

Defending Water Treatment Networks: Exploiting Spatio-temporal Effects for Cyber Attack Detection

1st Dongjie Wang
Department of Computer Science
University of Central Florida
Orlando, United States
wangdongjie@knights.ucf.edu

2nd Pengyang Wang
Department of Computer Science
University of Central Florida
Orlando, United States
pengyang.wang@knights.ucf.edu

3rd Jingbo Zhou
Baidu Research
Baidu Inc.
Beijing, China
zhoujingbo@baidu.com

4th Leilei Sun
Department of Computer Science
Beihang University
Beijing, China
leileisun@buaa.edu.cn

5th Bowen Du
Department of Computer Science
Beihang University
Beijing, China
dubowen@gmail.com

6th Yanjie Fu[†]
Department of Computer Science
University of Central Florida
Orlando, United States
yanjie.fu@ucf.edu

Abstract—While Water Treatment Networks (WTNs) are critical infrastructures for local communities and public health, WTNs are vulnerable to cyber attacks. Effective detection of attacks can defend WTNs against discharging contaminated water, denying access, destroying equipment, and causing public fear. While there are extensive studies in WTNs attack detection, they only exploit the data characteristics partially to detect cyber attacks. After preliminary exploring the sensing data of WTNs, we find that integrating spatio-temporal knowledge, representation learning, and detection algorithms can improve attack detection accuracy. To this end, we propose a structured anomaly detection framework to defend WTNs by modeling the spatio-temporal characteristics of cyber attacks in WTNs. In particular, we propose a spatio-temporal representation framework specially tailored to cyber attacks after separating the sensing data of WTNs into a sequence of time segments. This framework has two key components. The first component is a temporal embedding module to preserve temporal patterns within a time segment by projecting the time segment of a sensor into a temporal embedding vector. We then construct Spatio-Temporal Graphs (STGs), where a node is a sensor and an attribute is the temporal embedding vector of the sensor, to describe the state of the WTNs. The second component is a spatial embedding module, which learns the final fused embedding of the WTNs from STGs. In addition, we devise an improved one class-SVM model that utilizes a new designed pairwise kernel to detect cyber attacks. The devised pairwise kernel augments the distance between normal and attack patterns in the fused embedding space. Finally, we conducted extensive experimental evaluations with real-world data to demonstrate the effectiveness of our framework: it achieves an accuracy of 91.65%, with average improvement ratios of 82.78% and 22.96% with respect to F1 and AUC, compared with baseline methods.

I. INTRODUCTION

Water Treatment Networks (WTNs) are critical infrastructures that utilize industrial control systems, sensors and communication technologies to control the water purification processes to improve the water quality and distribution for drinking, irrigation, or industrial uses. Although it is a critical infrastructure, WTNs are vulnerable to cyber attacks. For

example, the water sector reported the fourth largest number of incidents in 2016 ¹. How does a cyber attack to WTNs look like? Figure 1 shows that a water treatment procedure includes six stages (*i.e.*, P1-P6), each of which is monitored by sensors; a cyber attack compromises the RO Feed Pump sensor of P4 to change the levels of chemicals being used to treat tap water. As a result, there is a compelling need for an effective solution to attack detection in WTNs.

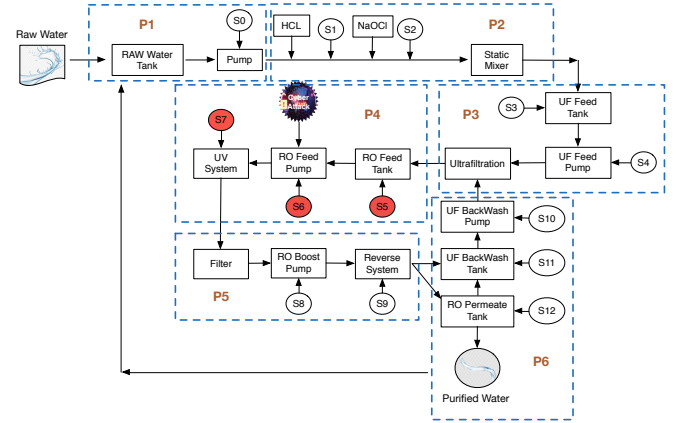


Fig. 1. Cyber attack example: one cyber attack happens at RO Feed Pump of P4, then the cyber attack effect spreads to other devices in P4.

In the literature, there are a number of studies about cyber attack detection in WTNs [1]–[4]. However, most of these studies only exploit traditional spatiotemporal data preprocessing and pattern extraction methods to distinguish attack patterns. Our preliminary explorations find that tremendous opportunities exist in solving the problem by teaching a machine to augment the differences between normal and attack patterns. To this end, in this paper, we aim to effectively solve

¹<https://www.osti.gov/servlets/purl/1372266>

the attack detection problem by augmenting the difference between normal and attack patterns in WTNs.

However, it is challenging to mine the spatio-temporal graph stream data of WTNs and identify the strategy to maximize the pattern divergence between normal and attack behaviors. By carefully examining the sensing data of WTNs, we identify three types of characteristics of cyber attacks: (1) *delayed effect*, meaning that many attacks will not take effects immediately, but usually exhibit symptoms after a while; (2) *continued effect*, meaning that the effects of attacks will sustain for a while, not disappear rapidly; (3) *cascading effect*, meaning that the effects of attacks propagate to other sensors across the whole WTNs. Specifically, the *delayed* and *continued* effects are both temporal, and the *cascading effect* is spatial. More importantly, these three effects are mutually coupled, impacted, and boosted in WTNs. A new framework is required to address the margin maximization between normal and attack pattern learning, under the three coupled effects.

Along this line, we propose a structured detection framework. This framework has two main phases: (1) spatio-temporal representation learning, which includes two modules: incorporating temporal effects and spatial effects; (2) improved unsupervised one-class detection, which utilizes a new designed pairwise kernel to make detection more accurate. Next, we briefly introduce our structured spatio-temporal detection framework named STD0.

Phase 1: Spatio-temporal representation learning. This phase aims to learn a good spatio-temporal representation over the sensing data of WTNs with two modules. The first module of this part is to integrate temporal effects. Cyber attacks in WTNs exhibit temporally-dependent attack behaviors, sequentially-varying attack purposes over time, and delayed negative impacts. Traditional methods (e.g., AR, MA, ARMA, ARIMA, arrival density of point process, change point detection) mostly model the patterns of data points at each timestamp. However, we identify that partitioning the sensing data into a sequence of time segments can help to better describe *delayed* and *continued effect* of attacks. Therefore, we propose to segment the sensing data into a sequence of time segments. We then exploit a sequence-to-sequence (seq2seq) embedding method to characterize the temporal dependencies within each time segment. To improve the seq2seq method, we develop a new neural reconstruction structure to reconstruct not just a time segment, but also first and second derivatives of momentum of the time segment. In this way, the improved seq2seq method can have the awareness of values, acceleration, and jerk (second order derivatives) of sensor measurements. Through this module, we obtain the temporal embedding of each time segment of each sensor.

The second module is to integrate spatial effects. The effects of cyber attacks in WTNs will spatially diffuse and propagate to other sensors over time. Therefore, exploring the propagation structure can significantly model attack patterns and improve detection accuracy. However, how can we capture the spatial structures of propagation? The topology of WTNs is a graph of interconnected sensors. We map the temporal

embedding of one time segment of a sensor to the graph of WTNs as node attributes. We construct the Spatio-Temporal Graphs (STGs), where a node is a sensor and an attribute is the temporal embedding of the sensor, to describe the state of the WTNs. In this way, the STGs not only contain spatial connectivity among different sensors, but also include temporal patterns by mapping temporal embedding. We develop graph embedding model to jointly learn the state representations of the WTNs from the relational STGs.

