Spatio-temporal Fusion Flow with Dynamic Graph Leaning for Unsupervised Multivariate Time Series Anomaly Detection*

Xinyuan Zhou[a], Shiyong Lan[a,*], Wenwu Wang[b], Yingjie Zong[c], Weiyi Wang[a], Hongyu Yang[a] and Ruiyi Lu[a]

[a]College of Computer Science, Sichuan University, Chengdu, 610065, China
[b]Centre for Vision, Speech and Signal Processing, University of Surrey, Guildford, GU2 7XH, UK
[c]School of Software Engineering, Xi'an Jiao-tong University, Xi'an, 710049, China

ARTICLE INFO

Keywords:
Anomaly detection
Unsupervised learning
Multivariate time series
Graph Neural Networks
Normalizing Flow

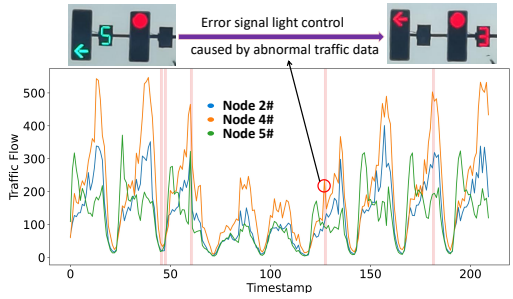
ABSTRACT

Detecting anomalies in multivariate time series (MTS) is essential for maintaining system safety in industrial environments. Due to the challenges associated with acquiring labeled data, unsupervised approaches have attracted increasing attention for anomaly detection. However, existing methods face challenges in capturing temporal dependencies within individual variable and spatial correlations among different variables. Moreover, traditional approaches that rely solely on point-wise or periodic criteria often fail to detect structurally complex anomalous patterns. To address these challenges, we propose an unsupervised anomaly detection framework, namely, Spatio-temporal Fusion Flow with Dynamic Graph Learning (STF²-DGL), which jointly models long-short-term temporal patterns and evolving inter-variable dependencies. The framework leverages an encoder to capture multi-scale temporal features, generates dynamic graph structures via spatio-temporal fusion, and integrates these representations into a structure-aware normalizing flow optimized with a graph-enhanced loss. This unified design enables STF²-DGL to effectively distinguish subtle and structurally complex patterns in multivariate time series. Extensive experiments on five real-world MTS datasets demonstrate that STF²-DGL outperforms the state-of-the-art baselines in terms of both accuracy and robustness. The code of our method is available at https://github.com/SYLan2019/STF²-DGL.

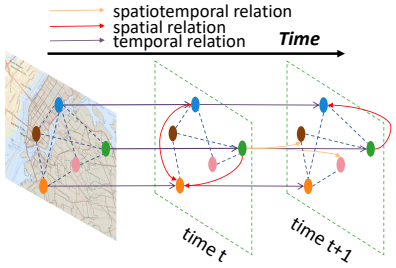
1. Introduction

Multivariate time series (MTS) are generated by multiple sensors, devices, and monitoring systems from different spatial locations. Each MTS consists of multiple variables that evolve over time [1]. For example, traffic flow data from various locations within an urban road network form a typical MTS, with each location (or node) representing a distinct variable transmitting traffic information. In each time window, a node with a specific traffic flow value represents an instance of the corresponding variable. Each variable exhibits both temporal dependence on its historical values and spatial correlation with other variables. MTS often contain valuable information that can be used to evaluate system behavior and performance trajectories. Fig. 1 demonstrates the complex spatio-temporal dependencies among the traffic nodes, as well as the impact of traffic anomaly detection on intelligent traffic management. Identification of anomalous patterns within these complex and high-dimensional data streams is essential to prevent catastrophic failures, minimize downtime, and optimize maintenance schedules.

With advances in sensor technology, large-scale MTS data are now routinely gathered during system operation in various applications. However, collecting annotations in industrial settings is often impractical or prohibitively expensive. As a result, unsupervised anomaly detection [2–4] has gained significant attention for its ability to function without the need for labeled data. Among these approaches, density-based methods [5–7] have emerged as an effective solution by estimating the likelihood of data points directly,



(a) Traffic anomaly leads to confusion in traffic control



(b) Complex spatio-temporal correlations

Figure 1: (a) Traffic anomaly leads to confusion in traffic control. For example, a traffic light is automatically set based on the normal traffic flow. The sudden appearance of abnormal flow causes the error of traffic control. (b) There are complex spatio-temporal correlations among nodes at different times, so it is necessary to use dynamic graph structure.

*This work is supported by National Natural Science Foundation of China project 62371324.
*Corresponding author: lanshiyong@scu.edu.cn.

without requiring anomaly-free training sets. These methods are built on the assumption that anomalies typically reside in low-density regions of the feature space, making the accurate modelling of data distributions a key challenge [8–11].

To better capture complex dependencies in MTS, recent research has introduced graph-based structures to model spatial correlations among variables. Graph neural network (GNN)-based models, such as graph deviation network (GDN) [12] and graph learning with transformer (GLT) [13], have demonstrated superior detection performance by leveraging inter-variable relationships. However, most existing graph-based density estimators still face three key limitations. First, the temporal feature extraction modules used in previous work are often based on standard recurrent neural networks (RNNs) [5, 6, 12, 14–16], which struggle to capture both long-term dependencies and fine-grained temporal variations, an essential requirement for modeling complex temporal dynamics within variables [17]. Second, many methods rely on static or heuristic graph construction [5, 12, 14] or consider only spatial cues in isolation [6, 7, 16], thus neglecting the joint influence of spatial structure and temporal evolution. This can lead to incomplete or biased representations of inter-variable interactions [18]. Third, existing models often overlook the structural shifts in inter-channel dependency patterns that emerge between normal and anomalous states [15, 19, 20], limiting their sensitivity to graph-level changes caused by anomalies [16].

These limitations highlight a key challenge: the need to learn variable-specific temporal features, dynamically model graph structures, and integrate both into representation learning and density estimation. On one hand, anomalies in MTS data can take various temporal forms: point anomalies may require fine-grained time resolution, while contextural or shapelet anomalies require long-range sequence modeling [21]. On the other hand, while many approaches focus on deviations in individual variable, recent studies [16] have revealed that structural shifts in inter-variable dependencies, reflected as changes in graph topology, are also strong indicators of anomalies.

To address these limitations, we propose a **S**patial-**T**emporal **F**usion **F**low with **D**ynamic **G**raph **L**earning (**STF²-DGL**) model, a novel unsupervised anomaly detection framework for MTS. First, we introduce the **T**emporal **F**eatures **E**ncoding (**TFE**) module, which integrates multivariate correlation modeling with multi-scale temporal extraction. This module leverages a composite architecture comprising a variable-wise transformer, long short term memory (LSTM), and multi-scale convolutional layers operating over varying receptive fields, enabling the model to concurrently capture fine-grained temporal fluctuations and long-range dependencies. Second, we propose the **A**daptive **D**ynamic **G**raph **G**enerating (**ADGG**) module, which decouples temporal, spatial representations and then recombines them through a spatio-temporal fusion framework that incorporates cross-attention and gated integration mechanisms. This design allows the resulting graphs to dynamically reflect latent interdependencies and structural transitions across both contexts and variables. Third, we develop a seamless integration between graph representations and normalizing flows through our **G**raph-**E**nhanced **F**low (**GEF**) module, which enables the model to effectively capture both spatial and temporal dependencies while incorporating graph structure as an inductive constraint to enhance representation learning. The temporal features extracted by the TFE module serve as node features, while the adjacency matrices produced by ADGG define the graph structure. These learned embeddings are then used to condition a normalizing flow that models the complex multivariate distribution. Moreover, we incorporate a graph structure-aware enhancement loss into the training objective. This loss promotes temporal smoothness and penalizes structural deviations for anomalous samples by measuring graph similarity across the batch. Our contributions are summarized as follows:

- We introduce the anomaly detection framework **STF²-DGL**, where dynamic graphs are constructed by modeling the spatio-temporal dependencies among the variables, and these graphs are then used to accurately identify the anomalous distributions.
- We design the **TFE** module to disentangle temporal feature learning with a variable-specific approach, thus enabling a robust capture of both long- and short-term patterns of each individual variable.
- We propose the **ADGG** module, which generates dynamic graph structures that are aware of both temporal dynamics and latent spatial correlations, a capability achieved by modeling these two contexts separately before their fusion.
- We propose the **GEF** module, which conditions the normalizing flow on learned dynamic graph representations, with the optimization process being further regularized by a novel loss function designed to penalize graph structural deviations between normal and anomalous samples.
- Extensive experiments are conducted on five real-world datasets to compare STF²-DGL against ten baseline methods, and the results demonstrate its superiority over the state-of-the-art (SOTA) methods.

2. RELATED WORK

2.1. Graph Structure Learning

In the field of MTS anomaly detection, methods for graph structure learning have become essential to improve detection accuracy by modeling complex relationships among variables. Early methods [12, 14, 22–24] focus on building static graphs or predefined adjacency matrices, assuming fixed relationships between variables. For instance, Graph WaveNet [22] proposes an adaptive method to learn graph structure based on training data. DSTAGNN [23] uses an improved Wasserstein distance [24] to calculate the transfer

costs between the node vectors in historical sequences, thus constructing the graph structure. However, once established, these graph structures remain fixed and do not adapt dynamically to varying inputs during the testing phase. In addition, GANF [5] models causal relationships between variables in MTS by learning a directed acyclic graph (DAG), which represents the graph as a continuous adjacency matrix. However, it does not account for the evolving interdependencies among the variables over time. As a result, it neglects the dynamic nature of real-world systems, making the approach less responsive to temporal changes in the time series.

In contrast, dynamic graphs capture the evolving spatial relationships between variables over time, offering a more accurate representation of complex structural patterns than static graphs. This temporal adaptability has led to increased research interest in dynamic graph learning across various real-world applications. For example, in methods [6, 15, 16, 25], the similarity between nodes is calculated for each time window through the self-attention mechanism. However, they only account for basic correlations between spatial nodes after aligning time windows, overlooking the rich long- and short-term temporal dynamics within each window. DGL-LS [26] builds dynamic graphs by aggregating sensor node features over multiple time intervals and utilizes an attention-based gating mechanism to update node dynamics, producing a sequence of adjacency matrices that reflect the evolving temporal graph structure. However, this approach relies on static initial node features to initiate dynamic graph learning and uses average aggregation (AGG) across time periods, which can potentially obscure important temporal variations. To address this limitation, we propose an ADGG module that decouples and recombines dynamic temporal and spatial information, enabling STF²-DGL to adaptively generate dynamic adjacency matrices that accurately reflect evolving interdependencies among the variables.

2.2. Unsupervised MTS Anomaly Detection

Broadly, deep learning-based unsupervised anomaly detection methods in MTS fall into two main paradigms [27]: forecasting-based methods (e.g., RNN [28], CNN [2], and HTM [29]) that recognize anomalies when the prediction error exceeds a certain threshold, and reconstruction-based methods (e.g., AE [19, 30], VAE [31], and Transformer [32, 33]) that identify anomalies through reconstruction discrepancies. However, forecasting models accumulate prediction errors over time, making them suitable only for short-term forecasting but not for complex long-term events [28]. Others predominantly focus on temporal patterns while neglecting evolving spatial correlations. For instance, LSTM-NDT [34] achieves dynamic threshold adaptation but assumes fixed inter-variable relationships, failing to detect anomalies caused by dependency shifts. Meanwhile, reconstruction-based models face a key limitation: insufficient reconstruction capacity leads to incomplete generation of normal regions, while excessive capacity unintentionally reconstructs anomalous regions. For example, OmniAnomaly [1] models latent spaces with Gaussian priors, struggling to capture multimodal patterns in complex MTS, while GAN-based methods [35] suffer from mode collapse when anomalies induce heterogeneous dependency structures. Additionally, contrastive-based methods [36, 37] employ multi-view augmentation to improve representational discriminability. This involves pulling similar samples together and pushing dissimilar samples apart within the representation space, with the anomaly score derived from the representation discrepancy [38].

