Learning Graph Structures With Transformer for Multivariate Time-Series Anomaly Detection in IoT

Zekai Chen, *Graduate Student Member, IEEE*, Dingshuo Chen, Xiao Zhang, *Member, IEEE*, Zixuan Yuan, and Xiuzhen Cheng, *Fellow, IEEE*

***Abstract*—Many real-world Internet of Things (IoT) systems, which include a variety of Internet-connected sensory devices, produce substantial amounts of multivariate time-series data. Meanwhile, vital IoT infrastructures, such as smart power grids and water distribution networks are frequently targeted by cyberattacks, making anomaly detection an important study topic. Modeling such relatedness is, nevertheless, unavoidable for any efficient and effective anomaly detection system, given the intricate topological and nonlinear connections that are originally unknown among sensors. Furthermore, detecting anomalies in multivariate time series is difficult due to their temporal dependency and stochasticity. This article presented GTA, a new framework for multivariate time-series anomaly detection that involves automatically learning a graph structure, graph convolution, and modeling temporal dependency using a transformer-based architecture. The connection learning policy, which is based on the Gumbel-softmax sampling approach to learn bidirected links among sensors directly, is at the heart of the learning graph structure. To describe the anomaly information flow between network nodes, we introduced a new graph convolution called influence propagation convolution. In addition, to tackle the quadratic complexity barrier, we suggested a multibranch attention mechanism to replace the original multihead self-attention method. Extensive experiments on four publicly available anomaly detection benchmarks further demonstrate the superiority of our approach over alternative state of the arts.**

***Index Terms*—Anomaly detection, graph learning, multivariate time series, self-attention.**

I. Introduction

DUE TO the fast rising number of Internet-connected sensory devices, the Internet of Things (IoT) infrastructure has created vast sensory data. IoT data are often characterized by their speed in terms of geographical and temporal dependency [1], [2], and it is frequently subjected to correspondingly rising abnormalities and cyberattacks [3], [4]. Many critical infrastructures constructed on the top of cyber–physical systems (CPSs) [5], such as smart power grids, water treatment and distribution networks, transportation, and autonomous cars, are especially in need of security monitoring [4], [6], [7]. As a result, an efficient and accurate anomaly detection system has a great research value because it can help with continuous monitoring of fundamental controls or indicators and promptly provide notifications for any probable anomalous occurrence.

In this work, we focus on anomaly detection for multivariate time series [8] as a copious amount of IoT sensors in many real-life scenarios consecutively generate substantial volumes of time-series data. For instance, in a secure water distribution (WADI) system [9], multiple sensing measurements, such as flowing meter, transmitting level, valve status, water pressure level, etc., are recorded simultaneously at each timestamp to form a multivariate time series. In this case, the central water treatment testbed is also known as an *entity*. It is commonly accepted to detect anomalies from the entity level instead of the sensor level since the overall status detection is generally worth more concern and less expensive. Predominantly, data from these sensors are highly correlated in a complex topological and nonlinear fashion: for example, opening a valve would result in pressure and flow rate changes, leading to further chain reactions of other sensors within the same entity following an internal mechanism. Nevertheless, the dependencies among sensors are initially hidden and somehow costly to access in most real-life scenarios, leading to an intuitive question of how to model such complicated relationships between sensors without knowing prior information?

Recently, deep learning-based techniques have demonstrated some promising improvements in anomaly detection due to the superiority in sequence modeling over high-dimensional data sets. Generally, the existing approaches can roughly fall into two lines: 1) reconstruction-based models (R-model) [6], [10]–[13] and 2) forecasting-based models (F-model) [4], [8], [14]–[17]. For example, autoencoders (AEs) [10] are a popular approach for anomaly detection, which uses reconstruction error as an outlier score. More recently, generative adversarial networks (GANs) [18], [19] based on reconstruction [6], [20] and RNN-based forecasting approaches [8], [17] have also reported promising performance for multivariate anomaly detection. However, these methods do not explicitly learn the topological structure among sensors, thus leaving room for improvements in modeling high-dimensional sensor data with considerable potential interrelationships appropriately.

Manuscript received March 31, 2021; revised May 27, 2021 and July 1, 2021; accepted July 12, 2021. Date of publication July 27, 2021; date of current version June 7, 2022. This work was supported in part by the Fundamental Research Funds of Shandong University. *(Corresponding author: Xiao Zhang.)*

Zekai Chen is with the Department of Computer Science, George Washington University, Washington, DC 20052 USA (e-mail: zech_chan@gwu.edu).

Dingshuo Chen, Xiao Zhang, and Xiuzhen Cheng are with the School of Computer Science and Technology, Shandong University, Qingdao 250100, China (e-mail: xiaozhang@sdu.edu.cn; xzcheng@sdu.edu.cn).

Zixuan Yuan is with the School of Business, Rutgers University, New Brunswick, NJ 08901 USA (e-mail: zy101@rutgers.edu).

Digital Object Identifier 10.1109/JIOT.2021.3100509

Graph convolutional networks (GCNs) [21]–[24] have recently revealed discriminative power in learning graph representations due to their permutation-invariance, local connectivity, and compositionality [21], [25]. Graph neural networks allow each graph node to acknowledge its neighborhood context by propagating information through structures. Recent works [4], [17], [26] then combined temporal modeling methods with GCNs to model the topological relationships between sensors. Specifically, most existing graph-based approaches [4], [25] aim to learn the graph structure by measuring the cosine similarity (or other distance metrics) between sensor embeddings and defining top-K closest nodes as the source node's *connections*, followed by a graph attention convolution to capture the information propagation process. However, we argue that: 1) dot products among sensor embeddings lead inevitably to quadratic time and space complexity regarding the number of sensors and 2) the tightness of spatial distance cannot entirely indicate that there exists a strong connection in a topological structure.

To address the problems above, we propose an innovative framework named graph learning with transformer for anomaly detection (GTA) in this article. We devise from the perspective of learning a global bidirected graph structure involving all sensors within the entity through a connection learning policy (LP) based on the Gumbel-Softmax Sampling trick to overcome the quadratic complexity challenge and the limitations of the top-K *nearest* strategy. The policy logits can automatically discover the hidden associations during the training process by determining whether any specific node's information should flow to the other targets to achieve the best forecasting accuracy while restricting each node's neighborhoods' scope as much as possible. The discovered hidden associations are then fed into the graph convolution layers for influence propagation (IP) modeling. We then integrate these graph convolution layers with different level dilated convolution layers to construct a hierarchical context encoding block specifically for temporal data. While recurrent mechanisms can be naturally applied to temporal dependency modeling, it is hard to parallelize in many mobile environments (e.g., IoT), which require high computation efficiency. Hence, we adopt the *Transformer* [27]-based architecture for the sequence modeling and forecasting due to the parallel efficiency and capability of capturing long-distance context information. We also propose a novel multibranch attention strategy to reduce the quadratic complexity of original self multihead attention.

The main *contributions* of our work are summarized as follows.

1) We propose a novel and differentiable *connection LP* to automatically learn the graph structure of dependency relationships between sensors. Meanwhile, each node's neighborhood field is restricted by integrating a new loss term for further inference efficiency.
2) We introduce a novel graph convolution named IP convolution to model the anomaly influence flowing process. A multiscale dilated convolution is then combined with the graph convolution to form an effective hierarchical temporal context encoding block.
3) We propose a novel multibranch attention mechanism to tackle the original multihead attention mechanism's quadratic complexity challenge.
4) We conduct extensive experiments on a wide range of multivariate time-series anomaly detection benchmarks to demonstrate the superiority of our proposed approach over state of the arts.

II. Related Work

The existing literature for addressing time-series anomaly detection usually can be divided into two major categories. The first category usually modeled each time-series variable independently, while the second category took into consideration the correlations among multivariate time series to improve the performance.