Phase 2: Improving Unsupervised One-Class Detection Model. In reality, WTNs are mostly normal yet with a small number of cyber attack events, so the attack data samples are quite rare. This trait makes the data distributions of normal and attack extremely imbalanced. How can we overcome the imbalanced data distribution to accurately detect cyber attack? One-class detection fits well this problem. In particular, one-class SVM (OC-SVM) is a popular detection model, in which a hyper-plane is identified to divide normal and abnormal data samples after being mapped to a high-dimensional space by kernel functions. While vanilla OC-SVM achieves promising results, the kernel functions can be improved by exploiting the similarities between data samples. Specifically, we propose a new pair-wise kernel function to augment the distance between normal and attack patterns by preserving similarity across different data samples. Consequently, normal data samples are grouped into a cluster, while abnormal data samples are pushed away from normal data. In our framework, we feed the learned state representations of the WTN into the improved OC-SVM to use the pairwise kernel to detect attacks.

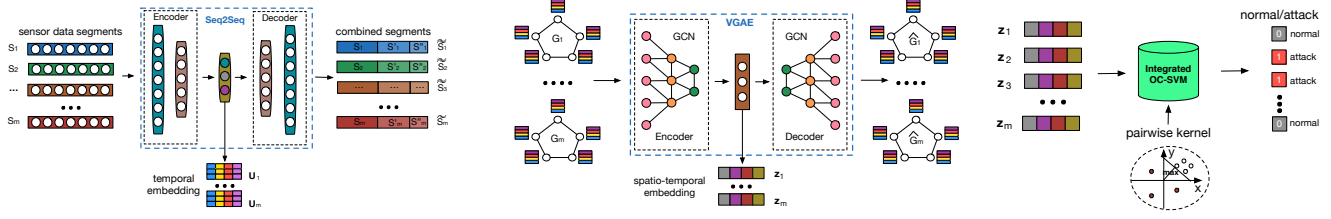
In summary, we develop a structured detection framework for cyber attack detection in WTNs. Our contributions are as follows: (1) we investigate an important problem of defending critical graph-structured infrastructures via cyber attack detection, which is important for building resilient and safe communities. (2) we develop a structured detection framework to maximize the margin between normal and attack patterns, by integrating spatio-temporal knowledge, deep representation learning, and pairwise one-class detection. (3) we implement and test our proposed framework with real-world water treatment network data and demonstrate the enhanced performance of our method. Specifically, our detection method achieves an accuracy of 91.65%, with average improvement ratios of 82.78% and 22.96% with respect to F1 and AUC, compared with baseline methods.

II. PROBLEM STATEMENT AND FRAMEWORK OVERVIEW

We first introduce the statement of the problem, and then present an overview of the framework.

A. Problem Statement

We aim to study the problem of cyber attack detection using the sensing data of WTNs. We observe that cyberattacks in WTNs exhibit not just spatial diffusion patterns, but also two temporal effects (i.e., delayed and continued). As a result, we partition the sensing data streams into non-overlapped time segments. We investigate detection on the time segment level.



(a) **P1:** Embedding time segments sequential patterns. (b) **P1:** Modeling spatio-temporal patterns over STG. (c) **P2:** Anomaly detection with data similarity.

Fig. 2. The overview of cyber attack detection framework in water treatment network

Definition 2.1: The WTN Attack Detection Problem. Formally, assuming a WTN consists of N sensors, given the sensing data streams of a WTN, we evenly divide the streams into m non-overlapped segments by every K sensory records. Consequently, we obtain a segment sequence $S = [S_1, S_2, \dots, S_i, \dots, S_m]$, where the matrix $S_i \in \mathbb{R}^{N \times K}$ is the i -th segment. Each segment is associated with a cyber attack status label: if a cyber attack happens within S_i , the label of this segment is marked as $y_i = 1$; Otherwise, $y_i = 0$. The objective is to develop a framework that takes the segment sequence S as inputs, and output the corresponding cyber attack labels for each segment to maximize detection accuracy.

B. Framework Overview

Figure 2 shows that our framework includes two phases: (1) Spatio-temporal representation learning (**P1**); (2) Improving unsupervised one-class detection (**P2**). Specifically, there are two modules in Phase 1: (a) Embedding time segment sequential patterns, in which a seq2seq model is leveraged to capture the temporal dependencies within a segment. Later, the learned representations are attached to each node (sensor) in the WTNs as node attributes to construct STGs. Be sure to notice that temporal patterns are integrated by attaching temporal embeddings as attributes; and spatial connectivity is integrated via a graph structure of sensors, which is introduced next. (b) Modeling spatio-temporal patterns over STGs, in which the fused embedding is learned through an encode-decode paradigm integrated with Graph Convolutional Network (GCN). The fused embedding summarizes the information of STGs to profile the spatio-temporal characteristics in WTNs. Finally, the Phase 2 exploits the fused embedding as inputs to detect attacks. The Phase 2 has one module, namely anomaly detection with pairwise segment similarity awareness. Specifically, the fused embedding is fed into an improved one-class anomaly detector integrated with awareness of pairwise segment similarity. Specifically, the similarities between two different segment embedding vectors are introduced to the pairwise kernel of the detector to augment the distance between normal and attack patterns.

III. PROPOSED METHOD

We first introduce time segment embedding, then illustrate the construction of spatio-temporal graphs using temporal embedding and sensor networks, present spatio-temporal graph-based representation learning, and, finally, discuss the integration with one-class detection.

A. Embedding Time Segments Sequential Patterns

We first model the sequential patterns of time segments. The sequential patterns of WTN involves two essential measurements: (1) changing rate and (2) the trend of changing rate, which correspond to the first and second derivatives respectively. Therefore, in addition to the raw data points in one segment, we introduce both the first and second derivatives to quantify the sequential patterns, resulting in an augmented segment. Formally, below we define the augmented segment.

Definition 3.1: Augmented Segment. Given a sensory data segment denoted by $S_i = [v_i^1, v_i^2, \dots, v_i^k, \dots, v_i^K]$, where $v_i^k \in \mathbb{R}^{N \times 1}$ denotes the sensory measurements of all the sensors of the i -th segment at the k -th record. Then, the corresponding first-order derivative segment S'_i is $S'_i = [\frac{\partial S_i}{\partial v_i^1}, \frac{\partial S_i}{\partial v_i^2}, \dots, \frac{\partial S_i}{\partial v_i^K}]$, and the corresponding second-order derivative segment S''_i is $S''_i = [\frac{\partial^2 S_i}{\partial v_i^1 \partial v_i^1}, \frac{\partial^2 S_i}{\partial v_i^1 \partial v_i^2}, \dots, \frac{\partial^2 S_i}{\partial v_i^K \partial v_i^K}]$. The augmented segment \tilde{S}_i is then defined as the concatenation of the raw segment, the first-order derivative segments, and the second-order derivative segments: $\tilde{S}_i = [S_i, S'_i, S''_i]$. For convenience, \tilde{S}_i also can be denoted as $\tilde{S}_i = [r_i^1, r_i^2, \dots, r_i^{3K-3}]$, where elements in $[r_i^1, r_i^2, \dots, r_i^K]$ corresponds to each element in S_i , elements in $[r_i^{K+1}, r_i^{K+2}, \dots, r_i^{2K-1}]$ corresponds to each element in S'_i , and the elements in $[r_i^{2K}, r_i^{2K+1}, \dots, r_i^{3K-3}]$ corresponds to each element in S''_i respectively.