More recently, GANF introduced a pioneering approach that adopts the density estimation strategy to tackle unsupervised MTS anomaly detection tasks. Furthermore, MTGFlow [20] and enhanced MTGFlow-cluster [6] leveraged dynamic graph structures and variable/cluster-aware normalizing flows to achieve robust and fine-grained density estimation for MTS. In addition, Graph-MoE [7] proposes a graph mixture of experts (Graph-MoE) network for MTS anomaly detection, which utilizes hierarchical representations from multiple GNN layers and integrates them through a memory-augmented router to model complex dependencies between variables and capture temporal correlations across levels. However, these methods use Gaussian assumptions and ignore the fact that both normal and abnormal states may contain multiple sub-states. USD [15] introduced an unsupervised soft contrastive learning framework, which addresses the limitations imposed by the Gaussian distribution assumption by using more accurate multi-Gaussian representations. MADGA [16] reframed anomaly detection as a graph alignment (GA) problem, revealing that anomalies manifest through significant shifts in the interdependency graph series from normal to anomalous states. However, these methods fail to adequately capture the topological relationships among sensors and the temporal variations within each sensor, leaving room for improvement in effectively modeling high-dimensional MTS data with complex and potentially significant interdependencies.

3. PRELIMINARY

In this section, we provide a brief introduction to dynamic graph and normalizing flow to better understand STF²-DGL and its extension.

3.1. From MTS to Dynamic Graph

Multivariate Time Series: MTS is a set of K correlated, time-dependent sequences, or variables, denoted as $D = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_K\}$. Each variable $\mathbf{x}_k \in \mathbb{R}^L$ consists of L observations. Following the procedure in [5], we normalize each time series $\mathbf{x}_k$ independently using its standard score:

$$\mathbf{x}_k = \frac{\mathbf{x}_k - \text{mean}(\mathbf{x}_k)}{\text{std}(\mathbf{x}_k)}, \tag{1}$$

where $\text{mean}(\mathbf{x}_k)$ and $\text{std}(\mathbf{x}_k)$ denote the mean and standard deviation of $\mathbf{x}_k$, respectively, of the time series for the k-th variable. To preserve local temporal correlations, we

employ a sliding-window method characterized by window size M and stride size S to sample the normalized MTS. By adjusting the parameters M and S, we obtain training samples denoted by $\mathbf{X}^c$, where c is the sampling index, and $\mathbf{X}^c$ succinctly represents the subsequence $\mathbf{X}^{cS:cS+M}$.

Data Augmentation: Following [15], we use the data augmentation strategy that generates new plausible samples by exploring the neighborhood discovery and linear interpolation strategies. For each sampled point $\mathbf{x}^c$, its neighborhood $N(\mathbf{x}^c)$ is defined using the Euclidean distance metric $ED(\cdot,\cdot)$, and consists of points $\mathbf{x}^c_{\text{new}}$ satisfying the criterion: $\text{ED}(\mathbf{x}^c, \mathbf{x}^c_{\text{new}}) \leq \epsilon$, where ϵ is a predefined threshold that controls the neighborhood radius. After that, a new sample point $\mathbf{x}^c_{\text{new}}$ is generated between the two points using the linear interpolation mechanism:

$$\mathbf{x}^c_{\text{new}} = \alpha \mathbf{x}^c + (1-\alpha)\mathbf{x}^c_{\text{new}}, \tag{2}$$

where $\alpha \sim U(0,1)$ is a random coefficient sampled from a uniform distribution. This method enables smooth transitions between samples, helping fill gaps in the data space and create a continuum of variations. As a result, it increases the diversity and representativeness of the data set, contributing to the improvement of the generalization and robustness of the model.

Dynamic Graph: For the sampled MTS window $\mathbf{X}^c \in \mathbb{R}^{K\times M}$, representing K variables over a time window of size M (i.e., $\mathbf{x}^c_j$, as the j-th row of $\mathbf{X}^c$, corresponds to a row vector of the j-th variable with M time steps, here $j = 1, 2, \ldots, K$), we construct a corresponding dynamic graph $\mathcal{G}^c = (\mathcal{V}^c, \mathcal{E}^c)$. The graph consists of a set of nodes $\mathcal{V}^c = \{v^c_1, v^c_2, \ldots, v^c_K\}$, with their features represented by $\mathcal{F}^c = \{f^c_1, f^c_2, ..., f^c_K\}$. The connections between variables are described by the edge set $\mathcal{E}^c \subseteq \mathcal{V}^c \times \mathcal{V}^c$, which is represented as a weighted adjacency matrix $\mathbf{A}^c \in \mathbb{R}^{K\times K}$.

Following [6], we calculate the query and key vectors for each variable k as $\mathbf{x}^c_k\mathbf{W}^{Query}$ and $\mathbf{x}^c_k\mathbf{W}^{Key}$ through a self-attention mechanism, where $\mathbf{W}^{Query}$ and $\mathbf{W}^{Key}$ are the query and key weights, respectively. The pairwise relationship e^c_{ij} between variable i and variable j is then obtained as follows:

$$e^c_{ij} = \frac{(\mathbf{x}^c_i\mathbf{W}^{Query})(\mathbf{x}^c_j\mathbf{W}^{Key})^T}{\sqrt{M}}. \tag{3}$$

The attention score a^c_{ij} is used to quantify the pairwise relation from node i to node j, calculated by:

$$a^c_{ij} = \frac{\exp(e^c_{ij})}{\sum_{j=1}^K \exp(e^c_{ij})}, \quad \mathbf{A}^c = \begin{bmatrix} a^c_{11} & \cdots & a^c_{1K} \\ \vdots & \ddots & \vdots \\ a^c_{K1} & \cdots & a^c_{KK} \end{bmatrix}, \tag{4}$$

where the attention matrix $\mathbf{A}^c$ serves as the adjacency matrix of the learned graph. Crucially, the graph structure $\mathcal{G}^c$ (specifically $\mathcal{V}^c$, $\mathcal{E}^c$ and $\mathbf{A}^c$) is dependent on the specific window sample $\mathbf{X}^c$, allowing the graph to adapt and capture the evolving inter-variable dependencies over time, thus distinguishing from a static graph where the structure remains fixed across all samples.

3.2. Flow-based Anomaly Detection

Normalizing Flow: Normalizing flow is an unsupervised density estimation method aimed at transforming the original data distribution $\mathcal{X}$ into a target distribution $\mathcal{Z}$ that is easier to handle through invertible transformations. Let $\mathbf{x} \in \mathbb{R}^d$ be a sample from distribution $\mathcal{X}$, and $\mathbf{z} \in \mathbb{R}^d$ be a sample from the target distribution $\mathcal{Z}$, where d is the embedding dimension. Since f_θ is bijective and differentiable, its inverse f_θ^{-1} also exists. Thus, the invertible bijection f_θ satisfies the following condition:

$$\mathbf{z} = f_\theta(\mathbf{x}), \quad \mathbf{x} = f_\theta^{-1}(\mathbf{z}). \tag{5}$$

Based on the change of variables formula, we obtain:

$$P_{\mathcal{X}}(\mathbf{x}) = P_{\mathcal{Z}}(\mathbf{z}) \left|\det \frac{\partial f_\theta}{\partial \mathbf{x}^T}\right|, \tag{6}$$

$$\mathcal{Z} = \mathcal{N}(\boldsymbol{\mu}, \mathbf{I}), \tag{7}$$

$$\boldsymbol{\mu} \sim \mathcal{N}(\mathbf{0}, \mathbf{1}), \tag{8}$$

where $\mathbf{I}$ is the identity matrix, and $\left|\det \frac{\partial f_\theta}{\partial \mathbf{x}^T}\right|$ denotes the determinant of the Jacobian matrix of the mapping f_θ, determining the scale of probability density change from one distribution to another.

In practical applications, constructing invertible and analytically tractable network structures (such as MAF [39] and RealNVP [40]) enables flexible and accurate density estimation of the original distribution $\mathcal{X}$, updating parameters θ through maximum likelihood estimation (MLE) during training. Additionally, by incorporating conditional information C, we can construct a conditional normalizing flow with stronger representational capabilities. The mapping is thus represented as $\mathbf{z} = f_\theta(\mathbf{x} \mid C)$, and the objective function for optimizing model parameters is given by:

$$\theta^* = \arg\max_\theta \left(\log\left(P_{\mathcal{Z}}(f_\theta(\mathbf{x} \mid C))\right) + \log\left(\left|\det \frac{\partial f_\theta}{\partial \mathbf{x}^T}\right|\right)\right). \tag{9}$$

Anomaly Detection under Multi-Gaussian Assumption: Unsupervised anomaly detection focuses on capturing anomalies in low-density regions where normal data clusters are compact and anomalous points are more dispersed. For complex systems described by MTS, the distribution of normal behavior is often multi-modal, comprising several distinct operational patterns [15]. Under the multi-Gaussian assumption, unsupervised anomaly detection posits that normal MTS behavior consists of multiple Gaussian-like modes. Anomalies are the segments $\mathbf{x}^c$ whose distributions are inconsistent with any of the Gaussian distributions representing normal behavior and are flagged through a composite density threshold $\rho(\mathbf{x}^c) < \theta$.

4. METHOD

4.1. Overview of STF²-DGL

Given an MTS dataset $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_K\}$ as defined in Section 3.1, our objective is to detect anomalous patterns

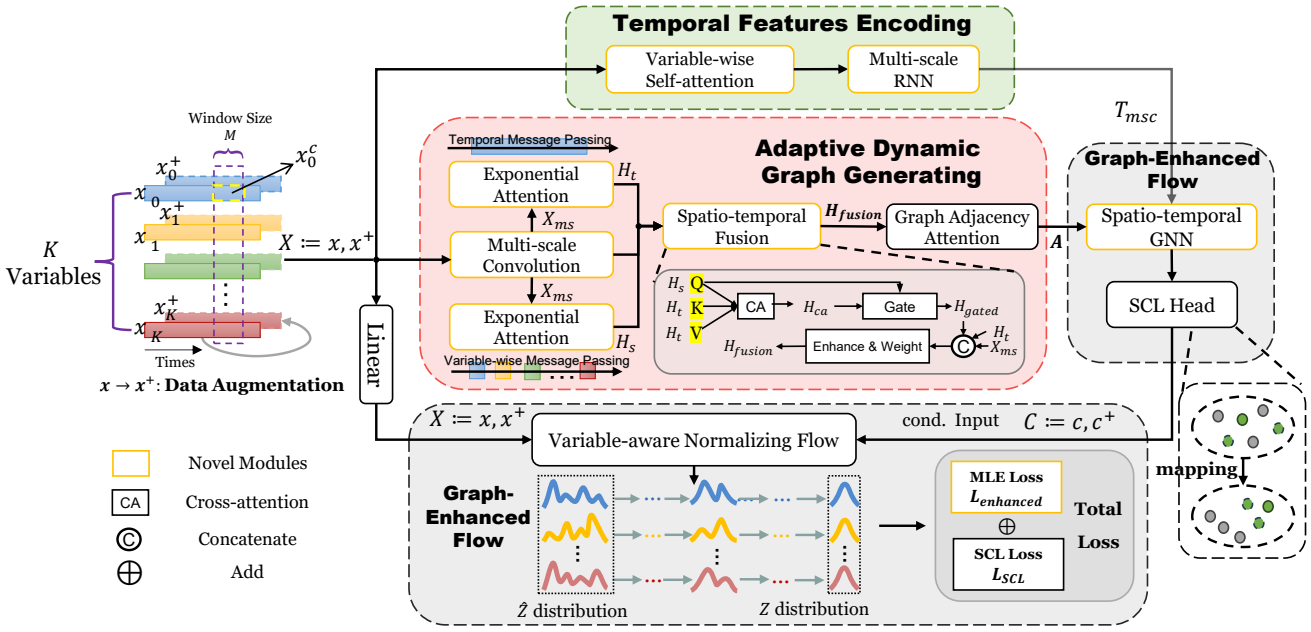


Figure 2: Overview of the proposed STF²-DGL framework, composed of three modules: Temporal Features Encoding (TFE), Adaptive Dynamic Graph Generating (ADGG), and Graph-Enhanced Flow (GEF). TFE captures variable-wise temporal patterns. ADGG dynamically constructs graph structures via adaptive spatio-temporal fusion. GEF integrates spatio-temporal condition learning, variable-aware normalizing flow, and enhanced loss, facilitating accurate anomaly detection.