A. Anomaly Detection in Univariate Time Series

The anomaly detection in univariate time series has drawn many researchers' attentions in recent years. Traditionally, the anomaly detection frameworks included two main phases: 1) estimation phase and 2) detection phase [28]. In the estimation phase, the variable values at one timestamp or time interval can be predicted or estimated by a specific algorithm. Then, the estimated values were compared with real values based on dynamically adjusted thresholds to detect anomalies in the detection phase. For example, Zhang *et al.* [29] applied ARIMA to capture the linear dependencies between the future values and the past values, thus modeling the time-series behavior for anomaly detection. Lu and Ghorbani [30] utilized wavelet analysis to construct the estimation model. With the development of deep learning, various neural network architectures have also been applied to anomaly detection. DeepAnt [31] was an unsupervised approach using convolutional neural network (CNN) to forecast future time-series values and adopted Euclidean distance to measure the discrepancy for anomaly detection. The LSTM neural network was also widely used in modeling time-series behaviors [6], [32], [33]. The LSTM-based encoder–decoder [32] reconstructed the variable values and measured the reconstruction errors for detection.

B. Anomaly Detection in Multivariate Time Series

In real-world scenarios, the time-series data acquisition sources could be multiple [34]. Therefore, many work began to pay attention to exploiting the correlations among multiple variables to improve the accuracy of anomaly detection. Jones *et al.* [35] extracted statistical and smoothed trajectory (SST) features of time series and utilized a set of nonlinear functions to model related variables to detect anomalies. Using the LSTM network as the base models to capture the temporal correlations of time-series data, MAD-GAN [6] proposed an unsupervised anomaly detection method combining GANs by considering complex dependencies amongst different time-series variables. Sakurada and Yairi [36] conducted dimensionality reduction based on autoencoders for anomaly detection. The ODCA framework [37] included three parts: 1) data preprocessing; 2) outlier analysis; and 3) outlier rank, which used

cross correlation to translate high-dimensional data sets to 1-D cross-correlation function. OmniAnomaly [13] was a stochastic model to avoid potential misguiding by uncertain instances, which used stochastic variable connection and normalizing flow to get reconstruction probabilities and adopted the streaming POT with drift (DSPOT) algorithm [38] for automatic threshold selection. Senin *et al.* [39] proposed two algorithms that conducted symbolic time-series discretization and used grammar reduction to compress the input sequence and compactly encode them with grammar rules. Those rarely used substrings in the grammar rules were regarded as anomalies. Autoregressive with exogenous inputs (ARX) and artificial neural network (ANN) [40] extracted time-series features and detected anomalous data points by conducting hypothesis testing on the extrema of residuals.

To cover the shortage that the convolution and pooling operators of CNNs are defined for regular grids, recent GNN [41] generalizes CNNs to graphs that are able to encode irregular and non-Euclidean structures. GNN adopted the localized spectral filters and used a graph coarsening algorithm to cluster similar vertices for speeding up. In this way, GNN efficiently extracted the local stationary property and captured the correlation between nodes. In real-world IoT environment, the graph structure modeling the correlations between sensors is often not predefined in advance. The graph deviation network (GDN) [4] learned the pairwise relationship by cosine similarity to elaborate the adjacent matrix which can be modeled as a graph. Then, it predicted the future values by graph attention-based forecasting and computed the absolute error value to evaluate the graph deviation score. MTAD-GAN [17] concatenated the feature-oriented and time-oriented graph attention layer to learn the graph structure and used both forecasting-based model and reconstruction-based model to calculate integrated loss. Then automatic threshold algorithm was adopted to perform anomaly detection.

III. Problem Statement

In this work, we focus on the task of multivariate time-series anomaly detection. Let $\mathcal{X}^{(t)} \in \mathbb{R}^M$ denote the original multivariate time-series data at any timestamp t, where M is the total number of sensors or any data measuring node within the same entity. M is also reported as the number of features or variables in some literature [4], [6], [13]. Considering the high unbalance between normal data and anomalies, we only construct the sequence modeling process on normal data (without anomalies) and make prediction on testing data (with anomalies) for anomaly detection. Specifically, we let $\mathcal{X}$ and $\hat{\mathcal{X}}$ represent the entire normal data and data with anomalies, respectively. For sequence modeling on normal data, we inherit a forecasting-based strategy to predict the time-series value $\mathbf{x}^{(t)} \in \mathbb{R}^M$ at next time step t (also known as single-step time-series forecasting) based on the historical data $\mathbf{x} = \{\mathbf{x}^{(t-n)}, \ldots, \mathbf{x}^{(t-1)}\}$ with a specific window size n. Therefore, given a sequence of historical n time steps of multivariate contiguous observations $\hat{\mathbf{x}} \in \mathbb{R}^{M \times n}$, the goal of anomaly detection is to predict the output vector $\hat{\mathbf{y}} \in \mathbb{R}^n$, where $\hat{\mathbf{y}}^{(t)} \in \{0, 1\}$ denotes binary labels indicating whether there is an anomaly at time tick t. Precisely, our proposed approach returns an anomaly score for each testing timestamp, and then the anomaly result can be obtained via selecting different thresholds.

We also provide some basic graph-related concepts for better understanding formulated as follows.

Definition 1 (Graph): A directed graph is formulated as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \ldots, M\}$ is the set of nodes, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges, where $\mathbf{e}_{i,j}$ represents the unidirected edge flowing from node i to node j.

Definition 2 (Node Neighborhood): Let $i \in \mathcal{V}$ denote a node and $\mathbf{e}_{i,j} \in \mathcal{E}$ denote the edge pointing from node i to node j. The neighborhood of any node i is defined as $\mathcal{N}(i) = \{j \in \mathcal{V} | \mathbf{e}_{i,j} \in \mathcal{E}\}$.

IV. Methodology

In most real-life scenarios of IoT, there are usually complex topological relationships between sensors where the entire entity can be seen as a graph structure. Each sensor is also viewed as a specific node in the graph. Previous methods [4], [25] focused on applying various distance metrics to measure the relations between nodes mostly by selecting the top-K closest ones as their neighbor dependencies. Different from the existing approaches, we devise a *directed graph structure LP* (see Fig. 1) to automatically learn the adjacency matrix among nodes such that the network can achieve the maximum benefits. The core of the LP is named the *Gumbel-Softmax Sampling* strategy [42], [43] inspired by the policy learning network in many reinforcement learning methods [44], [45]. These discovered hidden associations are then fed into the graph convolution layers for IP modeling. We then integrate these graph convolution layers with different level-dilated convolution layers together to construct a *hierarchical context encoding block*, specifically for temporal data. The outputs of the context encoding block are then applied positional encoding as the inputs of *Transformer* [27] for single-step time-series forecasting. We also propose a global attention strategy to overcome the quadratic computation complexity challenge of the multihead attention mechanism. Fig. 1 further illustrates the entire architecture in detail.

A. Gumbel-Softmax Sampling

The sampling process of discrete data from a categorical distribution is originally nondifferentiable, where typical backpropagation in deep neural networks cannot be conducted. Maddison *et al.* [42] and Jang *et al.* [43] proposed a differentiable substitution of discrete random variables in stochastic computations by introducing the Gumbel-Softmax distribution, a continuous distribution over the simplex that can approximate samples from a categorical distribution. In our graph LP with a total number of M candidate nodes, we let $\mathbf{z}^{i,j}$ be a binary connection control variable for any pair of nodes i and j with unidirected probabilities from node i to node j as $\{\pi_0^{i,j}, \pi_1^{i,j}\}$, where $\pi_0^{i,j} + \pi_1^{i,j} = 1$ and $\pi_1^{i,j}$ represents the probability that there exists an information flow from node i to node j in the graph (see Fig. 2). Similarly, by the Gumbel-Max

