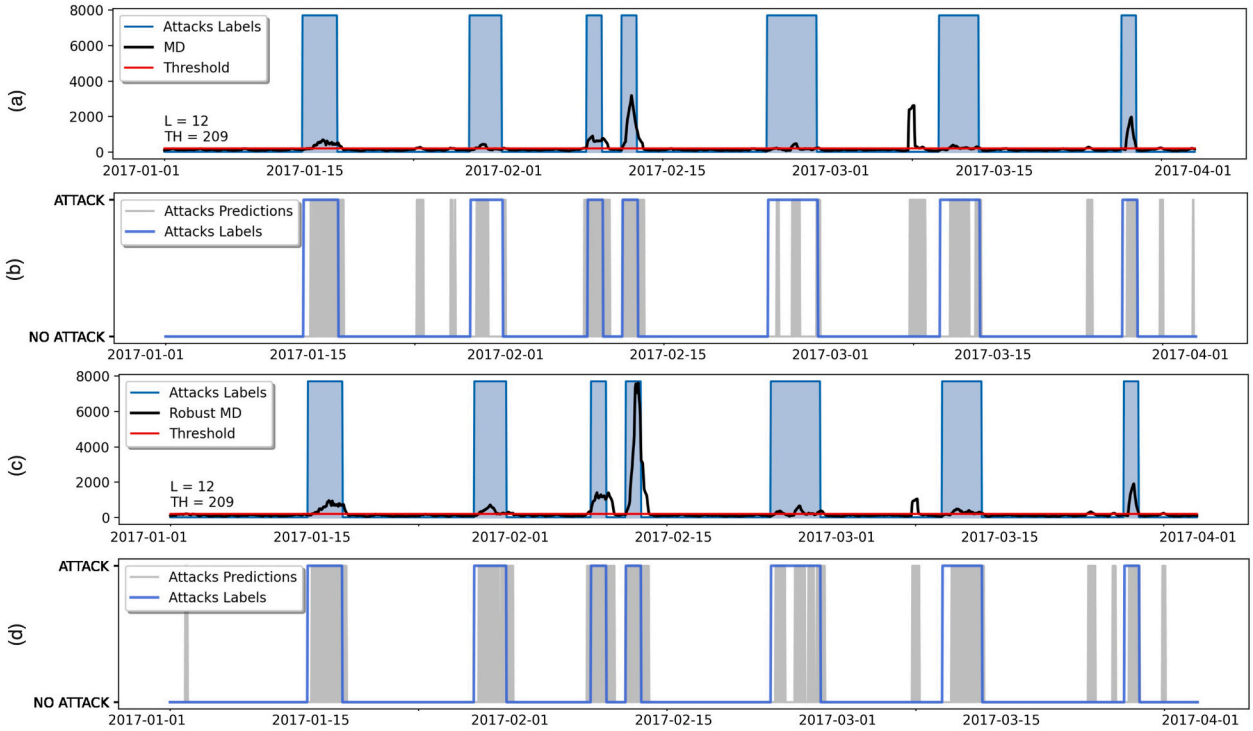


Fig. 8. (a) Apply threshold on TGCN (b) TGCN detection results on the test dataset (c) Apply threshold on TGCN with attention (d) TGCN with attention detection results on test dataset.

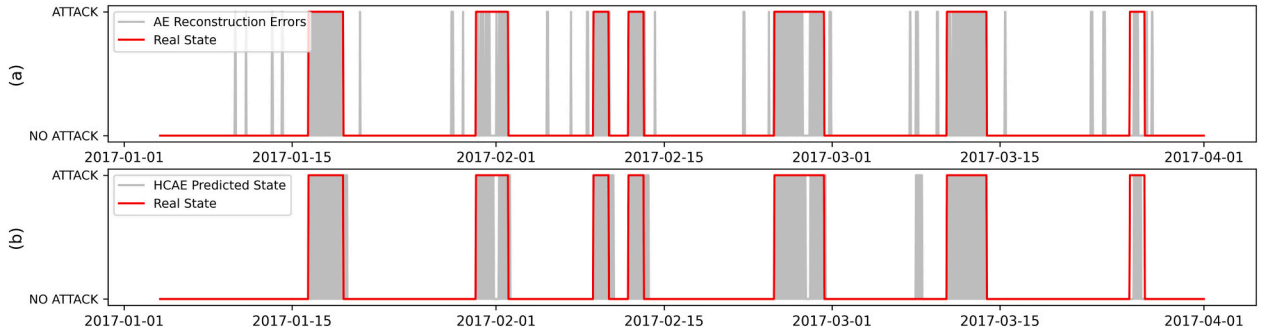


Fig. 9. (a) AE detection results on test dataset (b) HCAE detection results on test dataset.

Table 2
Attack detection performance comparison between baseline and improved models on GAN generated samples.

Performance metrics	Supervised model		Unsupervised model	
(GAN Samples)	TGCN	TGCN with attention	AE	HCAE
Precision	0.310	0.239	0.974	0.984
Recall	0.264	0.252	1	1
F1 Score	0.285	0.245	0.986	0.991
Accuracy	0.365	0.257	0.987	0.992
Specificity	0.458	0.262	0.976	0.985

unsupervised models is better on poisoned data (Table 2) than their performance on the test dataset (Table 1). This improved performance can be attributed to the fact that AE and HCAE models treat the training samples as non-sequential data, and the data are randomly placed with poisoned samples. Hence, they can detect the randomly distributed attacks from the poisoned dataset effectively.

5.3. RQ3: Feature localization

In this sub-section, we present the model's ability to identify the features impacted by an attack (feature localization). We perform feature localization by estimating the deviation of the features from the "NO ATTACK" dataset distribution.

The results presented so far suggest that the model customized with AI assurance methods and constraints performs better than their respective baseline model. Hence, for feature localization, we limit our evaluation to two models: TGCN with attention and HCAE. Table 3 presents the localization results for supervised and unsupervised models. The localized feature that matches the ground truth is highlighted in bold.

For TGCN with attention model, to localize the impacted features during an attack, we compare the mean squared error of the network from the testing set of its corresponding maximum error from the validation set (25 % of Dataset 1). The supervised model can successfully localize five attacked nodes among the seven attacks while failing to localize attacked nodes for Attack 9 and Attack 13.