We here provide an example of constructing an augmented segment. Suppose there are two sensors in WTNs, there are three measurement records in each time segment. In such WTN, N is 2 and K is 3. Considering the i -th segment $S_i = [[1, 3, 4], [2, 8, 5]]$, the size of S_i is 2×3 . We then calculate the S'_i by row: $S'_i = [[2, 1], [6, -3]]$. Afterward, $S''_i = [[-1], [-9]]$. Finally, we concatenate these three segments by row: $\tilde{S}_i = [[1, 3, 4, 2, 1, -1], [2, 8, 5, 6, -3, -9]]$.

Figure 2(a) shows the process of temporal embedding. The temporal embedding process develops a seq2seq model based on the encoder-decoder paradigm that takes a non-augmented segment as inputs, and reconstructs the corresponding augmented segment. The objective is to minimize the loss between the original augmented segment and the reconstructed one. Next, we provide an example about how our model operates the i -th segment S_i .

The encoding step feeds the segment S_i into a seq2seq encoder, and outputs the latent representation of the segment U_i . Formally, as shown in Equation 1, given the segment data

$\mathbf{S}_i = [\mathbf{v}_i^1, \mathbf{v}_i^2, \dots, \mathbf{v}_i^K]$, the first hidden state \mathbf{h}^1 is calculated by the first time step value. Then recursively, the hidden state of the previous time step \mathbf{h}^{t-1} and the current time step value \mathbf{v}_i^t are fed into a LSTM model to produce the current time step hidden state \mathbf{h}^t . Finally, we concatenate all of the hidden states by row (sensor) to obtain the latent feature matrix \mathbf{U}_i .

$$\begin{cases} \mathbf{h}^1 = \sigma(\mathbf{W}_e \mathbf{v}_i^1 + \mathbf{b}_e), \\ \mathbf{h}^t = LSTM([\mathbf{v}_i^t, \mathbf{h}^{t-1}]), \forall t \in \{2, \dots, K\}, \\ \mathbf{U}_i = CONCAT(\mathbf{h}^1, \mathbf{h}^2, \dots, \mathbf{h}^K), \end{cases} \quad (1)$$

where \mathbf{W}_e and \mathbf{b}_e are the weight and bias of the encoding step, respectively.

In the decoding step, the decoder takes \mathbf{U}_i as inputs and generates a reconstructed augmented segment: $[\hat{\mathbf{r}}_i^1, \hat{\mathbf{r}}_i^2, \dots, \hat{\mathbf{r}}_i^{3K-3}]$. Formally, as shown in Equation 2, the first hidden state $\hat{\mathbf{h}}^1$ of the decoder is copied from the last hidden state of encoder \mathbf{h}^K . Then, the previous time step hidden state $\hat{\mathbf{h}}^{t-1}$, the previous time step element $\hat{\mathbf{r}}_i^{t-1}$, and the latent feature vector \mathbf{U}_i are input into the LSTM model to produce the current time step hidden state $\hat{\mathbf{h}}^t$. Finally, the reconstructed value of current time step $\hat{\mathbf{r}}_i^t$ is produced by current hidden state $\hat{\mathbf{h}}^t$ that is activated by sigmoid function σ .

$$\begin{cases} \hat{\mathbf{h}}^1 = \mathbf{h}^K, \\ \hat{\mathbf{r}}_i^1 = \sigma(\mathbf{W}_d \hat{\mathbf{h}}^1 + \mathbf{b}_d), \\ \hat{\mathbf{h}}^t = LSTM([\hat{\mathbf{r}}_i^{t-1}, \hat{\mathbf{h}}^{t-1}, \mathbf{U}_i]), \forall t \in \{2, \dots, K\}, \\ \hat{\mathbf{r}}_i^t = \sigma(\mathbf{W}_d \hat{\mathbf{h}}^t + \mathbf{b}_d), \forall t \in \{2, \dots, K\}, \end{cases} \quad (2)$$

where \mathbf{W}_d and \mathbf{b}_d are the weight and bias for the decoding step respectively.

After the decoding step, we obtain the reconstructed augmented segment sequence $[\hat{\mathbf{r}}_i^1, \hat{\mathbf{r}}_i^2, \dots, \hat{\mathbf{r}}_i^{3K-3}]$. The objective is to minimize the reconstruction loss between the original and reconstructed augmented segment sequence. The overall loss is denoted as

$$\min \sum_{i=1}^m \sum_{k=1}^{3K-3} \|\mathbf{r}_i^k - \hat{\mathbf{r}}_i^k\|^2. \quad (3)$$

Along this line, we obtain the latent temporal embedding at the i -th time segment, denoted by \mathbf{U}_i .

B. Temporal Embedding as Node Attributes: Constructing Spatio-temporal Graphs

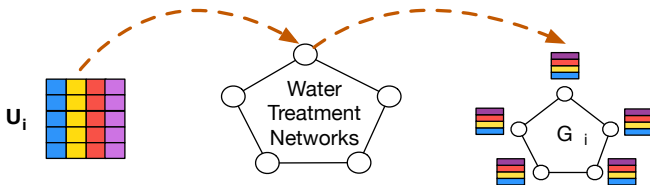


Fig. 3. The illustration of constructing spatio-temporal graphs.

The temporal embedding, obtained by Section III-A, describes and models the temporal effects of cyber attacks. Then, to further incorporate the spatial effects of WTNs, we map the temporal embedding to WTNs as node attributes.

Taking the temporal embedding of the i -th segment \mathbf{U}_i as an example. Since each row of \mathbf{U}_i is a temporal embedding of a segment of one sensor (node), we mapped each row of \mathbf{U}_i to the corresponding node (sensor) as attributes, resulting in an attributed WTNs G_i , which we call a *Spatio-temporal Graph* (STG) that preserves both the spatial and temporal effects.

C. Learning Representations of STGs

Figure 2(b) shows that we develop a spatiotemporal graph representation learning framework to preserve not just temporal patterns, but also spatial patterns in a latent embedding space. We take the STG of the i -th time segment, G_i , as an example to explain the representation process. Formally, we denote G_i by $G_i = (\mathbf{U}_i, \mathbf{A}_i)$, where \mathbf{A}_i is an adjacency matrix that describes the connectivity among different sensors; \mathbf{U}_i is a feature matrix that is formed by the temporal embedding of all the sensors of the i -th time segment. The representation learning process is formulated as: given the i -th STG G_i , the objective is to minimize the reconstruction loss between the input G_i and the reconstructed graph \hat{G}_i , by an encoding-decoding framework, in order to learn a latent embedding \mathbf{z}_i .

The neural architecture of the encoder includes two Graph Convolutional Network (GCN) layers. The first GCN layer take \mathbf{A}_i and \mathbf{U}_i as inputs, and then outputs the lower-dimensional feature matrix $\hat{\mathbf{U}}_i$. Specifically, the encoding process of the first GCN layer is given by:

$$\begin{aligned} \hat{\mathbf{U}}_i &= RELU(GCN(\mathbf{U}_i, \mathbf{A}_i)) \\ &= RELU(\hat{\mathbf{D}}_i^{-\frac{1}{2}} \mathbf{A}_i \hat{\mathbf{D}}_i^{-\frac{1}{2}} \mathbf{U}_i \mathbf{W}_0) \end{aligned} \quad (4)$$

where $\hat{\mathbf{D}}_i$ is the diagonal degree matrix of G_i , and \mathbf{W}_0 is the weight matrix of the first GCN layer.