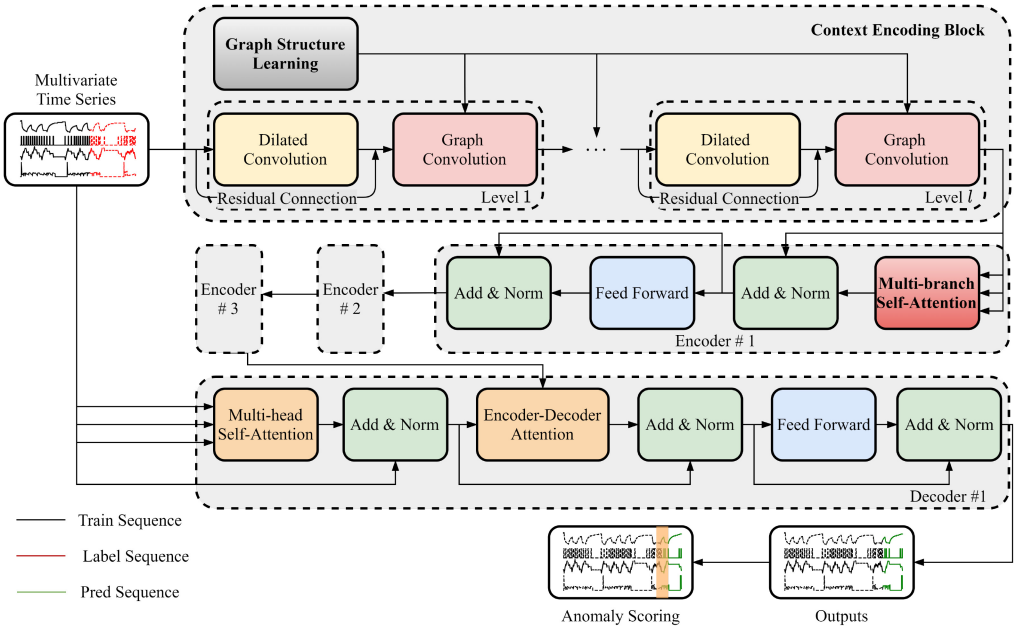

Fig. 1. Visualization of our proposed GTA's architecture with l levels dilated convolution and graph convolution, three encoder layers, and one decoder layer. Generally, the input multivariate time-series inputs are split into train sequences and label sequences, of which train sequences are fed into encoder while label sequences are fed to the decoder.

trick, we can sample any pair of nodes' connection strategy $z^{i,j} \in \{0, 1\}^2$ with

$$z^{i,j} = \underset{c\in\{0,1\}}{\arg\max}\left(\log \pi_c^{i,j} + g_c^{i,j}\right) \tag{1}$$

where g_0, g_1 are independent and identically distributed (i.i.d) samples drawn from a standard Gumbel distribution which can be easily sampled using inverse transform sampling by drawing $u \sim \text{Uniform}(0, 1)$ and computing $g = -\log(-\log u)$. We further substitute this arg max operation, since it is not differentiable, with a Softmax reparameterization trick, also known as the Gumbel-Softmax trick, as

$$z_c^{i,j} = \frac{\exp\left(\left(\log \pi_c^{i,j} + g_c^{i,j}\right)/\tau\right)}{\sum_{v\in\{0,1\}} \exp\left(\left(\log \pi_v^{i,j} + g_v^{i,j}\right)/\tau\right)} \tag{2}$$

where $c \in \{0, 1\}$ and τ is the temperature parameter to control Gumbel-Softmax distribution's smoothness, as the temperature τ approaches 0, the Gumbel-Softmax distribution becomes identical to the one-hot categorical distribution. As the randomness of g is independent of π, we can now directly optimize our gating control policy using standard gradient descent algorithms.

Compared to the previous graph structure learning approaches, our proposed method significantly reduces the computation complexity from $\mathcal{O}(M^2)$ to $\mathcal{O}(1)$ since it requires no dot products among high-dimensional node embeddings. Additionally, the graph structure LP is able to automatically learn the global topological connections among all nodes, thereby avoiding the limitation of selecting only the top-K *nearest* nodes as neighbors.

B. Influence Propagation via Graph Convolution

On the top of the learned topological structure, the graph convolution block aims to further model the IP process and update each specific node's representation by incorporating its neighbors' information. Considering the characteristics of tasks such as anomaly detection, usually, the occurrence of abnormalities is due to a series of chain influences caused by one or several nodes being attacked. Therefore, it is intuitive for us to model the relationships between upstream and downstream nodes by capturing both temporal and spatial differences. Thus, we define our IP convolution process concerning each specific node and its neighborhoods by applying a nodewise symmetric aggregating operation $\square$ (e.g., add, mean, or max) on the differences between nodes associated with all the edges emanating from each node. The updated output of IPConv at the ith node is given by

$$\mathbf{x}_i' = \sum_{j\in\mathcal{N}(i)} h_{\mathbf{\Theta}}\left(\mathbf{x}_i \| \mathbf{x}_j - \mathbf{x}_j \| \mathbf{x}_j + \mathbf{x}_i\right) \tag{3}$$

where $\square$ is chosen as summation in our method, $h_{\mathbf{\Theta}}$ denotes a neural network, i.e., multilayer perceptrons (MLPs), $\mathbf{x}_i \in \mathbb{R}^T$ represents the time-series embedding of node i and $||$ denotes the concatenation operation. We denote $\mathbf{x}_j - \mathbf{x}_i$ as the differences between nodes to *explicitly model the IP delay* from node j to i, captured by the value difference at each timestamp of the time-series embedding. We also incorporate the term $\mathbf{x}_i + \mathbf{x}_j$ with the differences to work as a *scale benchmark* such that the model can learn the truly generalized impact to the other nodes brought by anomalies instead of extreme values. Intuitively, for any specific node i, if one of its neighbor nodes j being attacked, node i shall be severely affected sooner or later due to the restricted topological relationship.

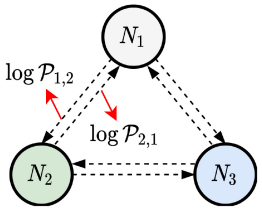

Fig. 2. Suppose we have three sensors (N_1, N_2, N_3) of which the dependencies are yet hidden. Our connection LP's main idea is to use the Gumbel-Softmax Sampling strategy to sample a random categorical vector for determining whether any directed connection between two nodes can be established. For N_1 and N_2, if the value of $\mathcal{P}_{1,2}$ is relatively high, it represents N_1 is highly possibly pointed to N_2, *vice versa*.

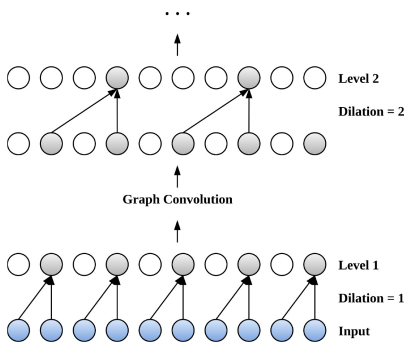

Fig. 3. Visualization of hierarchical dilated convolution combined with graph convolution.

Training Strategy and Regularization: Graph convolution based on the learned dependencies among sensors only aggregates the information from nodes' neighbors without taking efficiency into account. Under the extreme circumstance that all nodes are mutually connected, aggregating neighborhood information adds considerable noise to each node. However, it is preferred to form a compact subgraph structure for every single node, in which redundant connections are omitted as much as possible without deteriorating the forecasting accuracy. To this end, we propose a sparsity regularization $\mathcal{L}_s$ to enhance the compactness of each node by minimizing the log-likelihood of the probability of a connection being established as

$$\mathcal{L}_s = \sum_{1 \le i,j \le M, i \ne j} \log \pi_1^{i,j}. \tag{4}$$

Furthermore, to encourage better learning convergence, the *connection LP* is initialized with all nodes connected. We *warm up* the network weights by training with this *complete graph structure* for a few epochs to provide a good starting point for the policy learning.