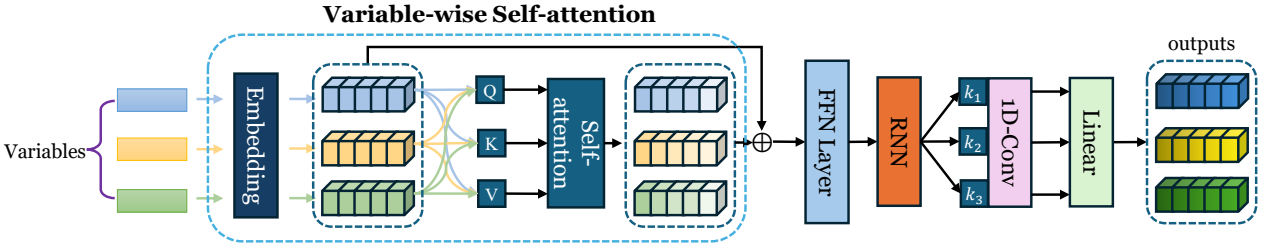


Figure 3: Illustration of the TFE module. The module independently embeds the raw sequences of different variables and captures temporal dependencies via self-attention mechanism. It further enhances contextual modeling through an LSTM and multi-scale 1D convolutions, whose outputs are fused through a linear layer.

without any labeled examples. We obtain the adjacency matrix $\mathbf{A}^c$ by learning the interdependencies between variables and encode the MTS data as node feature embeddings $\mathbf{T}^c$ to construct a dynamic graph $\mathcal{G}^c$. Our work is based on the hypothesis, in line with [16], that anomalies manifest themselves as low-probability samples under a learned density model or as deviations in a graph structure. To effectively address both aspects, we are motivated by the insights from prior research [6], which established the superiority of variable-specific density estimation over global models and highlighted the mutual nature of inter-variable dependencies.

Accordingly, our methodology is designed to be variable-aware and structurally informed. We begin with variable-wise temporal feature extraction to preserve the individual characteristics necessary for unique density modeling. Concurrently, we construct a dynamic graph to explicitly model the spatio-temporal interactions and mutual dependencies among variables. This approach enables the development of a more accurate density model for each variable, from which we derive the final anomaly score as the graph similarity-enhanced negative log-likelihood.

Our key contribution to STF²-DGL is the refinement of spatio-temporal condition generation, enabling the model to capture the spatio-temporal features within MTS data at each step for precise anomaly detection. Furthermore, we maximize the utility of the generated dynamic graph structures by combining graph similarity with soft contrastive learning, which constrains the network optimization to better approximate the true data distribution. The overview of STF²-DGL can be seen in Fig. 2. In general, STF²-DGL is composed of three main modules: Temporal Features Encoding (TFE), Adaptive Dynamic Graph Generating (ADGG), and Graph-Enhanced Flow (GEF).

4.2. Temporal Features Encoding

The detailed architecture of the TFE module is shown in Fig. 3. The TFE module, inspired by the original iTransformer [41], independently embeds the raw sequences of different variables in the MTS. It further enhances temporal

representation by integrating improved contextual memory capabilities and multi-scale feature extraction, achieved through the addition of an LSTM layer and a multi-scale convolutional module.

We design a variable-wise self-attention module to capture the unique temporal characteristics of each variable. Together with a feed-forward network for non-linear feature extraction, this module forms a variable-wise Transformer block. This block allows our model to efficiently learn multivariate correlations, particularly excelling in scenarios with long-term dependencies. It first embeds the sampled window series $\mathbf{x}_k^c$ of each variable k into a token representation through a multi-layer perceptron (MLP) as follows:

$$\mathbf{e}_k^c = \text{Embedding}(\mathbf{x}_k^c), \tag{10}$$

where $\mathbf{e}_k^c$ represents the embedding of the k-th variable in the sampling window c. The outputs for each instance are then aggregated into $\mathbf{E}^c = \{\mathbf{e}_1^c, \mathbf{e}_2^c, ..., \mathbf{e}_K^c\}$. The sequence $\mathbf{E}^c$ passes through a stack of L variable-wise Transformer blocks. Within each block, self-attention is applied along the variate dimension to explicitly model multi-variable correlations, while the feed-forward network captures nonlinear temporal patterns within each variable. The output of this stack is denoted as:

$$\mathbf{T}_{trm}^c = \text{TransformerBlocks}(\mathbf{E}^c), \tag{11}$$

where $\mathbf{T}_{trm}^c$ represents the refined temporal representations after multi-variable interaction modeling, with *trm* being an abbreviation of the Transformer blocks. To further enhance the temporal expressiveness of each variable, a shared MLP is applied to project the output into a higher-level representation:

$$\hat{\mathbf{T}}_{trm}^c = \text{MLP}(\mathbf{T}_{trm}^c), \tag{12}$$

where $\hat{\mathbf{T}}_{trm}^c$ denotes the projected feature representation of the sampled window c after the MLP layer.

To further enhance temporal modeling capability, we integrate an LSTM layer and a multi-scale convolutional module. Specifically, the feature representation $\hat{\mathbf{T}}_{trm}^c$ is sequentially processed by an LSTM network to capture richer sequential dependencies:

$$\mathbf{T}_{lstm}^c = \text{LSTM}(\hat{\mathbf{T}}_{trm}^c). \tag{13}$$

The LSTM effectively addresses one of the limitations of the Transformer module, which struggles to capture fine-grained temporal features over long sequences. Subsequently, parallel convolutional operations with varying kernel sizes $(k_1, k_2, \ldots, k_p)$ are applied to extract temporal patterns at multiple scales:

$$\mathbf{T}_{conv,i}^c = \text{Conv1D}^{(k_i)}(\mathbf{T}_{lstm}^c), \quad i = 1, 2, \ldots, p. \tag{14}$$

These multi-scale convolutional outputs are adaptively aggregated via learnable weights to produce a comprehensive multi-scale representation:

$$\mathbf{T}_{msc}^c = \sum_{i=1}^{p} \alpha_i \mathbf{T}_{conv,i}^c, \tag{15}$$

where $\mathbf{T}_{msc}^c$ is the final multi-scale temporal feature embedding, and α_i represents the importance weight for the i-th scale. This hybrid configuration allows the LSTM and the multi-scale convolution to extend the temporal analysis capabilities of the variable-wise Transformer. Together, this setup establishes a powerful framework for analyzing time series data, ensuring detailed recognition and modeling of both immediate and long-range dependencies, as well as the intricate correlations between variables, critical for anomaly detection in complex datasets.

4.3. Adaptive Dynamic Graph Generating

The ADGG module serves as a critical component for capturing and representing dynamic interdependencies between variables across temporal dimensions. Unlike traditional static graph approaches, ADGG employs a spatio-temporal fusion method for generating graphical structures, and dynamically adapts its structure based on input data characteristics, enabling more accurate representation of evolving relationships between time series variables.

As an initial step, ADGG extracts temporal features at multiple scales using a convolutional structure, as illustrated in Eqs. 14–15, where the input $\mathbf{X}^c \in \mathbb{R}^{K\times M}$ is refined into the multi-scale temporal representation $\mathbf{X}_{ms}^c \in \mathbb{R}^{K\times M}$. At the core of ADGG is a dual-stream attention mechanism, composed of parallel spatial and temporal attention layers. This architecture is designed to jointly capture dependencies across both spatial and temporal dimensions from the multi-scale features $\mathbf{X}_{ms}^c$.

Spatial Attention Layer: The spatial attention layer models the relationships between variables, where each variable is treated as a node, while its features are the corresponding temporal patterns. Following the Graph Attention Network (GAT-v2) [42] paradigm, the pairwise attention scores between variables i and j are computed as:

$$\mathbf{e}_{ij}^c = \mathbf{a}^T \text{LeakyReLU}(\mathbf{W}^c[\mathbf{x}_{ms,i}^c \| \mathbf{x}_{ms,j}^c]), \tag{16}$$

where $\mathbf{x}_{ms,i}^c$ and $\mathbf{x}_{ms,j}^c \in \mathbb{R}^M$ are the multi-scale temporal feature vectors for the variables i and j, $\mathbf{W}^c$ is the weight matrix of a shared linear transformation, $\mathbf{a}$ is the weight vector of the attention mechanism, and $\|$ denotes vector concatenation.

The attention coefficients α_{ij}^c are then derived by normalizing the scores across all possible neighbors using the softmax function:

$$\alpha_{ij}^c = \text{softmax}(\mathbf{e}_{ij}^c) = \frac{\exp(\mathbf{e}_{ij}^c)}{\sum_{j=1}^{K} \exp(\mathbf{e}_{ij}^c)}. \tag{17}$$

Finally, the updated representation for each variable $\mathbf{h}_i^c$ is computed as a weighted sum of the linearly transformed features of its neighbors, followed by a non-linear activation $\sigma(\cdot)$:

$$\mathbf{h}_i^c = \sigma\Big(\sum_{j=1}^{K} \alpha_{ij}^c \mathbf{x}_j^c\Big). \tag{18}$$

The complete output of the spatial attention layer is the set of all updated variable representations, denoted as $\mathbf{H}_s^c = \{\mathbf{h}_1^c, \mathbf{h}_2^c, ..., \mathbf{h}_K^c\}$.

Temporal Attention Layer: Similarly, the temporal attention layer is designed to capture dependencies among different time steps. This is achieved by first transposing the input feature matrix to $\mathbf{X}_t^c = (\mathbf{X}_{ms}^c)^\top \in \mathbb{R}^{M\times K}$. In this configuration, each time step is treated as a node, whose features are the collective states of all K variables at that instant. The same attention mechanism described above is then applied in all time steps.

The operations of these two parallel layers can be concisely summarized as:

$$\mathbf{H}_s^c = \text{SpatialAttentionLayer}(\mathbf{x}_{ms}^c), \tag{19}$$

$$\mathbf{H}_t^c = \text{TemporalAttentionLayer}(\mathbf{x}_t^c). \tag{20}$$

The ADGG module then employs a gating-based cross-attention mechanism that adaptively combines multi-scale raw features with spatial and temporal attention features. In this cross-attention scheme, spatial features are used as queries to guide the extraction of temporal context, while temporal features serve as keys and values. By querying temporal features with spatial representations, the module can effectively infer which temporal behaviors are relevant for establishing spatial dependencies, enabling the resulting graph structure to reflect both spatial locality and temporal correlation. The cross-attention operation is formulated as follows.

$$\mathbf{H}_{ca}^c = \text{MultiheadAttention}(\mathbf{H}_s^c, \mathbf{H}_t^c, \mathbf{H}_t^c), \tag{21}$$

where $\mathbf{H}_{ca}^c$ represents the fused features capturing the interaction between spatial and temporal representations. Afterwards, a feature gating mechanism is employed to control information flow between spatial and fused features:

$$\mathbf{g} = \sigma(\text{MLP}([\mathbf{H}_s^c \| \mathbf{H}_{ca}^c])), \tag{22}$$

$$\mathbf{H}_{gated}^c = \mathbf{H}_s^c \odot \mathbf{g} + \mathbf{H}_{ca}^c \odot (\mathbf{1} - \mathbf{g}), \tag{23}$$

where $\mathbf{H}_{gated}^c$ denotes the gated enhanced spatial feature, σ represents the sigmoid activation function, and $\odot$ indicates element-wise multiplication. Correspondingly, the expression $(\mathbf{1} - \mathbf{g})$ is also an element-wise subtraction, where $\mathbf{1}$ has a same shape as $\mathbf{g}$, but with all ones.

To effectively integrate the heterogeneous information from multi-scale raw features ($\mathbf{x}_{ms}^c$), the gated spatial features ($\mathbf{H}_{gated}^c$), and the temporal features ($\mathbf{H}_t^c$), we designed a dual-path fusion module which serves as a dynamic weight generator. The core motivation behind this design is to simultaneously capture complex non-linear interactions among the feature streams while adaptively preserving and re-weighting their original information. This is achieved through two parallel, complementary paths.