Since the latent embedding \mathbf{z}_i of the graph is sampled from one prior normal distribution, here the purpose of the second GCN layer is to assess the parameters of the prior distribution. This layer takes \mathbf{A}_i and $\hat{\mathbf{U}}_i$ as the input, then produces the mean value $\boldsymbol{\mu}$ and the variance value δ^2 of the prior normal distribution as the output. Thus the encoding process of the second GCN layer can be formulated as

$$\boldsymbol{\mu}, \log(\delta^2) = GCN(\hat{\mathbf{U}}_i, \mathbf{A}_i) = \hat{\mathbf{D}}_i^{-\frac{1}{2}} \mathbf{A}_i \hat{\mathbf{D}}_i^{-\frac{1}{2}} \hat{\mathbf{U}}_i \mathbf{W}_1, \quad (5)$$

where \mathbf{W}_1 is the weight matrix of the second GCN layer. Then we utilize the reparameterization trick to mimic the sample operation to construct the latent representation \mathbf{z}_i . The process is formulated as

$$\mathbf{z}_i = \boldsymbol{\mu} + \boldsymbol{\delta} \times \epsilon, \quad (6)$$

where $\epsilon \sim \mathcal{N}(0, 1)$.

The decoding step takes the latent representation \mathbf{z}_i as the input and outputs the reconstructed adjacent matrix $\hat{\mathbf{A}}_i$. The decoding process is denoted as

$$\hat{\mathbf{A}}_i = \sigma(\mathbf{z}_i \mathbf{z}_i^T). \quad (7)$$

In addition, the core calculation of the decoding step can be denoted as $\mathbf{z}_i \mathbf{z}_i^T = \|\mathbf{z}_i\| \|\mathbf{z}_i^T\| \cos \theta$. Owing to the \mathbf{z}_i is the node level representation, the inner product calculation is helpful to capture the correlation among different sensors.

We minimize the joint loss function \mathcal{L}_g during the training phase, which is formulated as Equation 8. \mathcal{L}_g includes two parts. The first part is Kullback-Leibler divergence between the distribution of \mathbf{z}_i and the prior standard normal distribution denoted by $\mathcal{N}(0,1)$. The second part is the squared error between \mathbf{A}_i and $\hat{\mathbf{A}}_i$. Our training purpose is to make the $\hat{\mathbf{A}}_i$ as similar as \mathbf{A}_i , and to let the distribution of \mathbf{z}_i as close as $\mathcal{N}(0,1)$. The total loss is denoted as

$$\mathcal{L}_g = \sum_{i=1}^m \underbrace{KL[q(\mathbf{z}_i|\mathbf{X}_i, \mathbf{A}_i)||p(\mathbf{z}_i)]}_{\text{KL Divergence between } q(\cdot) \text{ and } p(\cdot)} + \underbrace{\sum_{j=1}^w \|\mathbf{A}_i - \hat{\mathbf{A}}_i\|^2}_{\text{Loss between } \mathbf{A}_i \text{ and } \hat{\mathbf{A}}_i} \quad (8)$$

When the model converges, we apply the global average aggregation to \mathbf{z}_i . Then the \mathbf{z}_i becomes the graph-level representation of the WTNs, which contains the spatio-temporal information of the whole system at i -th time segment.

D. One-Class Detection with Data Similarity Awareness

In reality, most of sensor data are normal, and attacks related data are scarce and expensive. This indeed results into the problem of unbalanced training data. How can we solve the problem? Can we develop a solution that only uses normal data for attack detection? This is the key algorithm challenge for this phase. One-class classification is a promising solution that aims to find a hyperplane to distinguish normal and attack patterns only using normal data. Specifically, OC-SVM is a classical one-class classification model. OC-SVM includes two steps: (1) mapping low dimensional data into a high dimensional feature space by a kernel function. (2) learning the parameters of hyper-plane to divide normal and abnormal data via optimization.

Intuitively, in the hyperspace provided by OC-SVM, the normal (or abnormal) data are expected to be closer, while there should be a large distance between normal and abnormal data. In other words, similar data points should be closer to each other than dissimilar ones. However, traditional kernel functions (e.g., linear, nonlinear, polynomial, radial basis function (RBF), sigmoid) cannot preserve such characteristic well. How can we make data samples well-separated in order to achieve such characteristic? To address the question, we propose a new pairwise kernel function that is capable of reducing the distances between similar data points, while maximizing the distances between dissimilar ones. Formally, given the representation matrix $\mathbf{Z} = [\mathbf{z}_1, \dots, \mathbf{z}_i, \dots, \mathbf{z}_m]$, the pairwise kernel function is given by :

$$\text{Kernel} = \tanh \left(\frac{1}{\mathcal{D}(\mathbf{Z})} \mathbf{Z}\mathbf{Z}^T + \text{sim}(\mathbf{Z}, \mathbf{Z}^T) + \mathbf{c} \right) \quad (9)$$

where \mathbf{Z}^T is the transpose of \mathbf{Z} , $\mathcal{D}(\mathbf{Z})$ is the covariance matrix of \mathbf{Z} , and $\text{sim}(\mathbf{Z}, \mathbf{Z}^T) \in \mathbb{R}^{N \times N}$ is the pairwise similarity matrix between segments. Compared with the vanilla sigmoid kernel function, we add $\text{sim}(\mathbf{Z}, \mathbf{Z}^T)$, where the range of $\text{sim}(\mathbf{Z}, \mathbf{Z}^T)$ is $[-1, 1]$. If two segments are more similar, the

corresponding value in $\text{sim}(\mathbf{Z}, \mathbf{Z}^T)$ is closer to 1. Otherwise, the value is closer to -1 . Therefore, when two segments are similar (e.g., both are normal or abnormal samples), the proposed pairwise kernel function will push these two segments closer; otherwise, these two segments will be set away from each other.

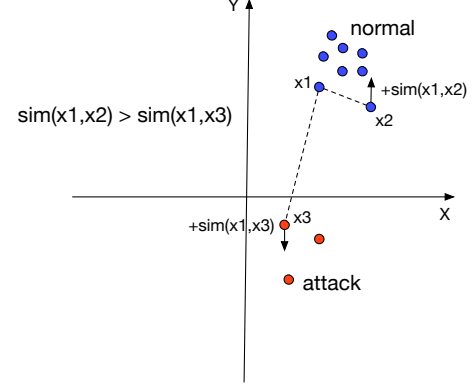


Fig. 4. The illustration of pairwise kernel, given normal data x_1 , owing to x_2 is normal and x_3 is attack, $\text{sim}(x_1, x_2) > \text{sim}(x_1, x_3)$ and the directions of $\text{sim}(x_1, x_2)$ and $\text{sim}(x_1, x_3)$ are opposite. Pairwise kernel increase the distance between x_2 and x_3 .

The pairwise kernel is able to enlarge the distance among different category samples in feature space, which makes the OC-SVM converge more easily and detect cyber attacks more accurate. Figure 2(c) shows the detection process of cyber attacks. The spatio-temporal embedding \mathbf{z}_i is fed into the integrated OC-SVM to detect cyber attacks by utilizing the pairwise kernel function, and to output the corresponding status labels of WTNs, to indicate whether a cyber attack happen or not at the i -th time segment.