C. Hierarchical Dilated Convolution

The dilated convolution [46] is widely used in sequence modeling due to its powerful capability in extracting high-level temporal context features by capturing sequential patterns of time-series data through standard 1-D convolution filters. Setting different dilation size levels can discover temporal patterns with various ranges and handle very long sequences. However, choosing the right kernel size is often a challenging problem for convolutional operations. Some previous approaches adopted the widely employed inception learning strategy [47] in computer vision which concatenates the outputs of convolutional filters with different kernel sizes followed by a weighted matrix. Unlike them, we propose a *hierarchical dilated convolution learning strategy* combined with the graph convolution above to fully explore the temporal context modeling process with different sequence lengths and receptive fields by setting multiscale dilation sizes. Specifically, as Fig. 3 illustrates, the bottom layer represents the multivariate time-series input (for some time t, onto which repeated dilated convolutions, with increasing dilation rates, are applied; the filter width is again set to equal two in the observed model). The first-level block applies dilated convolutions with the dilation rate equal to 1, meaning that the layer applies the filter onto two adjacent elements, $\mathbf{x}^{(t)}$ and $\mathbf{x}^{(t+1)}$, of the input series. The outputs of the first-level dilated convolutions are fed into the graph convolution module proposed above. Then, the second-level layer applies dilated convolutions, with the rate now set to equal 2, which means that the filter is applied onto elements $\mathbf{x}^{(t)}$ and $\mathbf{x}^{(t+2)}$ (note here that the number of parameters remains the same, but the filter width has been "widened"). By setting multiscale dilation sizes with a hierarchical learning style, abundant temporal representations concerning different temporal positions and sequence lengths can be effectively learned.

The hierarchical dilated convolution and the graph convolution together form the temporal context embedding progression, where dilated convolution captures the long-term temporal dependencies while graph convolution describes the topological connection relationships between sensors (or nodes). As a result, the final outputs have been well represented to be the inputs of the next forecasting procedure using the Transformer architecture.

D. More Efficient Multibranch Transformer

Transformer [27] has been widely used in sequence modeling due to the superior capability of the multihead attention mechanism in long-distance dependencies capturing. However, one main efficiency bottleneck in self-attention is that the pairwise token interaction dot-production incurs a complexity of $\mathcal{O}(n^2)$ with respect to the sequence length. To tackle this challenge, in this section, we first briefly review the background of some recent development of the multihead attention mechanism and then propose a more efficient Transformer architecture based on the innovative multibranch attention mechanism which is more computationally efficient.

Self-Attention in Transformers: The vanilla multihead self-attention mechanism was originally proposed by Vaswani *et al.* [27]. For a sequence of token representations $\mathbf{X} \in \mathcal{R}^{n \times d}$ (with sequence length n and dimensionality d), the self-attention function first projects them into queries $\mathbf{Q} \in \mathcal{R}^{n \times d_k}$, keys $\mathbf{K} \in \mathcal{R}^{n \times d_k}$, and values $\mathbf{V} \in \mathcal{R}^{n \times d_v}$, h times with different, learned linear projections to d_k, d_k, and d_v dimensions, respectively. Then, a particular scaled dot-product attention was computed to obtain the weights on

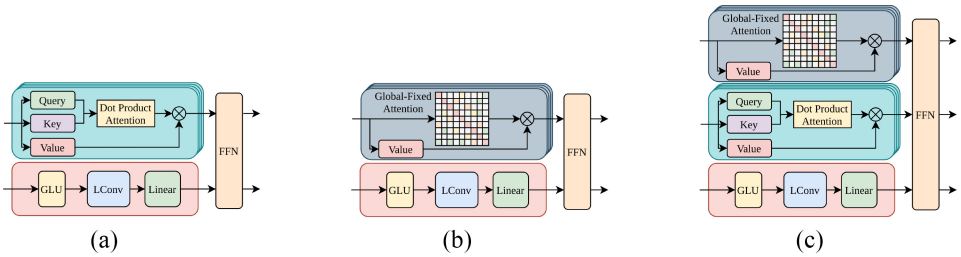

Fig. 4. Different variants of efficient multibranch attention mechanism. Right: Replacing vanilla multihead attention with a combination of both global-fixed attention and vanilla multihead attention and neighborhood convolution by splitting embeddings into multiple channels. (a) Vanilla multibranch transformer. (b) Multibranch transformer with global-fixed attention. (c) Our proposed multibranch attention mechanism.

the values as

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}. \tag{5}$$

Multihead attention allows the model to jointly attend to information from different representation subspaces at different positions. With a concatenated computing way, the final output of multihead attention is as follows:

$$\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O \tag{6}$$

in which h is the number of total heads. Each head is defined as

$$head_i = \text{Attention}\left(\mathbf{Q}W_i^Q, \mathbf{K}W_i^K, \mathbf{V}W_i^V\right) \tag{7}$$

where the projections are parameter matrices $W_i^Q \in \mathcal{R}^{d\times d_k}$, $W_i^K \in \mathcal{R}^{d\times d_k}$, $W_i^V \in \mathcal{R}^{d\times d_v}$, and $W^O \in \mathcal{R}^{hd_v\times d}$.

Global-Learned Attention: Recent research [48], [49] claim that the self-attention in Transformers can be substantially simplified with trivial attentive patterns at training time: only preserving adjacent and previous tokens is necessary. The adjacent positional information, such as "current token," "previous token," and "next token" are the key features learned across all layers by encoder self-attention. Instead of costly learning the trivial pattern using massive corpus with considerable computational resources, the conventional pairwise token interaction attention could be replaced by a more computation-efficient global attention pattern. In practice, manually predefine all global-fixed patterns is easy to implement but can barely cover all possible situations. To generalize the global-fixed attention pattern proposed in [50], we apply a parameter matrix $\mathbf{S} \in \mathcal{R}^{m\times m}$ $(m > n)$ as a learnable global compatibility function across all training samples following the *Synthesizer* [51]. Hence, each head adds m^2 parameters while reducing two projection matrices W^Q and W^K. The attention now has been as follows:

$$\text{Attention}(\mathbf{S}, \mathbf{V}) = \text{Softmax}(\mathbf{S})\mathbf{V} \tag{8}$$

where $\mathbf{S}$ is a learnable matrix which can be randomly initialized.

Multibranch Architecture for Transformers: Wu *et al.* [52] has demonstrated the effectiveness of multibranch attention in capturing global and local context patterns, especially under mobile computational constraints. As Fig. 4 illustrates, this double-branch architecture splits the original input sequences into two pieces along the embedding channel, followed by two attention branches: one convolution branch for extracting information in a restricted neighborhood and one multihead attention branch for capturing long-distance dependencies. As a substitute for vanilla self-attention, we apply a task-specific alignment matrix that learns globally across all training samples where attention weights are no longer conditioned on any input token in our architecture. By simply replacing the dot-production with global-learned alignment as Fig. 4 shows, the input sequences will only be projected into *value* matrices. A weighted sum up of *values* is then calculated using this global-learned attention. In order to explore a better tradeoff between computation efficiency and model performance, we propose to combine the pairwise token interactions and global-learned attention in terms of a branchwise mixing strategy.

Branchwise Mixing: For branchwise mixing, the input sequences are split into multiple branches along the embedding dimension as Fig. 4 clearly describes. Different from the original two-branch architecture, we build one more branch for global-learned attention. Thus

$$\begin{aligned}\text{Attention} &= \text{Concat}\left(\mathbf{A}^{(1)}, \mathbf{A}^{(2)}\right)\\ \mathbf{A}^{(1)} &= \text{MultiHead}\left(\mathbf{X}^{(1)}\right)\\ \mathbf{A}^{(2)} &= \text{Global}\left(\mathbf{X}^{(2)}\right)\end{aligned} \tag{9}$$

where $\mathbf{X}^{(1)} \in \mathcal{R}^{n\times d_1}$, $\mathbf{X}^{(2)} \in \mathcal{R}^{n\times d_2}$, and $d = d_1 + d_2$.

In our models, we only change the branch that captures the global contexts while remaining the local pattern extractor using either *lightConv* or *dynamicConv* [53].