Next, we present the feature localization performance of the unsupervised model. To localize the impacted features, we select the features

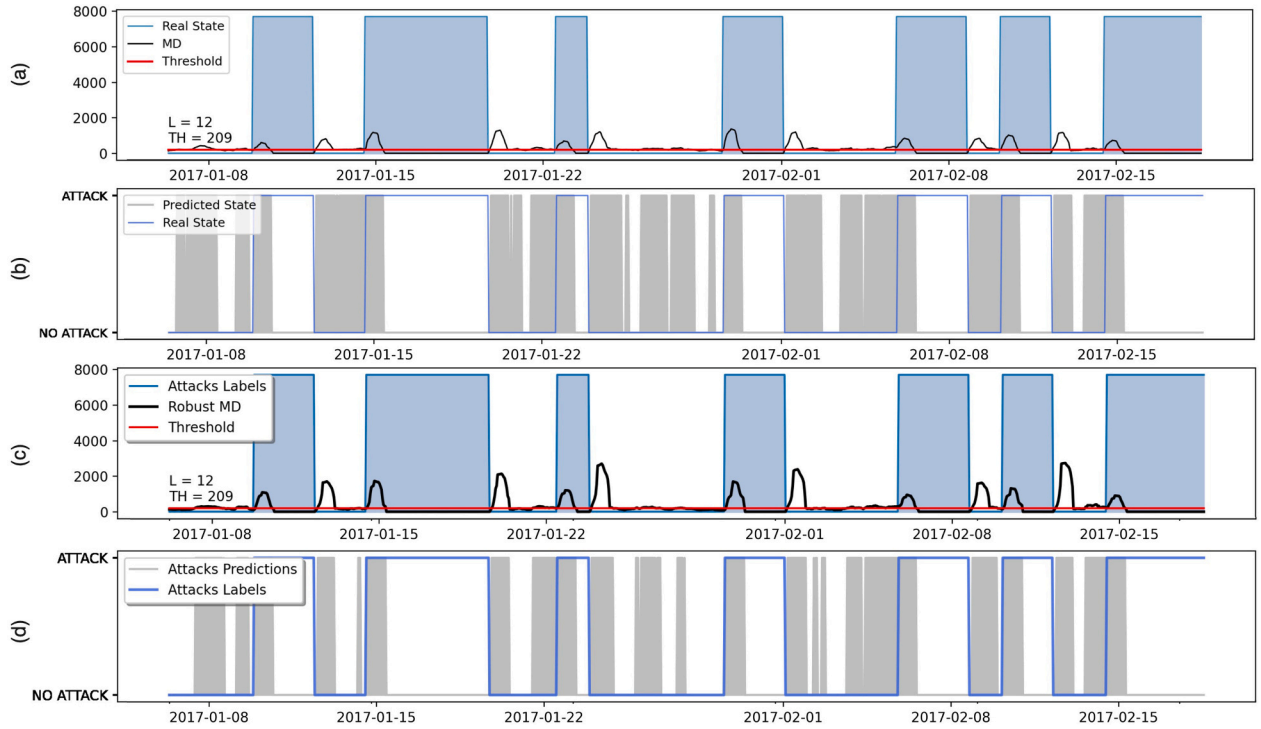


Fig. 10. (a) Apply threshold on TGCN; (b) TGCN detection results on the poisoned dataset (c) Apply threshold on TGCN with attention; (d) TGCN with attention detection results on poisoned dataset.

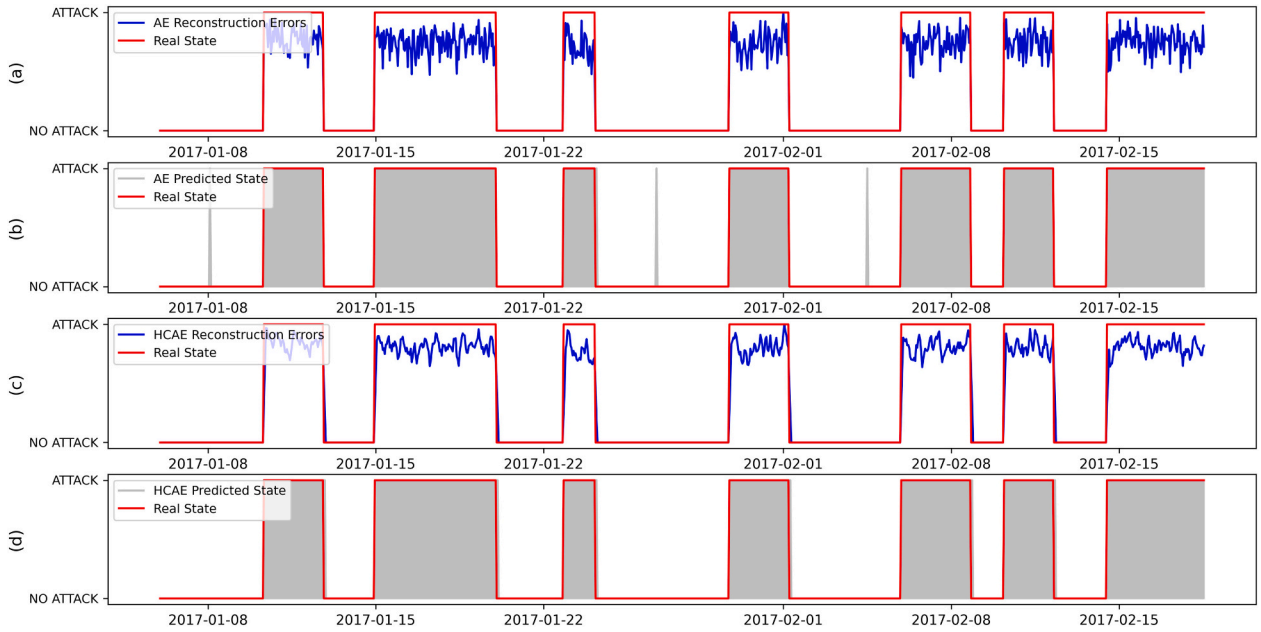


Fig. 11. (a) AE reconstruction errors on the poisoned dataset, (b) AE detection results on the poisoned dataset, (a) HCAE reconstruction errors on the poisoned dataset, (b) HCAE detection results on poisoned dataset.

Table 3
Feature localization results of TGCN with attention and HCAE on Dataset 3.