The first path is designed to model the high-order, non-linear relationships between the spatial, temporal, and raw features. A simple concatenation or summation would fail to capture these intricate dependencies. Therefore, we concatenate the three feature sets and process them through an MLP. This allows the model to learn a powerful, blended representation that synthesizes information from all sources into a new, enhanced feature space $\mathbf{H}_{enhanced}^c$:

$$\mathbf{H}_{enhanced}^c = \text{MLP}([\mathbf{X}_{ms}^c \| \mathbf{H}_{gated}^c \| \mathbf{H}_t^c]). \tag{24}$$

Concurrently, it is crucial to control the information flow from the original feature streams, as their relative importance may vary depending on the context. To this end, the second path implements a dynamic, content-adaptive gating mechanism. Instead of using static, learnable weights, this path generates a set of dynamic weights $\mathbf{W}_{f,1}, \mathbf{W}_{f,2}, \mathbf{W}_{f,3}$ based on the combined information from all three feature streams. These weights reflect the context-dependent importance of each stream and are used to compute an adaptive linear combination of the original features, denoted as $\mathbf{H}_{weighted}^c$:

$$\mathbf{H}_{weighted}^c = \mathbf{X}_{ms}^c \odot \mathbf{W}_{f,1} + \mathbf{H}_{gated}^c \odot \mathbf{W}_{f,2} + \mathbf{H}_t^c \odot \mathbf{W}_{f,3}. \tag{25}$$

This path ensures that valuable information from the original streams is preserved and dynamically modulated, rather than being lost during non-linear transformation.

This dual-path architecture allows the model to learn a rich and robust feature landscape. The final output representation $\mathbf{H}_{fusion}^c$ is produced by integrating these two complementary paths through a residual-like connection, followed by layer normalization:

$$\mathbf{H}_{fusion}^c = \text{LayerNorm}(\mathbf{H}_{weighted}^c + \mathbf{H}_{enhanced}^c). \tag{26}$$

This final step combines the newly synthesized features ($\mathbf{H}_{enhanced}^c$) with the adaptively preserved original features ($\mathbf{H}_{weighted}^c$), yielding a comprehensive representation that is both expressive and well-grounded. Subsequently, this representation is fed into the self-attention layer described in Section 3.1 to generate the learned graph structure $\mathbf{A}^c \in \mathbb{R}^{K\times K}$, which captures the complex dependencies within the multivariate time series.

In conclusion, the ADGG module represents a novel approach to dynamic graph generation for MTS data. It synergistically integrates the cross-attention mechanism and an adaptive fusion block to capture complex spatio-temporal dependencies. As a result, the ADGG module yields a more realistic dynamic graph structure, providing a powerful structural prior for downstream processing of our model.

4.4. Graph-Enhanced Flow

Inspired by [6, 15], our GEF module is designed to generate a comprehensive spatio-temporal condition vector $\mathbf{C}^c$ by integrating multi-scale temporal features ($\mathbf{T}_{msc}^c$) with adaptive graph representations ($\mathbf{A}^c$). To achieve this, the module first employs a spatio-temporal GNN, composed of a GCN layer and a subsequent GRU layer. The GCN layer initially captures spatial dependencies to produce spatially-aggregated features $\mathbf{C}_{GCN}^c$, while the GRU layer then models the sequential dynamics to yield a spatio-temporal representation $\mathbf{C}_{GRU}^c$. This sequential architecture separates spatial and temporal processing to ensure computational efficiency

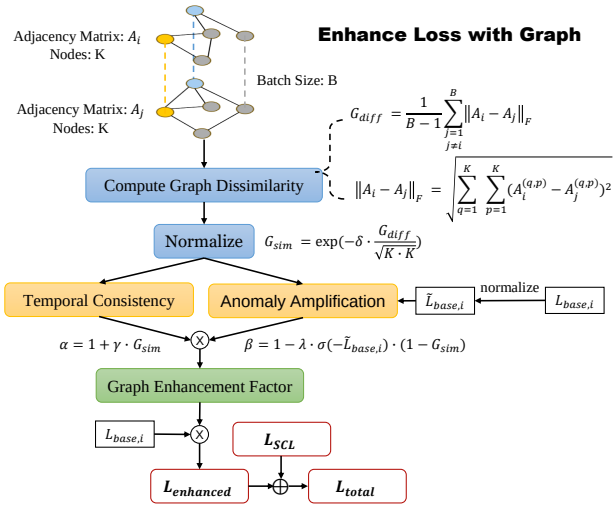


Figure 4: Illustration of the Graph-Enhanced Flow (GEF) module in STF²-DGL. The module computes graph dissimilarity among samples and integrates temporal consistency for anomaly amplification. The loss function is enhanced by incorporating a graph enhancement factor to effectively distinguish anomalous graph structures.

and the generation of a stable conditioning vector. Finally, this representation is passed to an MLP that serves as a soft contrastive learning head, which projects the features into a new space to pull similar samples closer and push dissimilar ones apart [15]. This hierarchical pipeline, which produces the final condition vector $\mathbf{C}^c$, is formally expressed as:

$$\mathbf{C}^c = \text{MLP}\left(\text{GRU}\left(\text{GCN}(\mathbf{T}^c_{msc}, \mathbf{A}^c)\right)\right). \tag{27}$$

Our graph-conditioned normalizing flow learns a bijective mapping between the input distribution and a simpler base distribution:

$$\begin{aligned}\mathcal{L}^c_{base} &= \frac{1}{K}\sum_{k=1}^{K}\log\left(P_{\mathcal{Z}_k}(f^\theta(\mathbf{x}^c_k \mid \mathbf{C}^c_k)) \cdot \left|\det\left(\frac{\partial f^\theta}{\partial \mathbf{x}^{c\,T}_k}\right)\right|\right) \\ &\approx \frac{1}{K}\sum_{k=1}^{K}\left(-\frac{1}{2}\|\mathbf{z}^c_k - \boldsymbol{\mu}_k\|_2^2 + \log\left|\det\left(\frac{\partial f^\theta}{\partial \mathbf{x}^{c\,T}_k}\right)\right|\right),\end{aligned} \tag{28}$$

where K is the number of nodes per graph , $\mathbf{z}$, $\mathcal{Z}$ and $\boldsymbol{\mu}$ are described in Section 3.2.

A key innovation of our approach is the novel graph-aware loss function that enhances detection performance by leveraging temporal consistency in graph structures, as illustrated in Fig. 4. Our core insight is that temporally adjacent normal samples exhibit similar graph structures, while anomalous samples produce distinct graph patterns [16, 43]. We formalize this by first quantifying the structural difference of a sample's graph from its peers within a batch. This is achieved by calculating the smoothed graph dissimilarity G^c_{diff}, using the Frobenius norm:

$$\begin{aligned}G^c_{\text{diff}} &= \frac{1}{B-1}\sum_{\substack{j=1 \\ j\neq c}}^{B}\|\mathbf{A}^c - \mathbf{A}^j\|_F \\ &= \frac{1}{B-1}\sum_{\substack{j=1 \\ j\neq c}}^{B}\sqrt{\sum_{q=1}^{K}\sum_{p=1}^{K}\left(\mathbf{A}^c_{(q,p)} - \mathbf{A}^j_{(q,p)}\right)^2},\end{aligned} \tag{29}$$

where B is the sampling batch size, and $\Omega(\cdot,\cdot)$ represents the graph enhancement factor. The choice of the Frobenius norm as the metric for graph dissimilarity is particularly well-suited for our MTS context. The fundamental reason is that the graphs within a batch, which represent temporally sequential samples, operate on a fixed node set. This not only bypasses the complex graph isomorphism problem but also allows the Frobenius norm to directly and efficiently quantify the total number of changes in edges between two moments, thus precisely measuring the actual change in graph structure.

Based on this dissimilarity, we compute a normalized graph similarity score G^c_{diff}, where a higher value indicates greater consistency with the batch:

$$G^c_{\text{sim}} = \exp\left(-\delta \cdot \frac{G^c_{\text{diff}}}{\sqrt{K \cdot K}}\right), \tag{30}$$

where δ is a scale hyperparameter.

This similarity score is then used to construct two modulating factors. The first is the temporal consistency factor α^c, which directly reflects the structural consistency:

$$\alpha^c = 1 + \gamma \cdot G^c_{\text{sim}}, \tag{31}$$

where γ is a hyperparameter. The second is the anomaly amplification factor β^c, which strengthens the learning signal for structurally distinct samples:

$$\beta^c = 1 - \lambda \cdot \sigma(-\hat{\mathcal{L}}^c_{base}) \cdot (1 - G^c_{\text{sim}}), \tag{32}$$

where $\hat{\mathcal{L}}^c_{base}$ denotes the z-score normalized log-probability, $\sigma(\cdot)$ is the sigmoid function, and λ is a hyperparameter.

Finally, these two factors are combined into an overall graph enhancement factor $\Omega(\cdot)$:

$$\Omega(\mathbf{A}^c, \mathcal{L}^c_{base}) = \alpha(\mathbf{A}^c) \cdot \beta(\mathbf{A}^c, \mathcal{L}^c_{base}), \tag{33}$$

which modulates the base loss $\mathcal{L}^c_{base}$ to yield the final enhanced loss $\mathcal{L}_{enhanced}$ as follows:

$$\mathcal{L}_{enhanced} = \frac{1}{B}\sum_{c=1}^{B}\left(\mathcal{L}^c_{base} \cdot \Omega(\mathbf{A}^c, \mathcal{L}^c_{base})\right). \tag{34}$$

In summary, this enhanced loss function establishes a more sophisticated learning objective that goes beyond simple density estimation. The enhancement factor $\Omega(\cdot)$ acts as a dynamic, sample-wise regularizer, amplifying the penalty

Algorithm 1: STF²-DGL: Spatio-temporal Condition Generation Module

Input: Sampling windows data: $\mathbf{X}^c$, Network parameters
Output: spatio-temporal conditions: $\mathbf{C}^c$

Function `Forward`($\mathbf{X}^c$)**:**
- $\mathbf{T}^c_{trm} \leftarrow$ VariableTransformer($\mathbf{X}^c$) by Eqs. 10–11 ; `// Variable-wise Transformer blocks`
- $\hat{\mathbf{T}}^c_{trm} \leftarrow$ MLP($\mathbf{T}^c_{trm}$) by Eq. 12 ;
- $\mathbf{T}^c_{lstm} \leftarrow$ LSTM($\hat{\mathbf{T}}^c_{trm}$) by Eq. 13 ;
- $\mathbf{T}^c_{msc} \leftarrow$ MultiScaleConv($\mathbf{T}^c_{lstm}$) by Eqs. 14–15 ; `// Multi-scale temporal representation`
- $\mathbf{X}^c_{ms} \leftarrow$ MultiScaleConv($\mathbf{X}^c$) by Eqs. 14–15 ;
- $\mathbf{H}^c_s, \mathbf{H}^c_t \leftarrow$ SpatialAttentionLayer($\mathbf{X}^c_{ms}$), TemporalAttentionLayer($(\mathbf{X}^c_{ms})^\top$) by Eqs. 19–20;
- $\mathbf{H}^c_{ca} \leftarrow$ CrossAttention($\mathbf{H}^c_s, \mathbf{H}^c_t$) by Eq. 21;
- $\mathbf{H}^c_{gated} \leftarrow$ GatedFusion($\mathbf{H}^c_s, \mathbf{H}^c_{ca}$) by Eqs. 22–23;
- $\mathbf{H}^c_{weighted} \leftarrow$ WeightedAdd($\mathbf{X}^c_{ms}, \mathbf{H}^c_{gated}, \mathbf{H}^c_t$) by Eq.25;
- $\mathbf{H}^c_{fusion} \leftarrow$ LayerNorm($\mathbf{H}^c_{weighted}$ + MLP($[\mathbf{X}^c_{ms} \| \mathbf{H}^c_{gated} \| \mathbf{H}^c_t]$)) by Eqs. 24–26; `// Final fused representation`
- $\mathbf{A}^c \leftarrow$ SelfAttention($\mathbf{H}^c_{fusion}$) by Eqs. 3–4 ; `// Generate dynamic graph adjacency matrix`
- $\mathbf{C}^c \leftarrow$ MLP(GRU(GCN($\mathbf{T}^c_{msc}, \mathbf{A}^c$))) by Eq. 27 ; `// Graph-enhanced feature extraction`
- **return** $\mathbf{C}^c$

for samples whose graph structures are inconsistent with the norm. Stability is achieved through several mechanisms. First, the modulation factors are explicitly bounded: $\alpha^c \in (1, 1+\gamma]$ and $\beta^c \in [1-\lambda, 1]$. Consequently, the overall scaling factor Ω is confined to $[1-\lambda, 1+\gamma]$, preventing it from exploding or collapsing gradients. This bounded design also mitigates the risk of over-weighting samples with drastic adjacency changes. And the dual-factor design (α for consistency, β for amplification) creates a balanced objective that stabilizes the loss landscape by rewarding stability in normal patterns while penalizing anomalous discrepancies.