Computation Analysis: Table I lists the different model variants explored within our proposed framework. The column $|\theta|$ refers to the total number of parameters in one self-attention module excluding the feedforward layer. Obviously, compared to the original scaled dot-production, the amount of computation of global-learned attention is directly reduced by half in terms of Mult-Adds. Our proposed multibranch mixing strategy increases the amount of calculation in varying degrees due to the mix with scaled dot-production. However, this is a tradeoff between computation complexity and model size. More precisely, when $m \leq \sqrt{2/h}d$, the global attention module is more computationally efficient than the other variants.

E. Anomaly Scoring

Inspired by [54], the original multivariate time-series inputs are split into two parts: 1) training sequences for the encoder and 2) label sequences for the decoder. The decoder receives

TABLE I
MEMORY AND COMPUTATION ANALYSIS ON DIFFERENT ATTENTION TYPES

Attention Type	$\|\theta\|$	# Mult-Adds	Global/Inter
Scaled Dot-Product	$4d^2$	$\mathcal{O}(4nd^2 + 2n^2d)$	Inter
Global-Learned	$m^2h + 2d^2$	$\mathcal{O}(2nd^2 + n^2d)$	Global
Branch-Wise Mixing	$4d_1^2 + m^2h + 2d_2^2$	$\mathcal{O}(4nd_1^2 + n^2d_1 + 2nd_2^2 + n^2d)$	Both

long sequence inputs, pads the target elements into zero, measures the weighted attention composition of the feature map, and instantly predicts output elements in a generative style. Let the single-step prediction denote as $\hat{\mathcal{Y}} \in \mathbb{R}^{M\times n}$. We apply the mean square error (MSE) between the predicted outputs $\hat{\mathbf{Y}}$ and observation $\mathcal{Y}$, as the loss function to minimize

$$\mathcal{L}_{\text{mse}} = \frac{1}{M}\sum_{t=1}^{n}\left\|\mathcal{Y}^{(t)} - \hat{\mathcal{Y}}^{(t)}\right\|_2^2. \tag{10}$$

Similar to the loss objective, the anomalous score compares the expected value at time t to the observed value, computing an anomaly score via the deviation level as

$$\hat{\mathbf{y}}^{(t)} = \sum_{i=1}^{M}\left\|\mathcal{Y}_i^{(t)} - \hat{\mathcal{Y}}_i^{(t)}\right\|_2^2. \tag{11}$$

Finally, we label a timestamp t as an anomaly if $\hat{\mathbf{y}}^{(t)}$ exceeds a fixed threshold. Since different approaches could be employed to set the threshold such as the extreme value theory [38], the same anomaly detection model could result in different prediction performance with different anomaly thresholds. Thus, we apply a grid search on all possible anomaly thresholds to search for the best F1-score (with notation **) and Recall (with notation *) in theory and report them.

V. EXPERIMENTS

A. Data Sets

We evaluate our method over a wide range of real-world anomaly detection data sets. **SWaT** [55], the Secure Water Treatment data set is collected from a water treatment testbed for cyberattack investigation initially launched in May 2015. The SWaT data set collection process lasted for 11 days, with the system operated 24 h per day such that the network traffic and all the values obtained from all 51 sensors and actuators are recorded. Due to the system working flow characteristics, there is a natural topological structure relationship between all sensing nodes. After this, a total of 41 attacks derived through an attack model considering the intent space of a CPS were launched during the last four days of the 2016 SWaT data collection process. As such, the overall sequential data is labeled according to normal and abnormal behaviors at each timestamp. **WADI** [9] Water Distribution data set is collected from a water distribution testbed as an extension of the SWaT testbed. It consists of a total of 16 days of continuous operations with 14 days under regular operation and two days with attack scenarios. The entire testbed contains 123 sensors and actuators. Moreover, soil moisture active passive satellite (SMAP) and Mars Science Laboratory rover (MSL) are two public data sets published by NASA [56]. Each data set has a training and a testing subset, and anomalies in both testing subsets have been labeled [8].

TABLE II
STATISTICAL SUMMARY OF DATA SETS **SWaT** AND **WADI**

Datasets	SWaT	WADI
Feature Desc.	All sensors and actuators.	
# Features	51	112
# Attacks	41	15
Attack durations (mins)	$2 \sim 25$	$1.5 \sim 30$
Training size (normal data)	49619	120899
Testing size (data with attacks)	44931	17219
Anomaly rate (%)	12.14	5.75

TABLE III
STATISTICAL SUMMARY OF DATA SETS **SMAP** AND **MSL**

Datasets	SMAP	MSL
Feature Desc.	Radiation, temperature, power, etc.	
# Features	25	25
Training size (normal data)	135183	58317
Testing size (data with anomalies)	427617	73729
Anomaly rate (%)	13.13	10.72

Tables II and III summarize the statistics of the four data sets. In order to fair comparison with other methods, the original data samples for **SWaT** and **WADI** are downsampled to one measurement every 10 s by taking the median values following [4].

B. Experimental Setup

1) Data Preprocessing: We perform a data standardization before training to improve the robustness of our model. Data preprocessing is applied on both training and testing set

$$\tilde{x} = \frac{x - \min X_{\text{train}}}{\max X_{\text{train}} - \min X_{\text{train}}} \tag{12}$$

where $\max(X_{\text{train}})$ and $\min(X_{\text{train}})$ are the maximum value and the minimum value of the training set, respectively.

2) Evaluation Metrics: We adopt the standard evaluation metrics in anomaly detection tasks, namely, Precision, Recall and F1 score, to evaluate the performance of our approach, in which

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{13}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{14}$$

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{15}$$

TABLE IV
EXPERIMENTAL RESULTS ON **SWaT** AND **WADI**

Datasets	Methods	Precision(%)	Recall(%)	F1-score
SWaT	PCA	24.92	21.63	0.23
	KNN	7.83	7.83	0.08
	FB	10.17	10.17	0.10
	AE	72.63	52.63	0.61
	DAGMM	27.46	69.52	0.39
	LSTM-VAE	96.24	59.91	0.74
	MAD-GAN	98.97	63.74	0.77
	GDN	**99.35**	68.12	0.81
	GTA* (ours)	74.91	**96.41**	0.84
	GTA**	94.83	88.10	**0.91**
	$\Delta_{\uparrow}$ (best F1)	-4.55%	**+29.33%**	**+12.35%**
WADI	PCA	39.53	5.63	0.10
	KNN	7.76	7.75	0.08
	FB	8.60	8.60	0.09
	AE	34.35	34.35	0.34
	DAGMM	54.44	26.99	0.36
	LSTM-VAE	87.79	14.45	0.25
	MAD-GAN	41.44	33.92	0.37
	GDN	**97.50**	40.19	0.57
	GTA* (ours)	74.56	**90.50**	0.82
	GTA**	83.91	83.61	**0.84**
	$\Delta_{\uparrow}$ (best F1)	-13.94%	**+108.04%**	**+47.37%**

Best performance in bold. Second-best with underlines.
* represents the results chosen by best Recall.
** represents the results chosen by best F1-score.
$\Delta_{\uparrow}$ represents the percentage increase between our best F1-score performance and the second-best method (GDN).

where TP represents the truly detected anomalies (also known as true positives), FP stands for the falsely detected anomalies (also known as false positives), TN represents the correctly classified normal samples (also known as true negatives), and FN is the misclassified normal samples (also known as false negatives). Given the fact that in many real-world anomaly detection scenarios, it is more vital for the system to detect all the real attacks or anomalies by tolerating a few false alarms. As such, we generally give more concern to Recall and the overall F1 score instead of Precision. Considering different anomaly score thresholds may result in different metric scores, we hence report both our best Recall and F1 results (with notations * and **, respectively) on all data sets for a thorough comparison.

Also, we adopt the point-adjust way to calculate the performance metrics following [13]. In practice, anomalous observations usually occur consecutively to form contiguous anomaly segments. An alert for anomalies can be triggered within any subset of an actual anomaly window. Thus, for any observation in the ground-truth anomaly segment, if it is detected as an anomaly or attack, we would consider this whole anomaly window is correctly detected and every observation point in this segment has been classified as anomalies. The observations outside the ground-truth anomaly segment are treated as usual. In all, we first train our model on the training set to learn the general sequence pattern and make the forecasting on the test set for anomaly detection.