Attacks labels	Real attacks description	Predicted feature localization			
		TGCN with attention	FP	HCAE	FP
Attack 8	Alteration of L_T3 thresholds leading to underflow	P_J256, **L_T3**, P_J289, L_T2	3	P_J256, **L_T3,** L_T6, P_J280, **F_PU4**, F_PU7	4
Attack 9	Alteration of L_T2	P_J289, P_J422, P_J300, L_T7	4	P_J422, P_J289, P_J280, P_J300, L_T6, F_PU1, L_T5, **F_V2**, L_T7, L_T4	9
Attack 10	Activation of PU3	**F_PU3**, P_J280, L_T7, L_T4, P_J269, F_PU1, F_PU9	6	**F_PU3**, S_PU10, F_PU10, L_T1, P_J269, P_J307, P_J14, P_J317	7
Attack 11	Activation of PU3	**F_PU3**, P_J280, L_T7, F_PU1, L_T4, L_T6, P_J307, P_J415, F_PU6, P_J289	9	L_T1, **F_PU3**, F_PU10, P_J269, P_J14, F_PU1, F_PU2	6
Attack 12	Alteration of L_T2 readings leading to overflow	P_J289, P_J300, **L_T2**	2	P_J300, P_J289, L_T1, P_J280, F_PU7, L_T5, L_T6, **L_T2**, P_J422	8
Attack 13	Change the L_T7 thresholds	L_T6	1	P_J307, P_J302, F_PU8, **F_PU10**, **L_T7**, L_T6, P_J306	4
Attack 14	Alteration of L_T4 signal	**L_T4**, L_T7, P_J415, **L_T6**	2	P_J415, F_PU7, L_T1, **L_T6**, P_J307, F_PU10, P_J14	6

with the highest number of deviations from the threshold (θ_{th}) by estimating their mean squared error. During a predicted attack, we pick the top features for which reconstruction errors deviate most from the threshold (θ_{th}).

Overall, the results indicate that the modified models can successfully localize various attacks, including alteration of thresholds, signals, and meter readings. Furthermore, we observe that TGCN with attention localizes the attacked features with a minimal number of False Positives (FP) among the two models. False positives are estimated for each attack category by subtracting the set of total nodes detected by the model in that category from the set of ground truths. For example, Consider Attack 12, in which the readings of L_T2 are altered. While both the models successfully localize the feature (L_T2), the TGCN with attention model, in addition to identifying L_T2, also identifies two additional features as potentially attacked. In contrast, HCAE models identify additional eight nodes within the proximity as potentially attacked features. This is because, the neighbor nodes show similar behavior during normal operations, and therefore, during an attack, the model predicts those neighbor nodes are highly likely to be attacked.

5.4. Deep H_2O model sensitivity analysis

In this section, we evaluate the attack detection outcomes of both supervised and unsupervised models using Shapley values (SHapley values are the outcome of a game theoretic approach that explain the output of any ML model). In literature, variance-based sensitivity analysis is a popular approach that explains black box models; primarily, Sobol-based methods are gaining traction [75]. However, this approach has one major limitation: it cannot explain localized observations. In our work, we are more concerned with local observation explanations than global ones since the models detect attacks from different nodes and time points in a WDS. Therefore, we elected to use Deep Explainer,[8] an enhanced approach from SHapley Additive exPlanations (SHAP) library similar to Kernel SHAP. It approximates the conditional expectations of SHAP values using a selection of background samples.

In this analysis, we provide all seven categories of attack samples separately, generate SHAP values, and plot the scaled SHAP values ranging from 0 to 100 in Fig. 12. This figure represents local feature importance during the attack detection by the models. From the figure, we can observe that feature localization (Table 3) and Deep Explainer provide similar insights about the model's outcome. By observing Fig. 12, it becomes evident that the model gives less attention to

[8] github.com/slundberg/shap.