E. Comparison with Related Work

Recently, lots of attempts have been made to detect cyber attacks in WTNs. For instance, Lin *et al.* utilized a probabilistic graphical model to preserve the spatial dependency among sensors in WTNs and a one-class classifier to detect cyber attacks [5]. Li *et al.* regarded the LSTM and RNN as the basic model of the GAN framework to develop an anomaly detection algorithm to detect cyber attacks in WTNs [3]. Raciti *et al.* constructed one real-time anomaly detection system based on cluster model [6]. However, these models exhibit several limitations when detecting cyber attacks: (i) the changing trend of sensing data in a time segment is not preserved; (ii) the spatial patterns among sensors are captured partially; (iii) the similarity between different data samples is not utilized completely.

In order to overcome these limitations, we propose a new spatio-temporal graph (STG) to preserve and fuse spatio-temporal effects of WTNs simultaneously. Moreover, a new pairwise kernel that utilizing the data similarity to augment the distance among different patterns is also proposed to improve the accuracy of cyber attack detection.

IV. EXPERIMENTAL RESULTS

We conduct experiments to answer the following research questions:

- (1) Does our proposed outlier detection framework (STOD) outperforms the existing methods?
- (2) Is the spatio-temporal representation learning component of STOD necessary for improving detection performance?
- (3) Is the proposed pairwise kernel better than other traditional kernels for cyber attack detection?
- (4) How much time will our method and other methods cost?

A. Data Description

We used the secure water treatment system (SWAT) data set that is from Singapore University of Technology and Design for our study. The SWAT has project built one water treatment system and a sensor network to monitor and track the situations of the system. Then, they construct one attack model to mimic the cyber attack of this kind of system in the real scenario. The cyber attacks to and the sensory data of the system are collected to form the SWAT dataset. Table I show some important statistics of the SWAT dataset. Specifically, the SWAT dataset include a normal set (no cyber attacks) and an attack set (with cyber attacks). The time period of the normal data is from 22 December 2015 to 28 December 2015. The time period of the attack data is from 28 December 2015 to 01 January 2016, and 01 February 2016. There is no time period overlap between the normal data and the attack data on 28 January 2015. It is difficult to identify more water treatment network datasets. In this study, we focus on validating our method using this dataset.

TABLE I
STATICS OF THE SWAT DATA SET

| Data Type | Sensor Count | Total Items | Attack Items | Pos/Neg |
|-----------|--------------|-------------|--------------|---------|
| Normal | 51 | 496800 | 0 | - |
| Attack | 51 | 449919 | 53900 | 7:1 |

B. Evaluation Metrics

We evaluate the performances of our method in terms of four metrics. Given a testset, a detection model will predict a set of binary labels (1: attack; 0: normal). Compared predicted labels with golden standard benchmark labels, we let tp , tn , fp , fn be the sizes of the true positive, true negative, false positive, false negative sets, respectively.

- (1) **Accuracy:** is given by:

$$Accuracy = \frac{tp + tn}{tp + tn + fp + fn} \quad (10)$$

- (2) **Precision:** is given by:

$$Precision = \frac{tp}{tp + fp} \quad (11)$$

- (3) **F-measure:** is the harmonic mean of precision and recall, which is given by:

$$F - measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (12)$$

- (4) **AUC:** is the area under the ROC curve. It shows the capability of a model to distinguish between two classes.

C. Baseline Algorithms

We compare the performances of our method (STOD) against the following ten baseline algorithms.

- (1) DeepSVDD [7]: expands the classic SVDD algorithm into a deep learning version. It utilizes a neural network to find the hyper-sphere of minimum volume that wraps the normal data. If a data sample falls inside of the hyper-sphere, DeepSVDD classifies the sample as normal, and attack otherwise. In the experiments, we set the dimensions of the spatio-temporal embedding \mathbf{z}_i to 28×28 .
- (2) GANomaly [8]: is based on the GAN framework. It develop a new version of generator by using the encoder-decoder-encoder structure. The algorithm regards the difference between the embedding of the first encoder and the embedding of the second encoder as the anomaly score to distinguish normal and abnormal. In the experiments, we set the dimension of the spatio-temporal embedding vector \mathbf{z}_i into 28×28 .
- (3) LODA [9]: is an ensemble outlier detection model. It collects a series of weak anomaly detectors to produce a strong detector. In addition, the model fits real-time data flow and is resistant to missing values in the data set. In the experiments, we fed the learned representations into the LODA to detect.
- (4) Isolation-Forest [10]. The IsolationForest isolates observations by randomly selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected feature. In the experiments, we input spatio-temporal embedding vector \mathbf{z}_i into Isolation-Forest, and set the number of estimators = 100, max sample numbers = 256.
- (5) LOF [11]. The principle of LOF is to measure the local density of data samples. If one data sample has low local density, the sample is an outlier. Otherwise, the sample is a normal sample. In the experiments, we input the spatio-temporal embedding vector \mathbf{z}_i into LOF and set the number of neighborhood = 20, the distance metric for finding neighborhoods is euclidean distance.
- (6) KNN [12]. KNN selects k nearest neighborhoods of one data sample based on a distance metric. KNN calculates the anomaly score of the data sample according to the anomaly situation of the k neighborhoods. In the experiments, we input spatio-temporal embedding vector \mathbf{z}_i into KNN, and set the number of neighborhoods = 5, the adopted distance metric is euclidean distance.
- (7) ABOD [13]. The ABOD method uses angle as a more robust measure to detect outliers. If many neighborhoods of one sample locate in the same direction to the sample, it is an outlier, otherwise, it is a normal sample. In the experiments, we input spatio-temporal embedding \mathbf{z}_i into ABOD, set $k = 10$. The angle metric is cosine value.
- (8) STODP1. We proposed to partition the sensing data into non-overlapped segments. The global mean pooling tech-

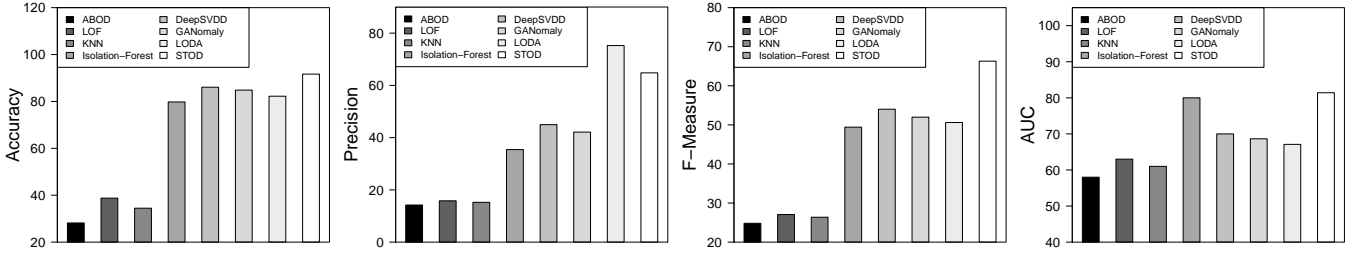


Fig. 5. Comparison of different models in terms of Accuracy, Precision, F-measure and AUC .