To maintain structural consistency between input space and latent representations, we employ the soft contrastive learning (SCL) loss based on similarity distributions proposed by [15]. In traditional contrastive learning (CL), the task is formulated as a binary classification between positive pairs, denoted as $(\mathbf{x}^{c1}, \mathbf{x}^{c2}) \sim P_{\mathbf{x}^{c1},\mathbf{x}^{c2}}$ sampled from the joint distribution and labeled as $H_{c1,c2} = 1$, and negative pairs labeled as $H_{c1,c2} = 0$ from the marginal distribution. Unlike conventional contrastive learning methods that rely on hard binary labels, this approach refines the CL loss by incorporating a soft weighting mechanism based on sample similarity assessments. The label $H_{c1,c2}$ is softened into $P_{c1,c2}$ to modulate the contribution of each pair, leading to the soft contrastive learning loss:

$$\mathcal{L}_{SCL}\left(\mathbf{x}^{c1}, \{\mathbf{x}^{c2}\}_{c2=1}^{N_K}\right) = -\sum_{j=1}^{N_K} \left\{P_{c1,c2}\log(Q_{c1,c2}) + (1-P_{c1,c2})\log(1-Q_{c1,c2})\right\}, \tag{35}$$

where N_K representing the number of negative pairs, $Q_{c1,c2} = \exp\left(S(\mathbf{z}^{c1}, \mathbf{z}^{c2})\right)$ denotes the density ratio, $S(\mathbf{z}^{c1}, \mathbf{z}^{c2})$ represents the cosine similarity between the embeddings $(\mathbf{z}^{c1}, \mathbf{z}^{c2})$ of $(\mathbf{x}^{c1}, \mathbf{x}^{c2})$, and $P_{c1,c2}$ is defined based on a kernel function $\kappa(\cdot)$ as:

$$P_{c1,c2} = \begin{cases} e^{\alpha}\kappa(\mathbf{C}^{c1}_{GRU}, \mathbf{C}^{c2}_{GRU}), & \text{if } H_{c1,c2} = 1 \\ \kappa(\mathbf{C}^{c1}_{GRU}, \mathbf{C}^{c2}_{GRU}), & \text{otherwise} \end{cases} \tag{36}$$

where α controls the influence of positive prior knowledge, and $(\mathbf{C}^{c1}_{GRU}, \mathbf{C}^{c2}_{GRU})$ denote the spatio-temporal condition vector generated after the GRU.

Compared to conventional CL, the SCL loss adopts a softer optimization target, enhancing robustness to noisy augmentations and improving the generalization capability of the learned representations under data-augmented conditions.

The combined objective function of our model is:

$$\mathcal{L}_{total} = \mathcal{L}_{enhanced} + \mathcal{L}_{SCL}. \tag{37}$$

The graph structure-enhanced loss $\mathcal{L}_{enhanced}$ incorporates temporal graph similarity to emphasize structural consistency across windows, while the soft contrastive learning loss $\mathcal{L}_{SCL}$ strengthens the model's ability to distinguish nuanced differences between normal and anomalous behaviors by pulling semantically similar representations closer and pushing dissimilar ones apart. By integrating these two complementary objectives, our framework achieves a balanced optimization that enhances both the probabilistic modeling precision and the discriminative.

4.5. Pseudocode

Algorithm 1 shows how spatio-temporal conditions are generated from input windows through temporal encoding, spatio-temporal feature fusion, and graph representation learning, as implemented in the TFE, ADGG, and GEF modules.

Algorithm 2 illustrates the training and testing procedure of the STF²-DGL model, where selected features are obtained through normalizing flow-based density estimation.

Algorithm 2: STF²-DGL Training and Testing

Input: Train windows Data: $\mathcal{D}_1 = \{\mathbf{x}_i^c\}_{i=1}^M$, Test windows Data: $\mathcal{D}_2 = \{\mathbf{x}_j^c\}_{j=1}^N$, Learning rate: α, Epochs: E, Batch size: B, Network parameters, Hyperparameters: γ, λ, δ

Output: The prediction of STF²-DGL: $\mathbf{z}^c$

Initialization;
for $i \leftarrow 0$ **to** E **do**
  for $b \leftarrow 0$ **to** $[M/B]$ **do**
    $\mathbf{X}^c \leftarrow \text{Sampling}(\mathcal{D}_1, b)$; // Sample a train batch data
    $\mathbf{X}^{c,+} \leftarrow \text{Augment}(\mathbf{X}^c)$ by Eq. 2 ; // Data augmentation
    $\mathbf{C}^c \leftarrow \texttt{Forward}(\mathbf{X}^c + \mathbf{X}^{c,+})$;
    $\mathbf{z}^c \leftarrow f_\theta(\mathbf{C}^c)$ by Eq. 5 ; // Make density estimates
    $\mathcal{L}_{base}^c \leftarrow \mathcal{L}_{MLE}(\mathbf{z}^c)$ by Eq. 28 ; // Cal. MLE Loss
    $\mathcal{L}_{enhanced} \leftarrow \mathcal{L}_{base}^c \cdot \Omega(\mathbf{A}^c, \mathcal{L}_{base}^c)$ by Eqs. 32–34 ; // Cal. Graph Enhanced Loss
    $\mathcal{L}_{\text{SCL}} \leftarrow \mathcal{L}_{\text{SCL}}(\mathbf{z}^c, \mathbf{C}^c)$ by Eq. 35 ; // Cal. SCL Loss
    $\mathcal{L}_{total} \leftarrow \mathcal{L}_{enhanced} + \mathcal{L}_{SCL}$ by Eq. 37 ; // Cal. total Loss
    Update model parameters using gradient descent;
  for $b \leftarrow 0$ **to** $[N/B]$ **do**
    $\mathbf{X}^c \leftarrow \text{Sampling}(\mathcal{D}_2, b)$; // Sample a test batch data
    $\mathbf{C}^c \leftarrow \texttt{Forward}(\mathbf{X}^c)$;
    $\mathbf{z}^c \leftarrow f_\theta(\mathbf{C}^c)$ by Eq. 5 ; // Make density estimates
return $\mathbf{z}^c$

Table 1
The settings of five public datasets.

Dataset	Channel number	Training set	Train Anom. (%)	Testing set	Test Anom. (%)
SWaT	51	269951	17.7	89984	5.2
WADI	123	103680	6.4	69121	4.6
PSM	25	52704	23.1	35137	34.6
MSL	55	44237	14.7	29492	4.3
SMD	38	425052	4.2	283368	4.1

5. EXPERIMENTS

5.1. Experiment Setup

5.1.1. Dataset and Implementation Details

Datasets Information: To enable a comprehensive comparison with baseline methods, we evaluate STF²-DGL on five publicly available datasets that are used across the main baselines: SWaT [44], WADI [45], PSM [46], MSL [34], and SMD [1]. These five datasets, as shown in Table 1, cover a diverse range of real-world scenarios—including industrial control systems (SWaT, WADI), server performance monitoring (PSM, SMD), and space exploration telemetry (MSL)—and are widely used benchmarks for evaluating anomaly detection methods in MTS.

Implementation Details: We follow the dataset configurations established in the GANF framework [5] and its variants MTGFlow [6] and USD [15]. Specifically, for the SWaT dataset, the original testing data is partitioned into 60% for training, 20% for validation, and 20% for testing. For all other datasets, 60% of the data is used for training and the remaining 40% for testing.

All datasets undergo preprocessing to remove missing values and are normalized to the range [0, 1]. The time series are then segmented into fixed-length sequences using a sliding window approach, where the initial window size is set to 60 with a fixed stride of 10. Subsequently, the optimal window size for each dataset is determined based on hyperparameter tuning experiments, as shown in Table 7. These sequences are subsequently used as inputs for model training and inference. The hyperparameters are set as follows: $\delta = 5.0$ in Eq. 30, $\gamma = 0.2$ in Eq. 31 and $\lambda = 0.5$ in Eq. 32. All experiments are run for 400 epochs and executed using PyTorch 2.4.1 on an NVIDIA GeForce RTX 4090 24GB GPU.

5.1.2. Evaluation Metric and Baselines Methods

Evaluation Metric: Following standard practice, our unsupervised anomaly detection framework STF²-DGL performs detection at the window level. Each window is labeled as anomalous if it contains at least one anomalous time point. To assess model performance, we employ two widely adopted evaluation metrics. One is the Area Under the Receiver Operating Characteristic Curve (AUROC) which evaluates the overall ability of the model to distinguish between normal and anomalous instances across all classification thresholds. A higher AUROC indicates better separability between classes. The other is the Area Under the Precision-Recall Curve (AUPRC) which focuses on the performance of the model under class imbalance by measuring the trade-off between precision and recall. It is especially useful for assessing how well the model detects true anomalies with minimal false positives. Together, these metrics jointly provide a comprehensive assessment of the model's detection performance under different threshold settings and operational conditions.

Table 2
Anomaly detection performance of Area Under the Receiver Operating Characteristic (AUROC) on five public datasets. The best results are highlighted in bold.

Methods	Journal-Year	Area Under the Receiver Operating Characteristic Curve (AUROC) SWaT	WADI	PSM	MSL	SMD	Average	Rank
DeepSAD [47]	ICLR, 2020	75.4 (±2.4)	85.4 (±2.7)	73.2 (±3.3)	61.6 (±0.6)	85.9 (±11.1)	76.3	8
DROCC [48]	ICML, 2020	72.6 (±3.8)	75.6 (±1.6)	74.3 (±2.0)	53.4 (±1.6)	76.7 (±8.7)	70.5	9
USAD [19]	KDD, 2020	78.8 (±1.0)	86.1 (±0.9)	78.0 (±2.0)	57.0 (±0.1)	86.9 (±11.7)	77.4	7
DAGMM [49]	ICLR, 2018	72.8 (±3.0)	77.2 (±0.9)	64.6 (±2.6)	56.5 (±2.6)	78.0 (±9.2)	69.8	10
DCdetector [36]	KDD, 2023	51.1 (±1.0)	55.9 (±0.4)	50.6 (±2.5)	44.8 (±1.4)	25.3 (±2.1)	52.7	11
TFMAE [37]	ICDE, 2024	56.1 (±1.5)	45.54 (±1.6)	33.9 (±1.9)	47.2 (±0.9)	30.5 (±1.4)	43.7	12
GANF [5]	ICLR, 2022	79.8 (±0.7)	90.3 (±1.0)	81.8 (±1.5)	64.5 (±1.9)	89.2 (±7.8)	81.1	6
MTGFlow [20]	AAAI, 2023	84.8 (±1.5)	91.9 (±1.1)	85.7 (±1.5)	67.2 (±1.7)	91.3 (±7.6)	84.2	5
MTGFlow Cluster [6]	TKDE, 2024	83.1 (±1.3)	91.8 (±0.4)	87.1 (±2.4)	68.2 (±2.6)	–	–	–
MADGA [16]	ICDM, 2024	88.9 (±0.7)	92.0 (±1.0)	87.2 (±1.4)	68.1 (±1.6)	92.1 (±6.7)	85.7	4
Graph-MoE [7]	AAAI, 2025	87.2 (±1.3)	94.2 (±0.8)	88.0 (±0.7)	72.1 (±1.1)	93.3 (±5.6)	87.0	3
USD [15]	ICASSP, 2025	90.2 (±0.9)	94.1 (±0.4)	88.9 (±2.4)	**74.2** (±2.0)	94.2 (±6.5)	88.3	2
STF²-DGL	OURS	**92.1** (±1.1)	**95.2** (±0.9)	**91.7** (±2.1)	73.1(±1.2)	**94.5** (±5.0)	**89.3**	**1**

Baselines: We benchmark our proposed method, STF²-DGL, against a range of SOTA methods, including both semi-supervised and unsupervised approaches.