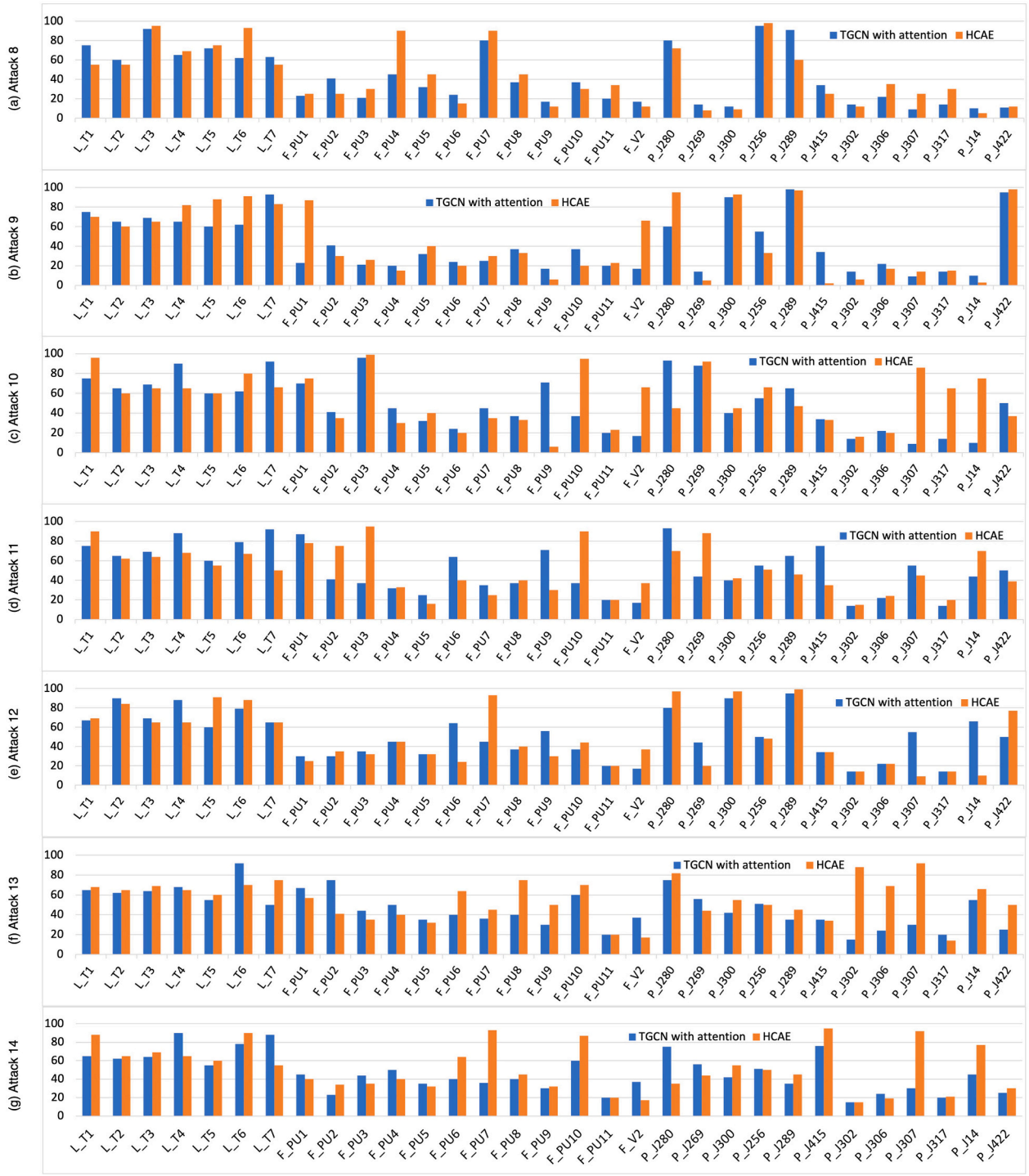


Fig. 12. Deep H_2O attack detection local explanations using Shapley values. Ground truths are as follows: (a) Attack 8: Alteration of L_T3 thresholds leading to underflow, (b) Attack 9: Alteration of L_T2, (c) Attack 10: Activation of PU3, (d) Attack 11: Activation of PU3, (e) Attack 12: Alteration of L_T2 readings leading to overflow, (f) Attack 13: Change the L_T7 thresholds, (g) Attack 14: Alteration of L_T4 signal.

deactivated nodes from the training set (Dataset 1), including PU3, PU5, PU6, PU9, and PU11 while giving much more importance to the flow of pumps that worked during the training set. Additionally, we observe that all tanks were given attention during all seven categories because they are always active in the training set. One shortcoming of both models is that they could not learn the relationships among junctions because of the imbalance of the training dataset; most influential junctions got prioritized by the model over less participating ones. Therefore, model attack localization was incorrect during attacks 9 and 13; L_T2 and L_T7 weren't detected because other junctions got more "attention", including P_J280, P_J289, P_J300, when compared to ground truth nodes V2 and PU10/PU11.

5.5. Comparison with BATADAL models

Next, we compare the attack classification performance of our models with the top-performing models from the BATADAL competition. To maintain consistency in our evaluation, all the models are

evaluated using the test dataset (Dataset 3). Table 4 presents the comparison results, where our models are highlighted in bold. The results indicate both the unsupervised and supervised model exhibits better performance. HCAE (unsupervised model), with a ranking score of 0.933, is ranked 3, whereas TGCN, with attention, achieves 0.845 and ranks eighth among the models from the BATADAL competition. Although our models do not achieve the highest ranking, they are superior compared to the top two models for the following reasons: 1). The top-ranked model is physics-based, and hence it is not relevant to compare with our model, an AI-based model. 2). The second-ranked model, although an AI-based model, might not perform well (detecting attacks) on previously unseen data. On the contrary, our models are scalable and demonstrate a better attack detection performance on unseen data.

Results indicate that adding the AI assurance methods to the TGCN model improves its overall performance. Compared to the baseline model (S =0.754), TGCN with attention model achieves a better score (S =0.845). Additionally, we observe that the time-to-detection (S_{TTD}) has improved significantly; the baseline model achieves 0.735, whereas the TGCN with attention model achieves 0.839. A higher S_{TTD} is significant in the context of WDS; an improved S_{TTD} score indicates that the TGCN with attention model can swiftly identify an attack at the earliest compared to its baseline model.

For unsupervised models, both HCAE and AE (baseline) achieve an identical score for time-to-detection (S_{TTD} = 0.947). However, we observe that the HCAE model has an improved TPR score (0.865) compared to its baseline (0.604). This results in the HCAE model achieving a ranking score (S = 0.933) substantially higher than its baseline (S = 0.873). Furthermore, a higher TPR indicates that the model detects most attack samples. Thus, improving the trustworthiness of the model during deployment.

Both TGCN with attention and HCAE models achieve a better ranking score compared to their respective baseline models.

6. Discussions

6.1. Water laws and public policy

Environmental and water laws govern our nation's water, air, waste, and other natural components. Most of the time, and due to the public's lack of awareness or attention, voters are usually drawn to water and environmental issues after wide-scale incidents of environmental damage, such as the Flint Water crisis[9] and its effects on safe drinking water in the state. The Clean Water Act (CWA) establishes the basic rules and benchmarks for regulating quality standards and discharging pollutants into the waters of the United States. The work presented in this manuscript aims to provide preventive measures for the health of water treatment plants against the rising dangers of cyber attacks. Deep H_2O is instrumental in governing cyber components of a water facility, providing recommendations to WDSs operators on when and where the attack occurs, and validating against water policies and Environmental Protection Agency (EPA) regulations. This project continues as a collaboration with WDSs in Northern Virginia and the District of Columbia (DC) to deploy Deep H_2O at local facilities and aim to expand it to other WDSs as well. Conclusions and future work items are presented next.

6.2. Conclusions and future work

This manuscript presents Deep H_2O, a novel cyber attack detection framework for WDSs. Deep H_2O applies AI assurance to two DL architectures, TGCN with attention and HCAE, and compares their performance improvement over their baseline models. For TGCN with attention model (supervised model), it has been observed that applying AI assurance, including attention and RMD with TGCN, improves the model's attack detection accuracy. Similarly, for HCAE (unsupervised model), applying AI assurance, including tide weights, orthogonality constraints, and other constraints, improves detection accuracy and F1-score of the HCAE model compared to AE.

The performance of both supervised and unsupervised models on poisoned data has been evaluated. For the supervised model, compared to its performance on the test dataset, it has been observed that most of the metrics decrease significantly. The supervised model struggles to perform (i.e., to detect an attack) if there is randomness in the dataset. Unlike the supervised model that performs poorly on poisoned data, our result indicates that the predictive performance of the unsupervised model (HCAE) is similar for the test data and the poisoned GAN data. No significant drop in the model's performance has been observed. To explain this phenomenon, the unsupervised model learns uncorrelated feature representation in the latent dimension and does not learn the sequential attributes. Hence the model can identify randomness in the poisoned data.

Result suggests that the HCAE model has better generalizability. Among the two models, the unsupervised model (HCAE) performs better in terms of ranking score and time-to-detection score. Also, HCAE is well generalized and regularized while detecting attacked samples on the BATADAL test set. This improved classification performance and recall values make HCAE a better choice for deployment in the WDS.