3) Baselines: We compare our GTA with a wide range of state of the arts in multivariate time-series anomaly detection, including: 1) *reconstruction-based models:* PCA, AE [10], KitNet [57], DAGMM [12], GAN-Li [20], OmniAnomaly [13], LSTM-VAE [11], and MAD-GAN [6] and 2) *forecasting-based models:* KNN [14], FB [15], MTAD-GAT [17], and GDN [4].

4) Training Settings: We implement our method and all its variants using Pytorch[1] version 1.7.0 with CUDA 10.1 and Pytorch Geometric Library [58] version 1.6.3. We conduct all experiments on four NVIDIA Tesla P100 GPUs. For time-series forecasting, we set the historical window size to 60 frames with a label series length as 30 to predict the value at next timestamp. The number of dilated convolution levels for temporal context modeling is set to 3. Also, the general model input embedding dimension is set to 128. For the conventional multihead attention mechanism, the number of heads is set to 8. In total, we have three encoder layers and two decoder layers and the dimensional of fully connected network is set to 128 which is equal to the model dimension. Additionally, we apply the dropout strategy to prevent overfitting with the dropout rate consistently equals 0.05. The models are trained using the Adam optimizer with learning rate initialized as $1e^{-4}$ and β_1 and β_2 as 0.9 and 0.99, respectively. A learning rate adjusting strategy is also applied. We train our models for up to 50 epochs and early stopping strategy is applied with patience of 10. We run each experiment for five trials and report the mean value.

C. Experimental Results

In Table IV, we show the anomaly detection accuracy in terms of precision, recall, and F1-score, of our proposed GTA method and other state-of-the-arts on data sets **SWaT** and **WADI**. Each of these baselines provides a specific threshold selection method, and the reported F1-score is calculated correspondingly. Our proposed GTA significantly outperforms all the other approaches on both data sets by achieving the best F1-score as 0.91 for **SWaT** and 0.84 for **WADI**. Astonishingly, compared to the second-best model GDN, GTA can achieve an overall 12.35% increase and an impressive 47.47% improvement in terms of the best F1-score on these two data sets, respectively. Moreover, we have the following observations.

1) Compared to the conventional unsupervised approaches, such as PCA, KNN, and FB, deep learning-based techniques (AE, LSTM-VAE, MAD-GAN, etc.) generally have a better detection performance on both data sets. By adopting the recurrent mechanism (RNN, GRU, and LSTM) in modeling long sequences and capturing the temporal context dependencies, the deep learning-based methods demonstrate superiority over the conventional methods.
2) DAGMM [12] aims to handle multivariate data *without* temporal information, indicating the input data contains only one observation instead of a historical time-series window. Hence, this approach is not suitable for temporal dependency modeling, which is crucial for multivariate time-series anomaly detection.

[1]https://pytorch.org/

TABLE V
EXPERIMENTAL RESULTS ON **SMAP** AND **MSL**

Method		SMAP			MSL		
		Precision(%)	Recall(%)	F1-score	Precision(%)	Recall(%)	F1-score
R-Models	KitNet	77.25	83.27	0.8014	63.12	79.36	0.7031
	GAN-Li	67.10	87.06	0.7579	71.02	87.06	0.7823
	LSTM-VAE	85.51	63.66	0.7298	52.57	95.46	0.6780
	MAD-GAN	80.49	82.14	0.8131	85.17	89.91	0.8747
	OmniAnomaly	74.16	**97.76**	0.8434	88.67	91.17	0.8989
F-Models	LSTM-NDT	**89.65**	88.46	0.8905	59.44	53.74	0.5640
	DAGMM	58.45	90.58	0.7105	54.12	**99.34**	0.7007
	MTAD-GAT	89.06	91.23	0.9013	87.54	94.40	0.9084
	GTA** (ours)	89.11	91.76	**0.9041**	**91.04**	91.17	**0.9111**

Best performance in bold. Second-best with underlines.
** represents the results chosen by best F1-score.

3) Most existing methods are based on recurrent neural networks to capture temporal dependency, including both reconstruction-based models (LSTM-VAE, OmniAnomaly, and MAD-GAN) and forecasting-based models (LSTM-NDT and MTAD-GAT). Of which, LSTM-NDT [8] is a deterministic model without leveraging stochastic information for modeling the inherent stochasticity of time series. LSTM-VAE [11] combines LSTM with VAE for sequence modeling; however, it ignores the temporal dependencies among latent variables. OmniAnomaly [13] was then proposed to solve this problem. Additionally, MAD-GAN [6] aims to adopt a general adversarial training fashion to reconstruct the original time series, which also uses recurrent neural networks. Nevertheless, the recurrent learning mechanism's core properties restrict the modeling process to be sequential. Past information has to be retained through the past hidden states, limiting the long-term sequence modeling capability of the model. The Transformer adopts a nonsequential learning fashion, and the powerful self-attention mechanism makes the context distance between any token of a time series shrink to one, which is of high importance to sequence modeling as more historical data can provide more pattern information.
4) Though GDN [4] is also a graph learning-based anomaly detection approach, it adopts the top-K *nearest* connection strategy to model the topological graph structure among sensors, which have certain limitations as we discussed in Section I. MTAD-GAT [17] directly utilizes the initial graph structure information by assuming all sensors are mutually connected, making it a complete graph that is not suitable for many real-life situations.

From Table V, we can see that the overall improvements in terms of best F1-score on data sets **SMAP** and **MSL** are not as impressive as Table IV shows. We argue the main difference of results between the NASA anomaly data sets and the Cyberattack data sets lies in the features' dependencies. SMAP provides measurements of the land surface soil moisture by measuring various separated attributes, such as radiation, temperature, computational activities, etc. Though these attributes are not entirely independent of each other, the internal relationships between them are much weaker than those within **SWaT** or **WADI** where any slight change that appears on one sensor can propagate to the whole network. Therefore, our proposed graph structure learning strategy might be more effective on data sets with a strong topological structure.

TABLE VI
ANOMALY DETECTION ACCURACY IN TERMS OF PRECISION(%), RECALL(%), AND F1-SCORE OF GTA AND ITS VARIANTS

Method	SWaT			WADI		
	Prec(%)	Rec(%)	F1-score	Prec(%)	Rec(%)	F1-score
GTA	**94.83**	**88.10**	**0.91**	**83.91**	**83.61**	**0.84**
w/o Graph	88.64	65.73	0.75	71.25	68.23	0.70
w/o LP	89.36	72.12	0.80	79.56	77.10	0.78
w/o Attn	78.75	65.34	0.71	74.75	70.90	0.73

D. Ablation Studies

To study each component of our approach's effectiveness, we gradually exclude the elements to observe how the model performance degrades on data sets **SWaT** and **WADI**. First, we study the significance of modeling the dependencies among sensors using graph learning. We directly apply the original time series as the inputs for the Transformer and make the forecasting without graph learned phase. Second, we study the significance of our proposed structure LP by substituting it with a static complete graph where every node is bidirectionally linked to each other. Finally, to study the necessity of the Transformer-based architecture for sequence modeling, we substitute the Transformer with a GRU-based recurrent neural network for forecasting. The results are summarized in Table VI and provide the following observations: 1) our proposed LP helps the graph convolution operation by capturing only proper information flow with noises filtered out; 2) there is a considerable gap between GTA and the variant without graph learning, without graph learning which again demonstrates the importance of topological structure modeling in handling multivariate time-series anomaly detection; and 3) transformer-based architecture exhibits superiority in sequence modeling, where the self-attention mechanism