- DeepSAD [47] (semi-supervised): Utilizes a small amount of labeled data to improve anomaly detection by modeling latent entropy differences between normal and anomalous instances.
- DROCC [48] (unsupervised): Constructs robust representations of normal samples on a local manifold, treating deviations as anomalies.
- USAD [19] (unsupervised): Uses adversarially trained autoencoders to detect anomalies based on reconstruction performance.
- DAGMM [49] (unsupervised): Integrates autoencoder features with a Gaussian mixture model for joint reconstruction and density estimation.
- DCdetector [36] (unsupervised): Constructs a contrastive architecture with a dual-branch, dual-attention structure where the two branches share network weights.
- TFMAE [37] (unsupervised): Uses two autoencoders with temporal and frequency masking, trained with a contrastive objective to detect anomalies in time series.
- GANF [5] (unsupervised): Enhances normalizing flows with a Bayesian structure to identify anomalies in low-density regions.
- MTGFlow [20] (unsupervised): Learns dynamic graph structures and variable-aware flows for fine-grained density-based anomaly detection.
- MTGFlow-Cluster [6] (unsupervised): Builds on MTGFlow with a clustering mechanism to improve density estimation and anomaly identification.
- MADGA [16] (unsupervised): Redefines anomaly detection as a graph alignment problem and identifying deviations through graph alignment using Wasserstein-based distances.
- Graph-MoE [7] (unsupervised): Utilizes hierarchical representations from multiple GNN layers and integrates them through the memory-augmented routers.
- USD [15] (unsupervised): Combines data augmentation with soft contrastive learning to improve the model's sensitivity to diverse normal and abnormal patterns beyond Gaussian assumptions.

5.2. Experiment Results

5.2.1. Comparative Results

Tables 2, 3 and Fig. 5 present the anomaly detection performance of our proposed STF²-DGL method compared to baseline methods, evaluated using AUROC and AUPRC metrics on five widely used public datasets. The results reported consist of averages and variances from five experimental runs [6, 7, 15]. Note that the higher standard deviation observed in the SMD results from its inclusion of 28 heterogeneous sub-datasets.

Our STF²-DGL achieves superior performance across most benchmarks. As shown in Table 2, our model outperforms state-of-the-art baselines in terms of AUROC on four datasets (SWaT, WADI, PSM, and SMD). On average, STF²-DGL achieves an AUROC of 89.3%, outperforming MTGFlow and USD by 5.1% and 1.0%, respectively. These improvements highlight the importance of learning and leveraging fine-grained graph structural differences between normal and anomalous data for accurate anomaly detection. Compared to MADGA and Graph-MoE, STF²-DGL achieves average improvements of 3.6% and 2.3%, respectively, demonstrating that effectively modeling spatio-temporal dependencies within multivariate time series is critical for robust performance across complex industrial scenarios. On the other hand, contrastive-based methods [36, 37] exhibit suboptimal performance on AUROC and AUPRC metrics. This finding is consistent with the research by Gungor et al. [50]. Although our model underperforms on the MSL dataset, previous studies [51, 52] have identified potential flaws within this dataset, noting trivialities, labeling errors, and unrealistic anomaly densities, which could undermine its validity. Specifically, the unrealistic anomaly density in the MSL dataset may bias our model when learning the normal graph structure, as

Table 3
Anomaly detection performance* of Area Under the Precision-Recall Curve (AUPRC) on five public datasets. The best results are highlighted in bold.

Methods	Journal-Year	Area Under the Precision-Recall Curve (AUPRC) SWaT	WADI	PSM	MSL	SMD	Average	Rank
DeepSAD [47]	ICLR, 2020	45.7 (±12.3)	23.1 (±6.3)	66.7 (±10.8)	26.3 (±1.7)	61.8 (±21.4)	44.7	6
DROCC [48]	ICML, 2020	26.4 (±9.8)	16.7 (±7.1)	60.7 (±11.4)	13.2 (±0.9)	20.7 (±15.6)	27.5	9
USAD [19]	KDD, 2020	18.8 (±0.6)	19.8 (±0.5)	57.9 (±3.6)	31.3 (±0.0)	68.4 (±19.9)	39.2	8
DAGMM [49]	ICLR, 2018	35.9 (±11.5)	35.7 (±5.8)	62.1 (±7.4)	25.5 (±3.4)	62.9 (±18.6)	44.4	7
DCdetector [36]	KDD, 2023	15.0 (±3.4)	10.2 (±5.1)	25.9 (±3.4)	11.3 (±1.2)	9.7 (±5.6)	14.4	11
TFMAE [37]	ICDE, 2024	16.1 (±1.8)	18.9 (±9.2)	21.6 (±5.9)	47.2 (±0.9)	15.8 (±6.1)	23.9	10
GANF [5]	ICLR, 2022	21.6 (±1.8)	39.0 (±3.1)	73.8 (±4.7)	31.1 (±0.2)	64.4 (±21.4)	46.0	5
MTGFlow [20]	AAAI, 2023	38.3 (±6.1)	42.2 (±4.9)	76.2 (±4.8)	31.1 (±2.6)	64.2 (±21.5)	50.5	4
MTGFlow Cluster [6]	TKDE, 2024	37.1 (±5.9)	40.2 (±3.5)	77.9 (±5.6)	29.4 (±3.1)	–	–	–
MADGA [16]	ICDM, 2024	37.5 (±8.5)	42.0 (±5.7)	78.1 (±4.3)	30.9 (±1.6)	75.1 (±16.7)	53.3	3
USD [15]	ICASSP, 2025	53.4 (±10.5)	45.5 (±4.9)	83.2 (±3.9)	31.9(±3.8)	73.6 (±19.7)	57.5	2
STF²-DGL	OURS	**55.4** (±5.9)	**51.4** (±3.6)	**84.9** (±3.3)	**32.6** (±2.2)	**76.8** (±16.2)	**60.2**	**1**

*Graph-MoE [7] code is not available, and its AUPRC results have not been published.

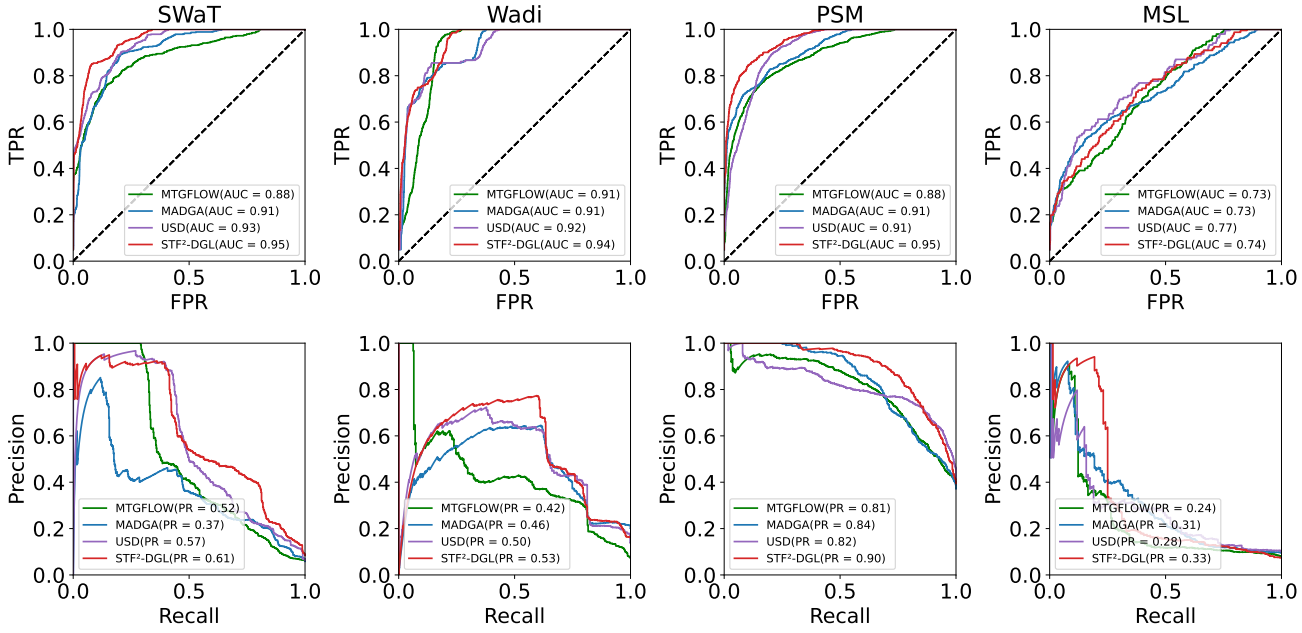


Figure 5: ROC and PR curves for the SWaT, WADI, PSM, and MSL datasets. The ROC curves depict the trade-off between true positive rate and false positive rate, while the PR curves illustrate the balance between precision and recall. The efficacy of the proposed methodology is demonstrated by way of a comparison with state-of-the-art approaches, thus highlighting the merits of the method.

our graph augmentation loss function is sensitive to minor changes in the graph's topology.

Our improvements are more pronounced with the AUPRC metric, as detailed in Table 3. Compared to state-of-the-art methods, STF²-DGL shows an average AUPRC increase of 2.7% across five datasets, with notable improvements of 2.0% on SWaT, 5.9% on WADI, and 3.2% on SMD. These enhancements clearly demonstrate the method's superior capability to identify anomalies with greater precision and recall, underscoring its effectiveness in complex anomaly detection tasks. Another important aspect demonstrated in Tables 3 is the stability of STF²-DGL. The model exhibits relatively lower variance across multiple datasets, reflected by the standard deviations in parentheses. This stability implies that STF²-DGL is less susceptible to random initialization, thereby ensuring consistent and reliable anomaly detection performance across diverse datasets and experimental conditions. Such robustness is crucial for practical applications, where reliable and repeatable results significantly enhance the usability of the detection system.

Table 4
Model efficiency analysis: Inference time, model size, FLOPs, and AUROC/AUPRC on the SWaT dataset.

Metric	MTGFlow	MADGA	USD	STF²-DGL
Inference Time (ms)	32.87±1.55	22.96±23.78	13.05±9.26	24.13±7.27
Model Size (M)	0.08	0.09	60.66	60.81
FLOPs (B)	12.40	12.40	43.39	55.91
AUROC	84.8 (±1.5)	88.9 (±0.7)	90.2 (±0.9)	92.1 (±1.1)
AUPRC	38.3 (±6.1)	37.5 (±8.5)	53.4 (±10.5)	55.4 (±5.9)

In addition, we conducted an efficiency analysis in inference time per batch, model size, and FLOPs between STF²-DGL and three state-of-the-art baselines. All methods

Table 5
Ablation study: AUROC comparison across four dataset. The best results are in bold.