The study uses multiple performance metrics, including time-to-detection score, classification score, ranking score, precision, recall, accuracy, and F1 score, to measure the model's performance. The F1 score improvement is focused on the various metrics because of the heavily imbalanced BATADAL dataset. Therefore, this particular case, the F1 score becomes an important metric that considers model attack prediction errors and accounts for the type of errors by taking the harmonic mean of precision and recall. That is, only if both precision and recall values are high the F1 score gets higher; in this study, a higher F1 score indicates higher "*ATTACK*" and "*NO ATTACK*" harmonic class detection. Additionally, the unsupervised model outperforms the supervised model for WDS, including a better F1 score. The unsupervised model is a one-class classification method that generalizes well regardless of the water systems' spatio-temporal structure, making the model simpler than TGCN with attention. Additionally, the unsupervised model does not require labeling, an expensive and time-consuming activity in the model development process.

The ability of both supervised and unsupervised models in feature localization has been evaluated. Localizing a feature is tedious for both models during a concealed attack. Although the results are not highly accurate, they are promising and vital for WDS. For instance, both models can identify attacked node(s) or neighboring nodes during an "*ATTACK*". Further refining the model hyper-parameters by applying a grid search technique can improve the performance and result in better feature localization results, which is a potential future work. The sensitivity analysis of two models showed that less important or sensitive variables were inactive in the training set, while active components were the most influential during a cyber-attack. However, some common junctions had high sensitivity or importance flags due to imbalanced training data.

Additionally, the extension of this work can be the following: 1) The GAN used in these experiments to generate synthetic data fails to replicate the time-series information from the original dataset. The attack samples are generated merely using GAN. Consequently, the next plan is to use TimeGAN [76], a variant of GAN, to generate sequential (time-series) synthetic data consisting of both attack and non-attack samples and test the performance of DL models on the time-series synthetic data. 2) A large metropolitan city can have multiple WDSs across various locations within the metroplex. A bad actor can start a concealed attack on one of the WDS and continue to spread the attack across all

[9] https://www.michigan.gov/mdhhs/inside-mdhhs/legal/flint-water-settlement.

Table 4
Comparison of AI Assured models with BATADAL competition models.

Authors/models	No. Of attacks detected	Ranking score (S)	Time-to-detection (S_{TTD})	Classification score (S_{CLF})	TPR	TNR
Housh and Ohar	7	0.97	0.965	0.975	0.953	0.997
Abokifa et al.	7	0.949	0.958	0.944	0.921	0.959
HCAE	7	0.933	0.947	0.919	0.865	0.983
Tsiami et al.	7	0.931	0.934	0.928	0.885	0.971
Giacomoni et al.	7	0.927	0.936	0.917	0.838	0.997
Brentan et al.	6	0.894	0.857	0.931	0.889	0.973
AE	7	0.873	0.947	0.800	0.604	0.972
TGCN with attention	7	0.845	0.839	0.851	0.774	0.927
Chandy et al.	7	0.802	0.835	0.768	0.857	0.678
Pasha et al.	7	0.773	0.885	0.66	0.329	0.992
TGCN	7	0.754	0.735	0.773	0.553	0.922
Aghashahi et al.	3	0.534	0.429	0.64	0.396	0.884

locations. To swiftly detect and prevent such attacks, Federated Learning (FL) techniques [77] can be adapted to learn from the initial concealed attack and leverage that information to prevent future attacks (of similar nature) across other WDSs. Furthermore, using the real-time data collected from the WDSs to retrain the DL model can significantly improve the detection performance of the model. However, given the geographically distributed nature of WDSs, it is essential to preserve the privacy of the real-time data (collected from the WDSs). Therefore, the plan is to use FL techniques to guarantee data and model privacy. 3) Training and deploying a DL model across different WDSs is challenging as the threshold might vary across different WDSs locations. This is further complicated by a set of different operations across WDSs. Another interesting idea is to explore Context learning [72] to enable DL models to be context-aware (such as population and weather) and efficiently detect attacks that vary based on different thresholds. Furthermore, training and evaluation of the Deep H_20 framework using real-world WDS datasets[10] such as: Water Distribution (WADI) dataset and Secure Water Treatment(SWaT) is a future task. Lastly, a plan to develop approaches that explain the model's outcomes to water plant operators could be a great study, which would result in higher adoption rates and increased trustworthiness [25] of such frameworks at water facilities in the United States.

Declaration of competing interest

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

Data availability

Data will be made available on request.

Acknowledgement

We thank Ashita Anuga and Jaganmohan Chandrasekaran for their edits and inputs to the text of the manuscript. Additionally, we express our appreciation to Nitish Gorentala for his discussions and inputs on GANs.

Appendix A. First seven attacks set descriptions (C-Town Dataset 2)

Table A.5
Dataset 2 attacks description [8].

Identifier	Starting time (D/M/Y H)	Ending time (D/M/Y H)	Duration (H)	Attack descriptions	SCADA concealment	Label (h)
1	13/09/2016 23	16/09/2016 00	50	Attacker alters SCADA transmission to PLC9 and changes L_T7 thresholds determining when pumps PU10/PU11 are switched on/off. Low levels in T7.	Replay attack on L_T7	42
2	26/09/2016 11	27/09/2016 10	24	Like Attack #1.	Like Attack #1 but replay attack extended on PU10/PU11 flow and status.	0
3	09/10/2016 09	11/10/2016 20	60	Attack alters L_T1 readings sent by PLC2 to PLC1, which reads a constant low level and keeps pumps PU1/PU2 on. Overflow in T1.	Polyline to offset L_T1 increase.	60
4	29/10/2016 19	02/11/2016 16	94	Like Attack #3.	Replay attack on L_T1, PU1/PU2 flow and status, as well as on pressure at pumps outlet (P_J269).	37
5	26/11/2016 17	29/11/2016 04	60	Working speed of PU7 reduced to 0.9 of nominal speed. Lower water levels in T4.		7
6	06/12/2016 07	10/12/2016 04	94	Like Attack #5, but speed reduced to 0.7.	Replay attack on L_T4.	73
7	14/12/2016 15	19/12/2016 04	110	Like Attack #6.	Replay attack on L_T4, as well as on PU6/PU7 flow and status.	0

[10] https://itrust.sutd.edu.sg/

Appendix B. Remaining seven attacks set descriptions (C-Town Dataset 3)

Table B.6
Dataset 3 attacks description [8].