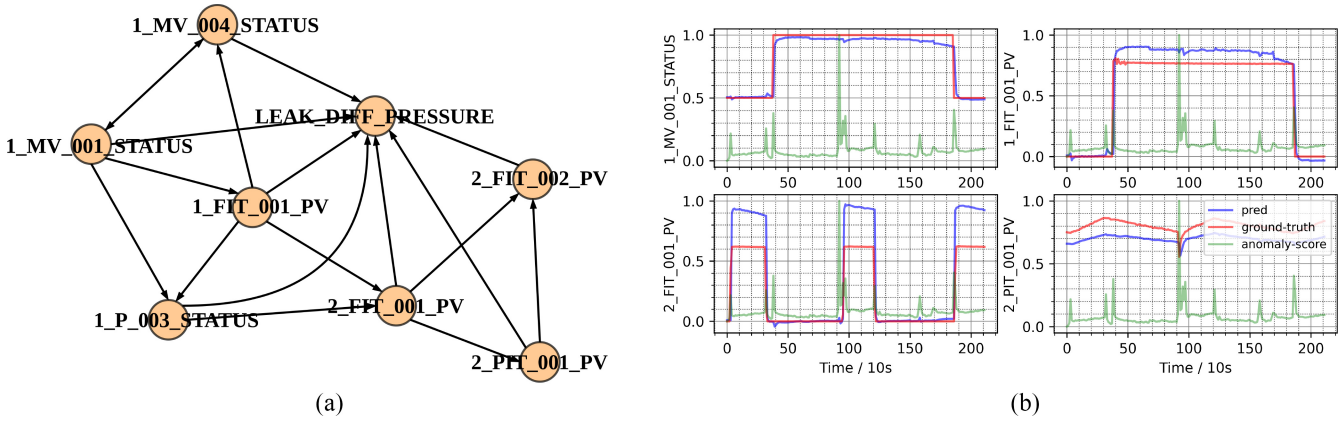

Fig. 5. Case study of showing an attack in **WADI**. (a) Partial graph structure learned by the LP. (b) Attacked sensor with three other malicious sensors.

plays a critical role. Moreover, these results again confirm that every component of our method is indispensable and make this framework powerful in multivariate time-series anomaly detection.

E. Graph Learning and Case Study

By introducing a case study of an actual attack from the Cyberattack data set **WADI**, we evaluate what a graph structure would the graph LP learn and how this helps us localize and comprehend an anomaly in this section. An assault with a period of 25.16 min was logged in the WADI data collecting log, which fraudulently turned on the motorized valve 1_MV_001_STATUS and caused an overflow on the primary tank. It is difficult for the operation engineers to find out the status of this valve manually because it is still within normal range. As a result, it is not easy to spot this oddity.

The water distribution treatment, for example, consists of three-state processes from the water supply, distribution network, and return water system, which are denoted as P1, P2, and P3, respectively. Every sensor and actuator, in every condition, is inextricably linked. The raw water inlet valve that regulates the SUTD entering, for example, is represented by 1_MV_001_STATUS. Because 1_FIT_001_PV is a downstream flow indicator transmitter of the water distribution, the value of 1_FIT_001_PV rises rapidly if the 1_MV_001 valve is turned on abruptly. As its outcome of the first stage propagates the influence from the raw water transfer pump to the second stage, 2_FIT_001_PV is also vulnerable to the same malicious attack. In addition, as LEAK_DIFF_PRESSURE becomes irregular during this procedure, the leaking water pressure grows without a doubt. Our graph LP learned a partial graph in Fig. 5(a), which almost properly depicts the topological interactions among sensors. The LEAD_DIFF_PRESSURE is almost related to every other displayed node as malicious information passes from upstream sensors to downstream ones. More importantly, Fig. 5(b) shows our model's predicted sensor curves (blue lines) against the ground truth (red lines) of sensors 1_FIT_001_PV, 2_FIT_001_PV, and 2_PIT_001_PV within the attack duration. The predictions of these sensors are consistently higher than the ground truth, where the anomaly score increases correspondingly. It is mainly because the input time series has been embedded with the graph structure information through the IP convolution operation. Sensors that are not directly attacked will still be severely affected if sensors that are highly related to them are attacked. Therefore, our model can capture this dependency and result in an *abnormal* prediction, which is vital for the following anomaly detection.

VI. Conclusion

In this work, we proposed GTA, a Transformer-based framework for anomaly detection that uses the introduced connection LP to automatically learn sensor dependencies. To simulate the information flow among the sensors in the graph, we devised an unique IP graph convolution. The inference speed of our proposed multibranch attention technique is greatly improved without sacrificing model performance. Extensive experiments on four real-world data sets demonstrated that our strategy outperformed other state-of-the-art approaches in terms of prediction accuracy. We also provided a case study to demonstrate how our approach identifies the anomaly by utilizing our proposed techniques. We aim to explore more about combining this approach with the online learning strategy to land it on the mobile IoT scenarios for future work.

References

[1] M. S. Mahdavinejad, M. Rezvan, M. Barekatain, P. Adibi, P. M. Barnaghi, and A. P. Sheth, "Machine learning for Internet of Things data analysis: A survey," 2018. [Online]. Available: https://arxiv.org/abs/1802.06305.

[2] Z. Cai and Z. He, "Trading private range counting over big IoT data," in *Proc. 39th IEEE Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Dallas, TX, USA, Jul. 2019, pp. 144–153. [Online]. Available: https://doi.org/10.1109/ICDCS.2019.00023

[3] M. Mohammadi, A. I. Al-Fuqaha, S. Sorour, and M. Guizani, "Deep learning for IoT big data and streaming analytics: A survey," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2923–2960, 4th Quart., 2018.

[4] A. Deng and B. Hooi, "Graph neural network-based anomaly detection in multivariate time series," in *Proc. 35th AAAI Conf. Artif. Intell.*, 2021, pp. 4027–4035.

[5] Z. Cai and X. Zheng, "A private and efficient mechanism for data uploading in smart cyber-physical systems," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 2, pp. 766–775, Apr.–Jun. 2020. [Online]. Available: https://doi.org/10.1109/TNSE.2018.2830307

[6] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, and S.-K. Ng, "MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks," in *Proc. 28th Int. Conf. Artif. Neural Netw.*, vol. 11730, 2019, pp. 703–716.

[7] X. Zheng and Z. Cai, "Privacy-preserved data sharing towards multiple parties in Industrial IoTs," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 5, pp. 968–979, May 2020. [Online]. Available: https://doi.org/10.1109/JSAC.2020.2980802

[8] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Söderström, "Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2018, pp. 387–395.

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

[10] C. C. Aggarwal, *Outlier Analysis*. Cham, Switzerland: Springer, 2013.

[11] D. Park, Y. Hoshi, and C. C. Kemp, "A multimodal anomaly detector for robot-assisted feeding using an LSTM-based variational autoencoder," 2017. [Online]. Available: https://arxiv.org/abs/1711.00614.

[12] B. Zong *et al.*, "Deep autoencoding Gaussian mixture model for unsupervised anomaly detection," in *Proc. 6th Int. Conf. Learn. Represent.*, 2018. [Online]. Available: OpenReview.net

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

[14] F. Angiulli and C. Pizzuti, "Fast outlier detection in high dimensional spaces," in *Proc. 6th Eur. Conf. Principles Data Min. Knowl. Discov.*, vol. 2431, 2002, pp. 15–26.

[15] A. Lazarevic and V. Kumar, "Feature bagging for outlier detection," in *Proc. 11th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2005, pp. 157–166.

[16] Y. Liang, Z. Cai, J. Yu, Q. Han, and Y. Li, "Deep learning based inference of private information using embedded sensors in smart devices," *IEEE Netw.*, vol. 32, no. 4, pp. 8–14, Jul./Aug. 2018. [Online]. Available: https://doi.org/10.1109/MNET.2018.1700349

[17] H. Zhao *et al.*, "Multivariate time-series anomaly detection via graph attention network," in *Proc. 20th IEEE Int. Conf. Data Min.*, Sorrento, Italy, 2020, pp. 841–850.