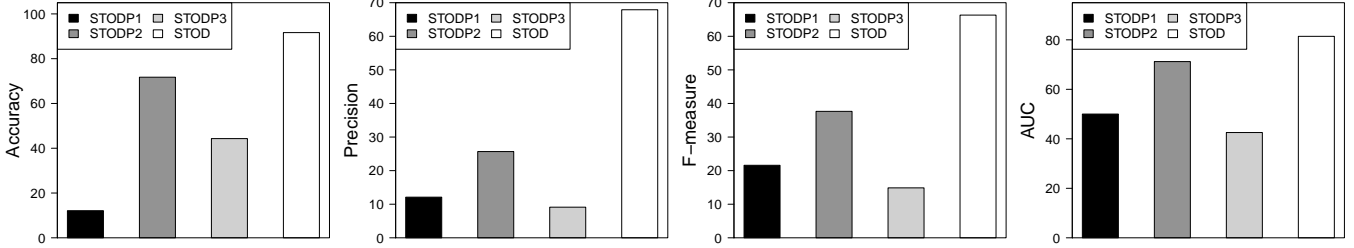


Fig. 6. Comparison of different phases of representation learning module based on Accuracy, Precision, F-measure, and AUC.

nique was then applied to fuse the segments of different sensors into an averaged feature vector. We fed the fused feature vector into OC-SVM for outlier detection. The kernel of OC-SVM is defined in Equation 9.

- (9) STODP2. We applied the global mean pooling to the temporal embedding vectors generated by Section 3.A to obtain a global feature vector of WTN, which was fed into OC-SVM for outlier detection. In addition, the kernel of OC-SVM is our proposed kernel function defined in Equation 9.
- (10) STODP3. In order to study the effect of Seq2Seq, we remove the Seq2Seq module of our framework pipeline. The temporal segments of different sensors are organized as graph set. The graph set is input into graph embedding module to obtain the final embedding. Finally, the embedding is input into OC-SVM to do outlier detection. The kernel of the OC-SVM is defined in Equation 9.

In the experiments, the spatio-temporal representation learning phase of our framework is used to preserve the spatio-temporal patterns and data characteristics into feature learning. The one-class outlier detection phase of our framework is used to detect the cyber attack status of the water treatment system based on the spatio-temporal representation. We only use normal data to train our model. After the training phase, our model has the capability to detect the status of the testing data set that contains both normal and attack data. All the evaluations are performed on a x64 machine with Intel i9-9920X 3.50GHz CPU and 128GB RAM. The operating system is Ubuntu 18.04.

D. Overall Performances

We compare our method with the baseline models in terms of accuracy, precision, f-measure and AUC. Figure 5 shows the average performances of our method (STOD) is the best in terms of accuracy, f-measure and AUC; our method ranks

second in terms of precision, compared with other baseline models. A potential interpretation of such observation is that the STOD captures the temporal effects (**delayed**, **continued**) and spatial effect (**cascading**) of cyber attacks by spatio-temporal representation learning part of STOD in a balanced way. With STOD captures more intrinsic features of cyber attacks, the model not only finds more attack samples but also makes fewer mistakes on normal samples. Thus, the distinguishing ability of STOD is improved greatly. But on a single evaluation metric, STOD maybe poorer than other baselines. Overall, STOD outperforms with respect to Accuracy, F-measure and ACU compared with baseline models, which signifies our detection framework owns the best attack detection ability.

Another observation is that the performances of LOF, ABOD, and KNN are much worse than other models. The possible reason is that these models exploit distance or angle-based assessment strategies. These geometrical measurements are vulnerable after projecting data into high dimensional space due to the “curse of dimensionality”. Thus, these models can not achieve excellent performances.

E. Study of Representation Learning

The representation learning phase of our framework include: (1) partitioning sensor data streams into segments; (2) modeling the temporal dependencies with seq2seq; (3) modeling the spatial dependencies with graph embedding. What role does each of the three steps play in our framework? We will iteratively remove each of the three steps to obtain three different variants, namely STODP1, STODP2, STODP3. We then test compare the three variants with our original framework to examine the importance of the removed step for improving detection performances

Figure 6 shows the experimental results of STOD, STODP1, STODP2, and STODP3, which clearly show that STOD out-

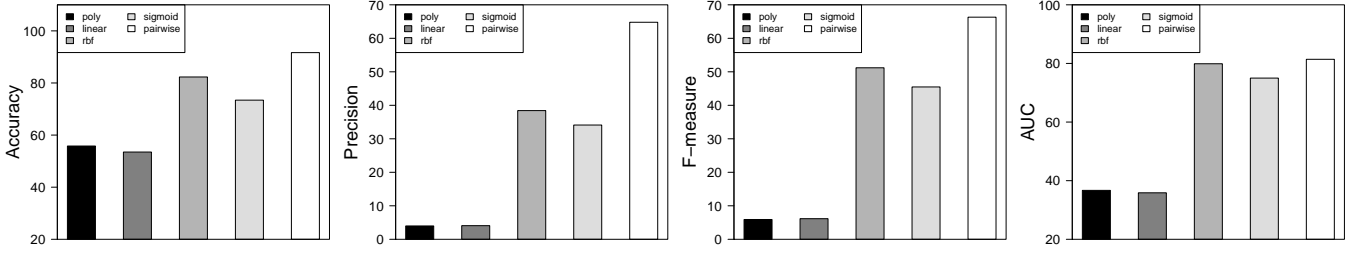


Fig. 7. Comparison of different kernels with respect to Accuracy, Precision, F-measure, and AUC.

performs STODP1, STODP2, and STODP3 in terms of accuracy, precision, f-measure, and AUC with a large margin. A reasonable explanation of this phenomenon is that attack patterns are spatially and temporally structured, and, thus, when more buried spatio-temporal patterns are modeled, the method becomes more discriminant. The results validate the three steps (segmentation, temporal, spatial) of the representation learning phase is critical for attack pattern characterization.

F. Study of Pairwise Kernel Function

The kernel function is vital for the SVM based algorithm family. An effective kernel function can map challenging data samples into a high-dimensional space, and make these data samples more separable in the task of detection. We design experiments to validate the improved performances of our pairwise kernel function by comparing our pairwise kernel function with other baseline kernel functions. Specifically, the baseline kernels are as follows:

- (1) **linear**. This kernel is a linear function. There are limited number of parameters in the linear kernel, so the calculation process is quick. The dimension of the new feature space is similar to the original space.
- (2) **poly**. This kernel is a polynomial function. The parameters of the kernel are more than the linear kernel. It maps data samples into high dimensional space.
- (3) **rbf**. This kernel is a Gaussian function that is a non-linear function. It exhibits excellent performance in many common situations.
- (4) **sigmoid**. This kernel is a sigmoid function. When SVM utilizing this function to model data samples, the effect is similar to using a multi-layer perceptron.

Figure 7 shows a comparison between our kernel and other baseline kernels with respect to all evaluation metrics. We observed that our kernel shows significant improvement, compared with other baseline kernels, in terms of Accuracy, Precision, F-measure, and AUC. This indicates that our kernel can effectively augment the attack patterns in original data, and maximize the difference between normal and attack patterns, by mapping original data samples into high dimensional feature space. This experiment validates the superiority of our pairwise kernel function.

G. Study of Time Costs

We aim to study the time costs of training and testing in different models. Specifically, we divided the dataset into six

non-overlap folds. We then used cross-validation to evaluate the time costs of different models.



Fig. 8. Comparison of different models based on training time cost

Figure 8 shows the comparison of training time costs among different models. We find that the training time costs of each model is relatively stable. An obvious observation is GANomaly has the largest training time cost than other models. This is because the encoder-decoder-encoder network architecture design is time-consuming. In addition, the training time of STOD is slightly longer than OC-SVM. This can be explained by the fact that the similarity calculation of pairwise kernel function increases time costs, since we need to calculate the similarities between two representation vectors of each training data sample.