Identifier	Starting time (D/M/Y H)	Ending time (D/M/Y H)	Duration (H)	Attack descriptions	SCADA concealment
8	16/01/2017 09	19/01/ 2017 06	70	Attacker gains control of PLC3 and changes L_T3 thresholds determining when pumps PU4/PU5 are switched on/off. Low levels in T3.	Replay attack on L_T3, as well as on PU4/PU5 flow and status.
9	30/01/2017 08	02/02/ 2017 00	65	Attack alters L_T2 readings arriving to PLC3, which reads a low level and keeps valve V2 OPEN. Attack leads T2 to overflow	Polyline to offset L_T2 increase
10	09/02/2017 03	10/02/ 2017 09	31	Malicious activation of pump PU3	
11	12/02/2017 01	13/02/ 2017 07	31	Similar to Attack #10	
12	24/02/2017 05	28/02/ 2017 08	100	Similar to Attack #9	Replay attack on L_T2, V2 flow and status, as well as on V2 inlet and outlet pressure readings (P_J14, P_J422)
13	10/03/2017 14	13/03/ 2017 21	80	Attacker gains control of PLC5 and changes the L_T7 thresholds determining when pumps PU10/PU11 are switched on/off. The pumps are forced to switch on/off continuously during attack	Replay attack on L_T7, PU10/ PU11 flow, and status, as well as on pumps inlet and outlet pressure readings (P_J14, P_J422). Inlet pressure concealment terminates before that of other variables.
14	25/03/2017 20	27/03/ 2017 01	30	Alteration of T4 signal arriving to PLC6. Overflow in T6.	

Appendix C. Hyper-parameters selection For Deep H_2O

Table C.7
Hyperparameters selection using random search (Bold values are the finally selected hyperparameters).

Hyperparameters	Baseline AE	HCAE	Baseline TGCN	TGCN with attention
Adam optimizer learning rate	**0.0001**, 0.001, 0.01, 0.1	**0.0001**, 0.001, 0.01, 0.1	0.0001, 0.001, 0.005, **0.01**, 0.1	0.0001, 0.001, **0.005**, 0.01, 0.1
Batch size	8, 16, **32**, 64, 128, 264	8, 16, **32**, 64, 128, 264	8, **16**, 32, 64, 128, 264	8, 16, 32, 64, **128**, 264
Sequence length	N/A	N/A	4, **8**, 16, 24, 32 h	4, **8**, 16, 24, 32 h
Number of epochs	**500**, 1000, 2500, 5000	**500**, 1000, 2500, 5000	500, **1000**, 2500, 5000	500, **1000**, 2500, 5000
Number of hidden layers	3,**5**,7,9,11	3,**5**,7,9,11	N/A	N/A
Hidden dimensions	N/A	N/A	8, 16, 32, **64**, 100, 128	8, **16**, 32, 64, 100, 128

References

[1] Z. Wang, H. Song, D.W. Watkins, K.G. Ong, P. Xue, Q. Yang, X. Shi, Cyber-physical systems for water sustainability: challenges and opportunities, IEEE Commun. Mag. 53 (5) (2015) 216–222.

[2] A. Hassanzadeh, A. Rasekh, S. Galelli, M. Aghashahi, R. Taormina, A. Ostfeld, K. Banks, A Review of Cybersecurity Incidents in the Water Sector, arXiv, 2020, https://doi.org/10.1061/%28ASCE%29EE.1943-7870.0001686 preprint arXiv: 2001.11144.

[3] N. Tuptuk, P. Hazell, J. Watson, S. Hailes, A systematic review of the state of cyber-security in water systems, Water 13 (1) (2021) 81.

[4] F. Robles, N. Perlroth, Dangerous stuff': Hackers tried to poison water supply of florida town, Retrieved from, The New York Times, 2021, https://www.nytimes.com/2021/02/08/us/oldsmar-florida-water-supply-hack.html.

[5] A. Greenberg, A hacker tried to poison a florida city's water supply, officials say, Retrieved 23rd of May from, in: Wired Magazine, 2021, https://www.wired.com/story/oldsmar-florida-water-utility-hack.

[6] C. Perrow, Normal Accidents: Living With High Risk Technologies-updated Edition, 1999.

[7] M. Board, N.R. Council, et al., Macondo Well Deepwater Horizon Blowout: Lessons for Improving Offshore Drilling Safety, National Academies Press, 2012.

[8] R. Taormina, S. Galelli, N.O. Tippenhauer, E. Salomons, A. Ostfeld, D.G. Eliades, M. Aghashahi, R. Sundararajan, M. Pourahmadi, M.K. Banks, et al., Battle of the attack detection algorithms: disclosing cyber attacks on water distribution networks, J. Water Resour. Plan. Manag. 144 (8) (2018), 04018048.

[9] Y. Bengio, A. Courville, P. Vincent, Representation learning: a review and new perspectives, IEEE Trans. Pattern Anal. Mach. Intell. 35 (8) (2013) 1798–1828.

[10] M.N.K. Sikder, F.A. Batarseh, Outlier Detection Using AI: A Survey, arXiv, 2021 preprint arXiv:2112.00588.

[11] L. Garcia, F. Brasser, M.H. Cintuglu, A.-R. Sadeghi, O.A. Mohammed, S.A. Zonouz, Hey, my malware knows physics! attacking plcs with physical model aware rootkit, , NDSS, 2017.

[12] Y. Mo, B. Sinopoli, Secure control against replay attacks, in: 2009 47th Annual Allerton Conference on Communication, Control, and Computing (Allerton), IEEE, 2009, pp. 911–918.

[13] A. Teixeira, I. Shames, H. Sandberg, K.H. Johansson, Revealing stealthy attacks in control systems, in: 2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton), IEEE, 2012, pp. 1806–1813.

[14] R. Taormina, S. Galelli, Deep-learning approach to the detection and localization of cyber-physical attacks on water distribution systems, J. Water Resour. Plan. Manag. 144 (10) (2018), 04018065.

[15] S. Adepu, A. Mathur, Distributed attack detection in a water treatment plant: method and case study, IEEE Trans. Dependable Secure Comput. 18 (1) (2018) 86–99.

[16] F. A. Batarseh M. O. Yardimci R. Suzuki M. N. K. Sikder Z. Wang W.-Y. Mao n.d. Realtime Management of Wastewater Treatment Plants Using AI.