[18] I. J. Goodfellow *et al.*, "Generative adversarial networks," 2014. [Online]. Available: https://arxiv.org/abs/1406.2661.

[19] Z. Cai, Z. Xiong, H. Xu, P. Wang, W. Li, and Y. Pan, "Generative adversarial networks: A survey towards private and secure applications," 2021. [Online]. Available: https://arxiv.org/abs/2106.03785.

[20] D. Li, D. Chen, J. Goh, and S.-K. Ng, "Anomaly detection with generative adversarial networks for multivariate time series," 2018. [Online]. Available: https://arxiv.org/abs/1809.04758.

[21] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, and M. Sun, "Graph neural networks: A review of methods and applications," 2018. [Online]. Available: https://arxiv.org/abs/1812.08434.

[22] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. 6th Int. Conf. Learn. Represent.*, 2018. [Online]. Available: OpenReview.net

[23] F. Wu, A. H. Souza Jr., T. Zhang, C. Fifty, T. Yu, and K. Q. Weinberger, "Simplifying graph convolutional networks," in *Proc. 36th Int. Conf. Mach. Learn.*, vol. 97, 2019, pp. 6861–6871.

[24] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *ACM Trans. Graph.*, vol. 38, no. 5, pp. 1–12, 2019.

[25] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Connecting the dots: Multivariate time series forecasting with graph neural networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2020, pp. 753–763.

[26] D. Cao *et al.*, "Advances in neural information processing systems," in *Spectral Temporal Graph Neural Network for Multivariate Time-Series Forecasting*, vol. 33, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds. Red Hook, NY, USA: Curran Assoc., Inc., 2020, pp. 17766–17778. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/cdf6581cb7aca4b7e19ef136c6e601a5-Paper.pdf

[27] A. Vaswani *et al.*, "Attention is all you need," in *Advances in Information Processing Systems (NeurIPS)*. Red Hook, NY, USA: Curran, 2017, pp. 5998–6008.

[28] A. Blázquez-García, A. Conde, U. Mori, and J. A. Lozano, "A review on outlier/anomaly detection in time series data," 2020. [Online]. Available: https://arxiv.org/abs/2002.04236.

[29] Y. Zhang, Z. Ge, A. G. Greenberg, and M. Roughan, "Network anomography," in *Proc. 5th Internet Meas. Conf. (IMC)*, Oct. 2005, pp. 317–330.

[30] W. Lu and A. A. Ghorbani, "Network anomaly detection based on wavelet analysis," *EURASIP J. Adv. Signal Process.*, vol. 2009, Jun. 2009, Art. no. 837601.

[31] M. Munir, S. A. Siddiqui, A. Dengel, and S. Ahmed, "DeepAnT: A deep learning approach for unsupervised anomaly detection in time series," *IEEE Access*, vol. 7, pp. 1991–2005, 2019.

[32] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "LSTM-based encoder-decoder for multi-sensor anomaly detection," 2016. [Online]. Available: https://arxiv.org/abs/1607.00148.

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

[34] A. A. Cook, G. Mısırlı, and Z. Fan, "Anomaly detection for IoT time-series data: A survey," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6481–6494, Jul. 2020.

[35] M. Jones, D. Nikovski, M. Imamura, and T. Hirata, "Exemplar learning for extremely efficient anomaly detection in real-valued time series," *Data Min. Knowl. Discov.*, vol. 30, no. 6, pp. 1427–1454, 2016.

[36] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in *Proc. MLSDA 2nd Workshop Mach. Learn. Sens. Data Anal.*, 2014, pp. 4–11.

[37] H. Lu, Y. Liu, Z. Fei, and C. Guan, "An outlier detection algorithm based on cross-correlation analysis for time series dataset," *IEEE Access*, vol. 6, pp. 53593–53610, 2018.

[38] A. Siffer, P.-A. Fouque, A. Termier, and C. Largouët, "Anomaly detection in streams with extreme value theory," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2017, pp. 1067–1075.

[39] P. Senin *et al.*, "Time series anomaly discovery with grammar-based compression," in *Proc. 18th Int. Conf. Extending Database Technol. (EDBT)*, Brussels, Belgium, Mar. 2015, pp. 481–492. [Online]. Available: OpenProceedings.org

[40] H. N. Akouemo and R. J. Povinelli, "Data improving in time series using ARX and ANN models," *IEEE Trans. Power Syst.*, vol. 32, no. 5, pp. 3352–3359, Sep. 2017.

[41] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. 30th Annu. Conf. Neural Inf. Process. Syst.*, 2016, pp. 3837–3845.

[42] C. J. Maddison, D. Tarlow, and T. Minka, "A* sampling," in *Advances in Neural Information Processing Systems (NeurIPS)*. Red Hook, NY, USA: Curran, 2014, pp. 3086–3094.

[43] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," in *Proc. 5th Int. Conf. Learn. Represent.*, 2017. [Online]. Available: OpenReview.net

[44] C. Rosenbaum, T. Klinger, and M. Riemer, "Routing networks: Adaptive selection of non-linear functions for multi-task learning," in *Proc. 6th Int. Conf. Learn. Represent.*, 2018. [Online]. Available: OpenReview.net

[45] Y. Guo, H. Shi, A. Kumar, K. Grauman, T. Rosing, and R. S. Feris, "SpotTune: Transfer learning through adaptive fine-tuning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Long Beach, CA, USA, 2019, pp. 4805–4814.

[46] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *Proc. 4th Int. Conf. Learn. Represent.*, 2016.

[47] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Boston, MA, USA, 2015, pp. 1–9.

[48] A. Raganato and J. Tiedemann, "An analysis of encoder representations in transformer-based machine translation," in *Proc. EMNLP Workshop BlackboxNLP Anal. Interpreting Neural Netw. NLP*, 2018, pp. 287–297.

[49] E. Voita, R. Sennrich, and I. Titov, "The bottom-up evolution of representations in the transformer: A study with machine translation and language modeling objectives," in *Proc. Conf. Empir. Methods Natural Lang. Process. 9th Int. Joint Conf. Nat. Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 4395–4405.

[50] A. Raganato, Y. Scherrer, and J. Tiedemann, "Fixed encoder self-attention patterns in transformer-based machine translation," in *Proc. Conf. Empir. Methods Natural Lang. Process. (EMNLP)*, 2020, pp. 556–568.

[51] Y. Tay, D. Bahri, D. Metzler, D.-C. Juan, Z. Zhao, and C. Zheng, "Synthesizer: Rethinking self-attention in transformer models," 2020. [Online]. Available: https://arxiv.org/abs/2005.00743.

[52] Z. Wu, Z. Liu, J. Lin, Y. Lin, and S. Han, "Lite transformer with long-short range attention," in *Proc. 8th Int. Conf. Learn. Represent.*, 2020. [Online]. Available: OpenReview.net

[53] F. Wu, A. Fan, A. Baevski, Y. N. Dauphin, and M. Auli, "Pay less attention with lightweight and dynamic convolutions," in *Proc. 7th Int. Conf. Learn. Represent.*, 2019. [Online]. Available: OpenReview.net

[54] H. Zhou *et al.*, "Informer: Beyond efficient transformer for long sequence time-series forecasting," 2020. [Online]. Available: https://arxiv.org/abs/2012.07436.

[55] A. P. Mathur and N. O. Tippenhauer, "SwaT: A water treatment testbed for research and training on ICS security," in *Proc. Int. Workshop Cyber Phys. Syst. Smart Water Netw.*, Vienna, Austria, 2016, pp. 31–36.

[56] P. O'Neill, D. Entekhabi, E. G. Njoku, and K. H. Kellogg, "The NASA soil moisture active passive (SMAP) mission: Overview," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, 2010, pp. 3236–3239.

[57] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," in *Proc. 25th Annu. Netw. Distrib. Syst. Security Symp.*, 2018.

[58] M. Fey and J. E. Lenssen, "Fast graph representation learning with pytorch geometric," 2019. [Online]. Available: https://arxiv.org/abs/1903.02428.