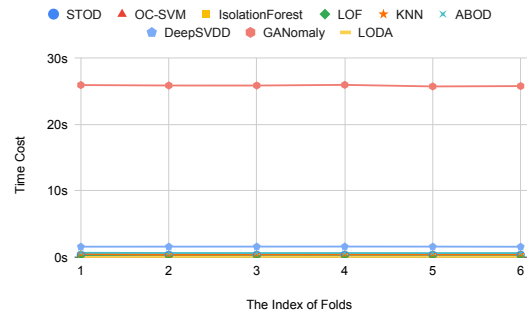


Fig. 9. Comparison of different models based on testing time cost.

Figure 9 shows the comparisons of testing time costs among different models. The testing time costs of each model are relatively stable as well. And many of them can complete the testing task within one second, except GANomaly. We find the testing time of our method is shorter than the training time by comparing Figure 8 and Figure 9. This can be explained by a strategy of our method: once the model training is completed, our method stores the kernel mapping parameters in order to

save time of computation. In addition, GANomaly still shows the largest testing time cost. The reason is that the testing phase of GANomaly needs to calculate two representation vectors of each testing data samples, and, thus, GANomaly doesn't use less time, compared with that of the training phase.

H. Case Study: Visualization for Spatio-temporal Embedding

The spatio-temporal representation learning phase is an important step in our structured detection framework. An effective representation learning method should be able to preserve the patterns of normal or attack behaviors and maximize the distances between normal and attack in the detection task. We visualize the spatio-temporal embedding on a 2-dimensional space, in order to validate the discriminant capabilities of our learned representations. Specifically, we first select 3000 normal and 3000 attack spatio-temporal embedding respectively. We then utilize the T-SNE manifold method to visualize the embedding. Figure 10 shows the visualization results of normal and attack data samples. We find that our representation learning result is discriminant in a transformed 2-dimensional space. As can be seen, the learned normal and attack representation vectors are clustered together to form dense areas. The observation shows that non-linear models are more appropriate for distinguishing normal and attack behaviors than linear methods.

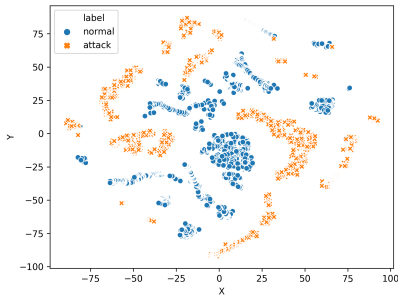


Fig. 10. visualization result for spatio-temporal embedding

V. RELATED WORK

Representation Learning. Representation learning is to learn a low-dimensional vector to represent the given data of an object. Representation learning approaches are three-fold: (1) probabilistic graphical models; (2) manifold learning approaches; (3) auto-encoder and its variants; The main idea of the probabilistic graphical model is to learn an uncertain knowledge representation by a Bayesian network [14], [15]. The key challenge of such methods is to find the topology relationship among nodes in the probabilistic graphical model. The manifold learning methods utilize the non-parametric approach to find manifold and embedding vectors in low dimensional space based on neighborhood information [16], [17]. However, manifold learning is usually time-costly. The discriminative ability of such methods is very high in many applications. Recently, deep learning models are introduced to conduct representation learning. The auto-encoder model is a classical neural network framework, which embeds the non-linear

relationship in feature space via minimizing the reconstruction loss between original and reconstructed data [18]–[20]. When representation learning meets spatial data, autoencoders can integrate with spatio-temporal statistical correlations to learn more effective embedding vectors. [21]–[23]. For instance, Singh et al, use the auto-encoder framework to learn the spatio-temporal representation of traffic videos to help detect the road accidents [24]. Wang et al. utilize spatio-temporal representation learning to learn the intrinsic feature of GPS trajectory data to help analyze driving behavior [25].

Deep Outlier Detection. Outlier detection is a classical problem with important applications, such as, fraud detection and cyber attack detection. Recently, deep learning has been introduced into outlier detection. According to the availability of outlier labels, deep anomaly detection can be classified into three categories: (1) supervised deep outlier detection; (2) semi-supervised deep outlier detection; (3) unsupervised deep outlier detection. First, supervised deep outlier detection models usually train a deep classification model to distinguish whether a data sample is normal or not [26], [27]. These models are not widely available in reality, because it is difficult to obtain data labels of outliers. Meanwhile, data imbalance is a serious issue that degrades the performances of supervised models. Second, semi-supervised outlier detection methods usually train a deep auto-encoder model to learn the latent embedding of normal data [28]–[30], then the learned embedding vectors are used to accomplish outlier detection task. In deep semi-supervised outlier detection, one-class classification is an important research direction. For instance, Liu et. al, proposed to detect the anomaly data on uncertain data by SVDD algorithm [31]. Many experiments have shown the adaptability of one class SVM. Third, unsupervised outlier detection models do not need any label information, they detect outliers depends on the intrinsic rules (e.g., scores, distance, similarity) of data [32]–[34]. Such methods are appropriate for scenarios that are hard to collect label information.

Cyber Attack Detection in Water Treatment Network. Water purification plants are critical infrastructures in our local communities. Such infrastructures are usually vulnerable to cyber attacks. Early detection of cyber attacks in water treatment networks is significant for defending our infrastructure safety and public health. There are many existing studies about outlier detection in water treatment networks [2], [4], [5], [35], [36]. For instance, Adepu et al. studied the impact of cyber attacks on water distribution systems [37]. Goh et al. designed an unsupervised learning approach that regards Recurrent Neural Networks as a temporal predictor to detect attacks [1]. Inoue et al. compared the performances of Deep Neural Network and OC-SVM on outlier detection in water treatment networks [38]. Raciti et al. developed a real-time outlier detection system by clustering algorithms and deployed the system into a water treatment network [6]. However, there is limited studies that integrate deep graph representation learning, spatiotemporal patterns, and one-class detection to more effectively address cyber attack problems.

VI. CONCLUSION REMARKS

We studied the problem of cyber attack detection in water treatment networks. To this end, we proposed a structured detection framework to integrate spatial-temporal patterns, deep representation learning, and one-class detection. Specifically, we first partitioned the sensing data of WTNs into a sequence of fixed-size time segments. We then built a deep spatiotemporal representation learning approach to preserve the spatio-temporal patterns of attacks and normal behaviors. The representation learning approach includes two modules: (1) a temporal embedding module, which preserves the temporal dependencies within a time segment. Then, we constructed the spatiotemporal graphs by mapping the temporal embedding to the WTN as node attributes. (2) a spatial embedding module, which learns the fused spatio-temporal embedding from the spatiotemporal graphs. In addition, we developed an integrated one-class detection method with an improved pairwise kernel. The new kernel is capable of augmenting the difference between normal and attack patterns via the pairwise similarity among deep embedding vectors of system behaviors. Finally, we conducted extensive experiments to illustrate the effectiveness of our method: STOD achieves an accuracy of 91.65%, with average improvement ratios of 82.78% and 22.96% with respect to F1 and AUC, compared with the baseline methods.