Method	Affected Modules	SWaT	WADI	PSM	MSL	Average	Average Change
STF²-DGL	-	**92.1 (±1.1)**	**95.2 (±0.9)**	**91.7 (±2.1)**	**73.1 (±1.2)**	**88.0**	-
w/o. Variable-wise Att	TFE	90.9 (±1.9)	93.8 (±1.5)	90.1 (±2.2)	71.9 (±1.8)	86.7	−1.3
w. Mamba	TFE	83.6 (±3.4)	92.5 (±1.7)	88.9 (±2.5)	70.5 (±1.5)	83.9	−4.1
w/o. LSTM	TFE	90.1 (±1.5)	92.1 (±1.1)	88.5 (±1.3)	71.6 (±1.2)	85.6	−2.4
w/o. Multi-scale Conv	TFE & ADGG	87.2 (±2.9)	91.4 (±1.0)	88.2 (±1.2)	71.8 (±0.9)	84.7	−3.3
w/o. Temporal Exponential Att	ADGG	85.4 (±2.5)	91.5 (±1.7)	87.5 (±2.9)	70.1 (±0.8)	83.6	−4.4
w/o. View-Fusion	ADGG	89.4 (±3.4)	93.2 (±1.1)	89.1 (±1.7)	70.3 (±1.6)	85.5	−2.5
w/o. Graph	ADGG & GEF	82.1 (±0.9)	90.5 (±0.8)	88.5 (±1.9)	69.5 (±0.9)	82.7	−5.3
w. sDAG	ADGG & GEF	83.5 (±1.3)	90.8 (±1.5)	88.9 (±2.0)	71.2 (±1.2)	83.6	−4.4
w/o. GEF	GEF	89.9 (±1.1)	93.0 (±1.0)	89.1 (±2.7)	71.5 (±2.1)	85.9	−2.1
w/o. GRU	GEF	91.7 (±2.4)	94.0 (±1.4)	90.8 (±1.6)	72.0 (±1.3)	87.1	−0.9
w. STGCN	GEF	91.9 (±1.4)	93.6 (±1.5)	90.5 (±2.5)	72.0 (±1.5)	87.0	−1.0
w. GAT	GEF	88.2 (±2.7)	91.2 (±2.1)	87.9 (±2.5)	70.5 (±2.1)	84.5	−3.5
w/o. $\mathcal{L}_{enhanced}$	GEF	89.7 (±1.6)	93.5 (±1.1)	90.5 (±1.3)	72.4 (±1.4)	86.5	−1.5
w. WD & GWD	GEF	90.3 (±2.6)	92.9 (±2.0)	90.2 (±1.7)	70.1 (±1.8)	85.9	−2.1

were trained on the SWaT dataset under identical parameter settings. The model efficiency analysis is presented in the table. Table 4 indicates that the STF²-DGL model has an inference time of 24.13ms, a model size of 60.81M, and 55.91B FLOPs. While this represents a slight increase in computational overhead compared to the state-of-the-art USD model, it achieves average improvements of 2.1% in AUROC and 3.7% in AUPRC. We consider this as a reasonable trade-off between accuracy and efficiency.

5.2.2. Ablation Study

We conducted an ablation study to evaluate the contributions of each component within our STF²-DGL model, as illustrated in Table 5. Specifically, we assessed the model's performance under different configurations:

- **w/o. Variable-wise Att**: Removes the variable-wise Transformer blocks from the TFE module, i.e. retaining only LSTM and multi-scale convolution for temporal feature extraction.
- **w. Mamba**: Replaces the entire TFE module with a Mamba network to evaluate its effectiveness in modeling temporal dynamics.
- **w/o. LSTM**: Removes the LSTM module from the TFE module.
- **w/o. Multi-scale Conv**: Eliminates all multi-scale convolution layers from both the TFE and ADGG modules to assess the role of multi-scale temporal modeling.
- **w/o. Temporal Exponential Att**: Uses only spatial attention in the ADGG module, aligning the graph construction strategy with the MTGFlow [6] method and omitting the temporal exponential attention.
- **w/o. View-Fusion**: Replaces the spatio-temporal fusion module with simple feature concatenation to evaluate the benefit of cross-view integration.
- **w/o. Graph**: STF²-DGL without any graph module. The standardized flow conditions are generated directly from the temporal modeling modules (TFE and GRU).
- **w. sDAG**: Uses a static Directed Acyclic Graph (sDAG) rather than dynamic graphs.
- **w/o. GEF**: Substitutes GCN+GRU and Graph-aware Loss with a GNN backbone and standard SCL Loss.
- **w/o. GRU**: Removes the GRU layer from the GEF module to examine the impact of sequential dependency modeling on global variable features.
- **w. STGCN**: Substituting GCN and GRU modules with a STGCN [53] module.
- **w. GAT**: Substitutes GCN and GRU modules with a GAT [42] module.
- **w/o. $\mathcal{L}_{enhanced}$**: Excludes the graph structure-enhanced factor from the loss function, reverting to the loss design used in USD for comparison.
- **w. WD & GWD**: Replaces the Frobenius norm-based graph similarity computation with the Wasserstein distance ($\mathcal{D}_{wd}$) for node embeddings and the Gromov-Wasserstein distance ($\mathcal{D}_{gwd}$) for adjacency matrices, following the MADGA [16] method.

As can be seen from the results, removing the variable-wise Transformer blocks (w/o. variable-wise Transformer), removing the LSTM module (w/o. LSTM), or replacing the entire TFE module with Mamba (w. Mamba) led to a significant performance drop, confirming that our design better captures temporal dependencies. Furthermore, the flow-only model (w/o. Graph) outperforms non-flow baselines, such as DROCC, USAD and DAGMM (as shown in Table 2). This demonstrates that the normalizing flow is an effective method for anomaly detection.

Furthermore, removing multi-scale convolutions (w/o. Multi-scale Conv) or temporal exponential attention (w/o. Temporal Exponential Att) also caused a notable decline in performance, highlighting the importance of multi-scale modeling and the necessity of temporal attention in capturing time-evolving graph structures. Substituting the View-Fusion module with simple concatenation (w/o. View-Fusion) further degraded performance, validating the effectiveness of our cross-view integration. Removing the GRU layer from the global feature encoder (w/o. GRU) slightly reduced performance, suggesting that sequential modeling remains beneficial for capturing variable-level dynamics. In addition, the GEF module outperformed STGCN (w. STGCN) and

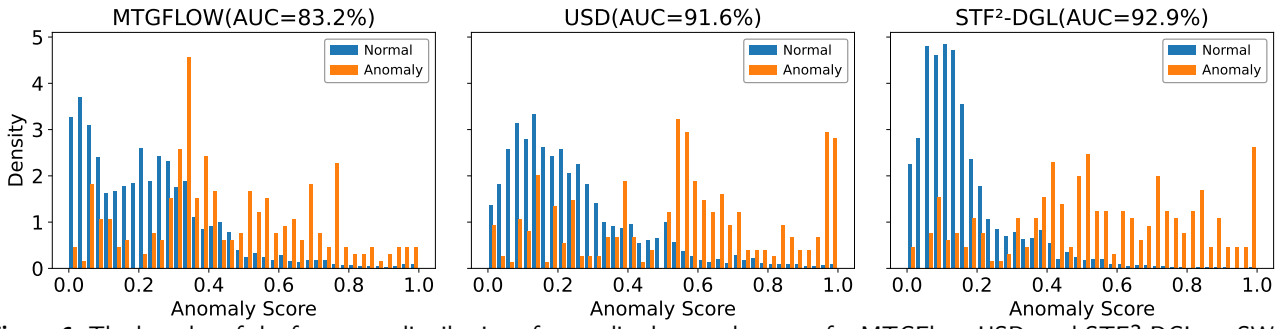


Figure 6: The bar plot of the frequency distribution of normalized anomaly scores for MTGFlow, USD, and STF²-DGL on SWaT dataset. The x-axis denotes the anomaly score, and the y-axis indicates the frequency or density of the corresponding score.This demonstrates that STF²-DGL achieves better discrimination between normal and abnormal states, consistent with its highest AUC of 92.9%.

Table 6
Incremental Component Analysis: Performance comparison on the SWaT dataset by independently integrating each core module of STF²-DGL into the baseline method MTGFlow.

Variants	Graph Learning(ours)	GEF	Graph-aware Loss	AUROC
MTGFlow	×	×	×	84.8±1.5
MTGFlow+TFE	✓	×	×	87.5±2.4
MTGFlow+ADGG	✓	×	×	88.0±1.3
MTGFlow+TFE+ADGG	✓	×	×	90.1±1.8
MTGFlow+STGNN	×	✓	×	86.9±2.0
MTGFlow+$\mathcal{L}_{enhanced}$	×	×	✓	85.6±2.5
MTGFlow+STGNN+$\mathcal{L}_{enhanced}$	×	✓	✓	88.3±2.2
STF²-DGL	✓	✓	✓	**92.1±1.1**

GAT (w. GAT) modules, whose complexity led to performance degradation. This confirms that our strategy of separating spatial (GCN) and temporal (GRU) processing generates a more stable conditioning vector, avoiding the risk of overfitting.

On the loss design side, excluding the graph-structure-enhanced term (w/o. $\mathcal{L}_{enhanced}$) or replacing our Frobenius norm-based graph similarity with Wasserstein-based distances (w. WD & GWD) led to noticeable performance drops, indicating that our similarity formulation achieves a better balance between structural fidelity and optimization stability.

To validate the effectiveness of our core modules, we integrated them separately into the existing flow-GCN-based baseline, MTGFlow, as shown in Table 6. TFE and ADGG brought the most significant performance improvements when used individually, with their combination yielding an even more pronounced effect. Similarly, the GEF and the Graph-aware Loss also showed positive effects. These results demonstrate that each of our proposed modules is effective, either independently or in combination, and reflect the modules' ability for generalization.

5.2.3. Qualitative Results

Normalized Anomaly Scores: Fig. 6 presents the histogram of anomaly scores produced by MTGFlow, USD, and STF²-DGL on the SWaT dataset. The x-axis denotes the normalized anomaly scores, while the y-axis represents the corresponding frequency. Compared to MTGFlow and USD, STF²-DGL produces a sharper separation between normal and anomalous samples, with normal scores concentrated near 0 and anomalies distributed towards higher scores. In contrast, the distributions from MTGFlow and USD exhibit greater overlap between normal and anomalous samples, leading to ambiguity in classification thresholds. This distributional separation highlights STF²-DGL's superior ability to map structurally consistent normal patterns to compact low-density regions in the output space, while effectively amplifying irregular signals caused by anomalies. The contrast in score dispersion directly supports the quantitative result that STF²-DGL achieves the highest AUC on the SWaT dataset, confirming its enhanced discriminative capacity in distinguishing normal from abnormal behaviors.

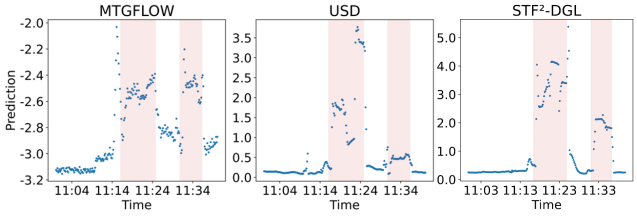


Figure 7: Point plot of anomaly prediction outputs from MTGFlow, USD, and the proposed STF²-DGL. The x-axis denotes the time index, and the y-axis represents the log-likelihood scores. Ground truth anomalies are highlighted with a red background. Compared to MTGFlow and USD, STF²-DGL demonstrates more accurate anomaly detection, with clearer separation between normal and abnormal regions.

Point-wise Anomaly Prediction: Fig. 7 presents the point-wise anomaly prediction outputs from MTGFlow, USD, and the proposed STF²-DGL model. The x-axis denotes the time index, and the y-axis represents the corresponding log-likelihood scores, with ground truth anomalies highlighted in red. As illustrated in this figure, STF²-DGL yields a more distinct separation between normal and abnormal regions compared to the baselines. MTGFlow exhibits high fluctuation across the entire sequence, and USD shows moderate anomaly localization but still suffers from noisy responses. In contrast, STF²-DGL produces consistently low log-likelihood values during anomalous intervals, while maintaining stable high confidence in normal regions.