[17] S. Amin, A.A. Cárdenas, S.S. Sastry, Safe and secure networked control systems under denial-of-service attacks, in: Hybrid Systems: Computation and Control: 12th International Conference, HSCC 2009, San Francisco, CA, USA, April 13-15, 2009, Proceedings 12, Springer, 2009, pp. 31–45.

[18] J. Bruna, W. Zaremba, A. Szlam, Y. LeCun, Spectral Networks and Locally Connected Networks on Graphs, arXiv, 2013 preprint arXiv:1312.6203.

[19] J. Bai, J. Zhu, Y. Song, L. Zhao, Z. Hou, R. Du, H. Li, A3t-gcn: attention temporal graph convolutional network for traffic forecasting, ISPRS Int. J. Geo Inf. 10 (7) (2021) 485.

[20] K. Cho, B. van Merriënboer, D. Bahdanau, Y. Bengio, On the properties of neural machine translation: encoder–decoder approaches, in: Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, Association for Computational Linguistics, Doha, Qatar, 2014, pp. 103–111, https://doi.org/10.3115/v1/W14-4012. https://aclanthology.org/W14-4012.

[21] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, Y. Bengio, Show, attend and tell: neural image caption generation with visual attention, in: International Conference on Machine Learning, PMLR, 2015, pp. 2048–2057.

[22] J. Xiao, H. Ye, X. He, H. Zhang, F. Wu, T.-S. Chua, Attentional factorization machines: learning the weight of feature interactions via attention networks, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, 2017, pp. 3119–3125.

[23] G.J. McLachlan, Mahalanobis distance, Resonance 4 (6) (1999) 20–26.

[24] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, Adv. Neural Inf. Proces. Syst. 27 (2014).

[25] F.A. Batarseh, L. Freeman, C.-H. Huang, A survey on artificial intelligence assurance, J. Big Data 8 (1) (2021) 1–30.

[26] M.N.K. Sikder, F.A. Batarseh, P. Wang, N. Gorentala, Model-agnostic scoring methods for artificial intelligence assurance, in: 2022 IEEE 29th Annual Software Technology Conference (STC), 2022, pp. 9–18, https://doi.org/10.1109/STC55697.2022.00011.

[27] P. Radanliev, D. De Roure, M. Van Kleek, O. Santos, U. Ani, Artificial intelligence in cyber physical systems, AI & Soc. 36 (3) (2021) 783–796.

[28] S. Gurrapu, F.A. Batarseh, P. Wang, M.N. Kabir Sikder, N. Gorentala, M. Gopinath, DeepAg: Deep Learning Approach for Measuring the Effects of Outlier Events on Agricultural Production and Policy, in: 2021 IEEE Symposium Series on Computational Intelligence (SSCI), Orlando, FL, USA, 2021, pp. 1–8, https://doi.org/10.1109/SSCI50451.2021.9659921.

[29] S. Gurrapu, N. Sikder, P. Wang, N. Gorentala, M. Williams, F.A. Batarseh, Applications of Machine Learning For Precision Agriculture and Smart Farming, in: The International FLAIRS Conference Proceedings 34, 2021, https://doi.org/10.32473/flairs.v34i1.128497.

[30] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, H. Li, T-gcn: a temporal graph convolutional network for traffic prediction, IEEE Trans. Intell. Transp. Syst. 21 (9) (2019) 3848–3858.

[31] J. Bai, J. Zhu, Y. Song, L. Zhao, Z. Hou, R. Du, H. Li, A3t-gcn: attention temporal graph convolutional network for traffic forecasting, ISPRS Int. J. Geo Inf. 10 (7) (2021) 485.

[32] I.C. Covert, B. Krishnan, I. Najm, J. Zhan, M. Shore, J. Hixson, M.J. Po, Temporal graph convolutional networks for automatic seizure detection, in: Machine Learning for Healthcare Conference, PMLR, 2019, pp. 160–180.

[33] L. Tsiami, C. Makropoulos, Cyber—physical attack detection in water distribution systems with temporal graph convolutional neural networks, Water 13 (9) (2021) 1247.

[34] M.S. Mahmud, J.Z. Huang, X. Fu, Variational autoencoder-based dimensionality reduction for high-dimensional small-sample data classification, Int. J. Comput. Intell. Appl. 19 (01) (2020), 2050002.

[35] C. Zhou, R.C. Paffenroth, Anomaly detection with robust deep autoencoders, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 665–674.

[36] C.C. Aggarwal, An introduction to outlier analysis, in: Outlier Analysis, Springer, 2017, pp. 1–34.

[37] J. Sun, X. Wang, N. Xiong, J. Shao, Learning sparse representation with variational auto-encoder for anomaly detection, IEEE Access 6 (2018) 33353–33361.

[38] B. Zong, Q. Song, M.R. Min, W. Cheng, C. Lumezanu, D. Cho, H. Chen, Deep autoencoding gaussian mixture model for unsupervised anomaly detection, in: International Conference on Learning Representations, 2018. https://openreview.net/forum?id=BJJLHbb0-.

[39] K. Wang, C. Gou, Y. Duan, Y. Lin, X. Zheng, F.-Y. Wang, Generative adversarial networks: introduction and outlook, IEEE/CAA J. Autom. Sin. 4 (4) (2017) 588–598.

[40] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial networks, Commun. ACM 63 (11) (2020) 139–144.

[41] W. Zhou, X.-M. Kong, K.-L. Li, X.-M. Li, L.-L. Ren, Y. Yan, Y. Sha, X.-Y. Cao, X.-J. Liu, Attack sample generation algorithm based on data association group by Gan in industrial control dataset, Comput. Commun. 173 (2021) 206–213.

[42] M.H. Shahriar, N.I. Haque, M.A. Rahman, M. Alonso, G-ids: generative adversarial networks assisted intrusion detection system, in: 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC), IEEE, 2020, pp. 376–385.

[43] N. Kadosh, A. Frid, M. Housh, Detecting cyber-physical attacks in water distribution systems: one-class classifier approach, J. Water Resour. Plan. Manag. 146 (8) (2020), 04020060.

[44] K.W. Min, Y.H. Choi, A.K. Al-Shamiri, J.H. Kim, Application of artificial neural network for cyber-attack detection in water distribution systems as cyber physical systems, in: International Conference on Harmony Search Algorithm, Springer, 2019, pp. 82–88.

[45] X.-Y. Zou, Y.-L. Lin, B. Xu, Z.-B. Guo, S.-J. Xia, T.-Y. Zhang, A.-Q. Wang, N.-Y. Gao, A novel event detection model for water distribution systems based on data-driven estimation and support vector machine classification, Water Resour. Manag. 33 (13) (2019) 4569–4581.