REFERENCES

- [1] J. Goh, S. Adepu, M. Tan, and Z. S. Lee, "Anomaly detection in cyber physical systems using recurrent neural networks," in *2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)*. IEEE, 2017, pp. 140–145.
- [2] M. Romano, Z. Kapelan, and D. Savić, "Real-time leak detection in water distribution systems," in *Water Distribution Systems Analysis 2010*, 2010, pp. 1074–1082.
- [3] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, and S.-K. Ng, "Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks," in *International Conference on Artificial Neural Networks*. Springer, 2019, pp. 703–716.
- [4] C. Feng, T. Li, and D. Chana, "Multi-level anomaly detection in industrial control systems via package signatures and lstm networks," in *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2017, pp. 261–272.
- [5] Q. Lin, S. Adepu, S. Verwer, and A. Mathur, "Tabor: A graphical model-based approach for anomaly detection in industrial control systems," in *Proceedings of the 2018 Asia Conference on Computer and Communications Security*, 2018, pp. 525–536.
- [6] M. Raciti, J. Cucurull, and S. Nadjim-Tehrani, "Anomaly detection in water management systems," in *Critical infrastructure protection*. Springer, 2012, pp. 98–119.
- [7] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft, "Deep one-class classification," in *International conference on machine learning*, 2018, pp. 4393–4402.
- [8] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, "Ganomaly: Semi-supervised anomaly detection via adversarial training," in *Asian Conference on Computer Vision*. Springer, 2018, pp. 622–637.
- [9] T. Pevný, "Loda: Lightweight on-line detector of anomalies," *Machine Learning*, vol. 102, no. 2, pp. 275–304, 2016.
- [10] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 2008, pp. 413–422.
- [11] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 2000, pp. 93–104.
- [12] P. Soucy and G. W. Mineau, "A simple knn algorithm for text categorization," in *Proceedings 2001 IEEE International Conference on Data Mining*. IEEE, 2001, pp. 647–648.
- [13] H.-P. Kriegel, M. Schubert, and A. Zimek, "Angle-based outlier detection in high-dimensional data," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2008, pp. 444–452.
- [14] N. Friedman, "Inferring cellular networks using probabilistic graphical models," *Science*, vol. 303, no. 5659, pp. 799–805, 2004.
- [15] M. J. Johnson, D. K. Duvenaud, A. Wiltchko, R. P. Adams, and S. R. Datta, "Composing graphical models with neural networks for structured representations and fast inference," in *Advances in neural information processing systems*, 2016, pp. 2946–2954.
- [16] B. Zhu, J. Z. Liu, S. F. Cauley, B. R. Rosen, and M. S. Rosen, "Image reconstruction by domain-transform manifold learning," *Nature*, vol. 555, no. 7697, pp. 487–492, 2018.
- [17] W. Wang, Y. Yan, F. Nie, S. Yan, and N. Sebe, "Flexible manifold learning with optimal graph for image and video representation," *IEEE Transactions on Image Processing*, vol. 27, no. 6, pp. 2664–2675, 2018.
- [18] Y. Wang, H. Yao, and S. Zhao, "Auto-encoder based dimensionality reduction," *Neurocomputing*, vol. 184, pp. 232–242, 2016.
- [19] H.-I. Suk, S.-W. Lee, D. Shen, A. D. N. Initiative *et al.*, "Latent feature representation with stacked auto-encoder for ad/mci diagnosis," *Brain Structure and Function*, vol. 220, no. 2, pp. 841–859, 2015.
- [20] J. Calvo-Zaragoza and A.-J. Gallego, "A selectional auto-encoder approach for document image binarization," *Pattern Recognition*, vol. 86, pp. 37–47, 2019.
- [21] L. Cedolin and B. Delgutte, "Spatiotemporal representation of the pitch of harmonic complex tones in the auditory nerve," *Journal of Neuroscience*, vol. 30, no. 38, pp. 12 712–12 724, 2010.
- [22] C.-Y. Ma, M.-H. Chen, Z. Kira, and G. AlRegib, "Ts-lstm and temporal-inception: Exploiting spatiotemporal dynamics for activity recognition," *Signal Processing: Image Communication*, vol. 71, pp. 76–87, 2019.
- [23] Z. Pan, Y. Liang, W. Wang, Y. Yu, Y. Zheng, and J. Zhang, "Urban traffic prediction from spatio-temporal data using deep meta learning," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 1720–1730.
- [24] D. Singh and C. K. Mohan, "Deep spatio-temporal representation for detection of road accidents using stacked autoencoder," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 3, pp. 879–887, 2018.
- [25] P. Wang, X. Li, Y. Zheng, C. Aggarwal, and Y. Fu, "Spatiotemporal representation learning for driving behavior analysis: A joint perspective of peer and temporal dependencies," *IEEE Transactions on Knowledge and Data Engineering*, 2019.
- [26] Y. Yamanaka, T. Iwata, H. Takahashi, M. Yamada, and S. Kanai, "Autoencoding binary classifiers for supervised anomaly detection," in *Pacific Rim International Conference on Artificial Intelligence*. Springer, 2019, pp. 647–659.
- [27] Y. Kawachi, Y. Koizumi, S. Murata, and N. Harada, "A two-class hyper-spherical autoencoder for supervised anomaly detection," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 3047–3051.
- [28] L. Ruff, R. A. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft, "Deep one-class classification," in *Proceedings of the 35th International Conference on Machine Learning*, vol. 80, 2018, pp. 4393–4402.
- [29] R. Chalapathy, A. K. Menon, and S. Chawla, "Anomaly detection using one-class neural networks," *arXiv preprint arXiv:1802.06360*, 2018.
- [30] M. Zhao, L. Jiao, W. Ma, H. Liu, and S. Yang, "Classification and saliency detection by semi-supervised low-rank representation," *Pattern Recognition*, vol. 51, pp. 281–294, 2016.
- [31] B. Liu, Y. Xiao, L. Cao, Z. Hao, and F. Deng, "Svdd-based outlier detection on uncertain data," *Knowledge and information systems*, vol. 34, no. 3, pp. 597–618, 2013.
- [32] Y. Liu, Z. Li, C. Zhou, Y. Jiang, J. Sun, M. Wang, and X. He, "Generative adversarial active learning for unsupervised outlier detection," *IEEE Transactions on Knowledge and Data Engineering*, 2019.
- [33] S. Wang, Y. Zeng, X. Liu, E. Zhu, J. Yin, C. Xu, and M. Kloft, "Effective end-to-end unsupervised outlier detection via inlier priority of discriminative network," in *Advances in Neural Information Processing Systems*, 2019, pp. 5960–5973.
- [34] W. Lu, Y. Cheng, C. Xiao, S. Chang, S. Huang, B. Liang, and T. Huang, "Unsupervised sequential outlier detection with deep architectures,"

IEEE Transactions on Image Processing, vol. 26, no. 9, pp. 4321–4330, 2017.

- [35] D. T. Ramotsoela, G. P. Hancke, and A. M. Abu-Mahfouz, “Attack detection in water distribution systems using machine learning,” *Human-centric Computing and Information Sciences*, vol. 9, no. 1, p. 13, 2019.
- [36] S. Adepur and A. Mathur, “Using process invariants to detect cyber attacks on a water treatment system,” in *IFIP International Conference on ICT Systems Security and Privacy Protection*. Springer, 2016, pp. 91–104.
- [37] S. Adepur, V. R. Palleti, G. Mishra, and A. Mathur, “Investigation of cyber attacks on a water distribution system,” *arXiv preprint arXiv:1906.02279*, 2019.
- [38] J. Inoue, Y. Yamagata, Y. Chen, C. M. Poskitt, and J. Sun, “Anomaly detection for a water treatment system using unsupervised machine learning,” in *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 2017, pp. 1058–1065.