[46] F. Bagherzadeh, M.-J. Mehrani, M. Basirifard, J. Roostaei, Comparative study on total nitrogen prediction in wastewater treatment plant and effect of various feature selection methods on machine learning algorithms performance, J. Water Process Eng. 41 (2021), 102033.

[47] F. Bagherzadeh, A.S. Nouri, M.-J. Mehrani, S. Thennadil, Prediction of energy consumption and evaluation of affecting factors in a full-scale wwtp using a machine learning approach, Process Saf. Environ. Prot. 154 (2021) 458–466.

[48] M.-J. Mehrani, F. Bagherzadeh, M. Zheng, P. Kowal, D. Sobotka, J. Makinia, Application of a hybrid mechanistic/machine learning model for prediction of nitrous oxide (n2o) production in a nitrifying sequencing batch reactor, Process Saf. Environ. Prot. 162 (2022) 1015–1024.

[49] W.-Y. Mao, M. Yardimci, M. Nguyen, D. Sobien, L. Freeman, F.A. Batarseh, A. Rahman, V. Fordham, Trustworthy ai solutions for cyberbiosecurity challenges in water supply systems, in: The International FLAIRS Conference Proceedings 35, 2022.

[50] C.H. Yoong, J. Heng, Framework for continuous system security protection in swat, in: Proceedings of the 2019 3rd International Symposium on Computer Science and Intelligent Control, 2019, pp. 1–6.

[51] S. Adepu, A. Mathur, Distributed detection of single-stage multipoint cyber attacks in a water treatment plant, in: Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security, 2016, pp. 449–460.

[52] M. Macas, C. Wu, An unsupervised framework for anomaly detection in a water treatment system, in: 2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA), IEEE, 2019, pp. 1298–1305.

[53] G. Zizzo, C. Hankin, S. Maffeis, K. Jones, Adversarial attacks on time-series intrusion detection for industrial control systems, in: 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), IEEE, 2020, pp. 899–910.

[54] E. Anthi, L. Williams, M. Rhode, P. Burnap, A. Wedgbury, Adversarial attacks on machine learning cybersecurity defences in industrial control systems, J. Inf. Secur. Appl. 58 (2021), 102717.

[55] S. Zhang, D. Zhou, M.Y. Yildirim, S. Alcorn, J. He, H. Davulcu, H. Tong, Hidden: hierarchical dense subgraph detection with application to financial fraud detection, in: Proceedings of the 2017 SIAM International Conference on Data Mining, SIAM, 2017, pp. 570–578.

[56] L. Backstrom, J. Leskovec, Supervised random walks: predicting and recommending links in social networks, in: Proceedings of the Fourth ACM International Conference on Web Search and Data Mining, 2011, pp. 635–644.

[57] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.

[58] Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, IEEE Trans. Neural Netw. 5 (2) (1994) 157–166.

[59] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling, in: NIPS 2014 Workshop on Deep Learning, December 2014, 2014.

[60] M.N.K. Sikder, F.A. Batarseh, Outlier Detection Using Ai: A Survey, in: AI Assurance, Elsevier, in, 2022, pp. 231–292.

[61] P.J. Rousseeuw, Least median of squares regression, J. Am. Stat. Assoc. 79 (388) (1984) 871–880.

[62] J. Zhai, S. Zhang, J. Chen, Q. He, Autoencoder and its various variants, in: 2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2018, pp. 415–419, https://doi.org/10.1109/SMC.2018.00080.

[63] Z. Chen, C.K. Yeo, B.S. Lee, C.T. Lau, Autoencoder-based network anomaly detection, in: 2018 Wireless Telecommunications Symposium (WTS), IEEE, 2018, pp. 1–5.

[64] L. Bottou, Large-scale machine learning with stochastic gradient descent, Retrieved 23rd of May from, in: Proceedings of COMPSTAT'2010, Springer, 2010, pp. 177–186.

[65] S. Ruder, An overview of gradient descent optimization algorithms, arXiv, 2016 preprint arXiv:1609.04747.

[66] G. Pang, C. Shen, L. Cao, A.V.D. Hengel, Deep learning for anomaly detection: a review, ACM Computing Surveys (CSUR) 54 (2) (2021) 1–38.

[67] C.C. Tan, C. Eswaran, Performance comparison of three types of autoencoder neural networks, in: 2008 Second Asia International Conference on Modelling & Simulation (AMS), IEEE, 2008, pp. 213–218.

[68] D. Alain, S. Olivier, Gated autoencoders with tied input weights, in: International Conference on Machine Learning, PMLR, 2013, pp. 154–162.

[69] L. Huang, X. Liu, B. Lang, A.W. Yu, Y. Wang, B. Li, Orthogonal weight normalization: Solution to optimization over multiple dependent stiefel manifolds in deep neural networks, in: Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[70] M. Kim, H.-C. Choi, Uncorrelated feature encoding for faster image style transfer, Neural Netw. 140 (2021) 148–157.

[71] S.C. Douglas, S.-I. Amari, S.-Y. Kung, On gradient adaptation with unit-norm constraints, IEEE Trans. Signal Process. 48 (6) (2000) 1843–1847.

[72] A. Erba, R. Taormina, S. Galelli, M. Pogliani, M. Carminati, S. Zanero, N. O. Tippenhauer, Constrained concealment attacks against reconstruction-based anomaly detectors in industrial control systems, in: Annual Computer Security Applications Conference, 2020, pp. 480–495.

[73] L. Muñoz-González, B. Pfitzner, M. Russo, J. Carnerero-Cano, E.C. Lupu, Poisoning attacks with generative adversarial nets, arXiv, 2019 preprint arXiv:1906.07773.

[74] P. Branco, L. Torgo, R.P. Ribeiro, A survey of predictive modeling on imbalanced domains, ACM Computing Surveys (CSUR) 49 (2) (2016) 1–50.

[75] F. Bagherzadeh, T. Shafighfard, Ensemble machine learning approach for evaluating the material characterization of carbon nanotube-reinforced cementitious composites, 2022.

[76] J. Yoon, D. Jarrett, M. Van der Schaar, Time-series generative adversarial networks, Adv. Neural Inf. Proces. Syst. 32 (2019).

[77] Q. Yang, Y. Liu, T. Chen, Y. Tong, Federated machine learning: concept and applications, ACM Trans. Intell. Syst. Technol. 10 (2) (2019) 1–19.