Dynamic Adjacency Matrices: Fig. 8 visualizes the dynamic adjacency matrices generated by STF²-DGL across adjacent time steps on the PSM dataset. The figure is divided

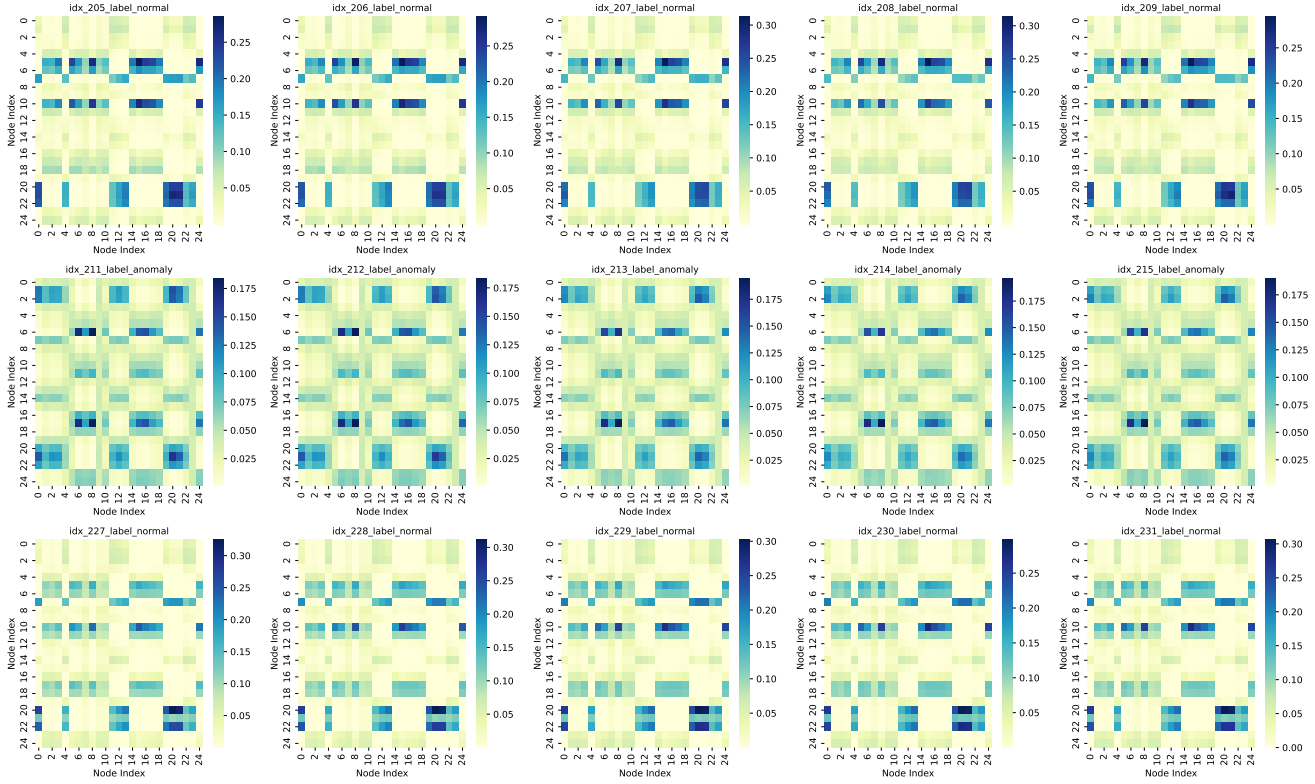


Figure 8: Visualization of the learned dynamic adjacency matrices from STF²-DGL across adjacent time steps on the PSM dataset. The first row shows the 5 time steps before the anomaly, the second row corresponds to the anomaly, and the third row shows the 5 time steps after. The structural differences between normal and anomalous samples are clearly observed, indicating that STF²-DGL effectively captures the relational shifts useful for distinguishing anomalies.

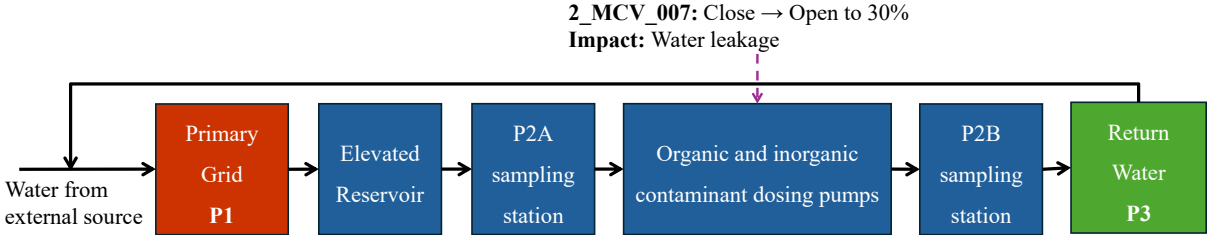


Figure 9: Three stages "P1, P2, P3" and attack #11 in the WADI dataset. Solid arrows indicate flow of water and sequence of processes.

into three rows: the first row corresponds to the 5 time steps preceding the anomaly, the second row contains the anomaly period, and the third row shows the 5 time steps following the anomaly. As observed, the structure of the learned graphs changes significantly during the anomaly period. Furthermore, the adjacency matrices show high structural consistency during the normal pre- and post-anomaly periods, which demonstrates the model's stability in learning general graph patterns for normal data without overreacting to minor fluctuations. These distinct topological shifts indicate that STF²-DGL effectively captures relational perturbations induced by anomalies. The recovery of graph patterns after the anomaly also demonstrates the model's temporal sensitivity and adaptability. This result highlights STF²-DGL's ability to model time-dependent inter-variable dependencies, providing strong interpretability and reliability in unsupervised anomaly detection.

Interpretable Anomaly Detection: A case study of a real-world attack on the WADI dataset [45] demonstrates how the learned graphs capture semantically meaningful relationships among variables. Fig. 9 shows three distinct control processes labeled P1 through P3 of WADI, each controlled by its own set of Programmable Logic Controllers (PLCs). The details of Attack #11 are as follows: Maliciously open '2_MCV_007' in order to produce water leakage before water reaches consumers, and '2_PIT_002' will feedback a decrease in reading to less than 0.1 bar. According to the visualization of graph node relationships shown in Fig. 10, the learned graph of STF²-DGL accurately reflects this physical reality. The node #61 corresponding to the attack source, '2_MCV_007', is correctly identified as the central hub of the anomaly. Crucially, its most heavily weighted edge connects directly to the node #89, '2_PIT_002', mirroring the real-world cause-and-effect

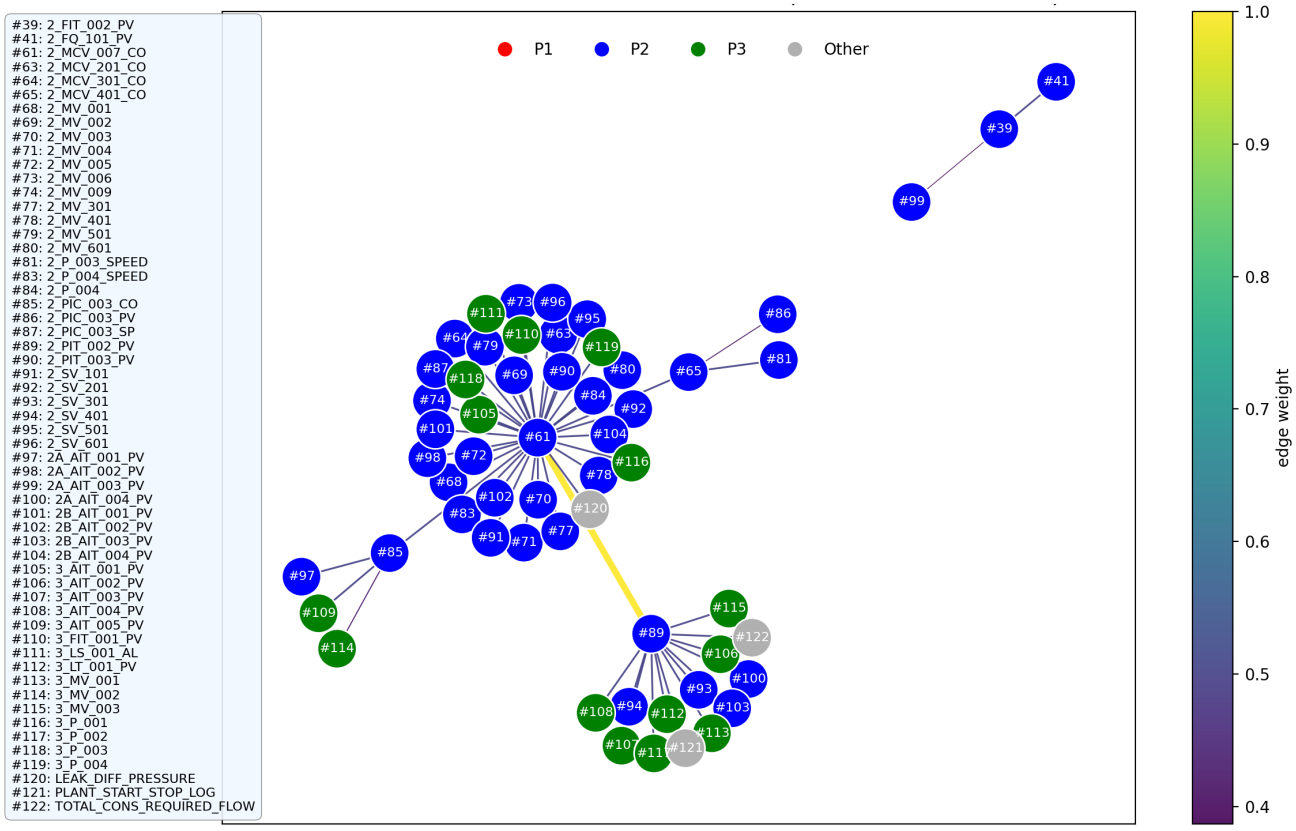


Figure 10: Anomaly interpretability analysis of attack #11 in the WADI dataset. Nodes of different colors represent different stages. The color of the connection lines indicates the degree of relationship between variables. Edges with weights below 0.4 are ignored.

relationship. The graph also correctly isolates unrelated factors by showing only weak correlations to the preceding process (P1). This clear correspondence between the graph topology and the physical events confirms that STF²-DGL learns coherent dependencies, enabling interpretable fault localization.

5.2.4. Hyperparameter Analysis

Table 7 presents the AUROC performance under varying window sizes and Transformer block depths across four datasets. We observe that performance is sensitive to both hyperparameters, but their optimal values vary by dataset characteristics. These results highlight that both the temporal scale of the data and the model depth play a critical role in overall model performance.

In addition to temporal modeling, we further investigate the impact of structural modeling hyperparameters within the graph-aware loss function. Specifically, we analyze the sensitivity of the graph-aware loss with respect to the scale hyperparameter δ and the weighting coefficients γ and λ, as shown in Tables 8 and 9. The parameter δ controls the sensitivity of the graph similarity function to structural variations by scaling the Frobenius norm between adjacency matrices. A smaller δ underemphasizes subtle topological changes, potentially leading to missed anomalies, while an excessively large δ may amplify noise and result in unstable

Table 7
Parameters Analysis: window size and number of blocks. AUROC comparison on four datasets. The best results are in bold.

Dataset	Window Size	blocks=1	blocks=2	blocks=3
SWaT	40	**92.1 (±1.1)**	87.8(±5.1)	87.2(±2.3)
	60	91.5 (±1.4)	88.3(±2.8)	82.5(±3.7)
	80	90.7(±1.3)	85.1(±3.9)	81.4(±3.5)
WADI	40	92.4(±1.8)	89.9(±1.1)	91.6(±1.2)
	60	92.5(±2.1)	90.9(±2.3)	**95.2(±0.9)**
	80	85.9(±0.9)	92.1(±1.5)	94.2(±1.5)
PSM	40	90.5(±3.1)	90.2(±1.4)	87.3(±2.6)
	60	**91.7(±2.1)**	90.1(±3.6)	88.2(±3.2)
	80	85.9(±2.4)	89.7(±1.7)	85.5(±3.0)
MSL	40	72.5(±2.3)	71.4(±0.9)	71.2(±1.1)
	60	**73.1(±1.2)**	72.2(±1.5)	71.0(±1.1)
	80	71.8(±1.9)	72.0(±1.7)	71.5(±0.5)

detection. The results show that the model achieves optimal performance at $\delta = 5.0$, indicating that a moderate scaling factor strikes an effective balance between capturing meaningful deviations in dynamic graph structure and maintaining robustness to minor fluctuations. For γ and λ, the best AUROC is obtained when $\gamma = 0.2$ and $\lambda = 0.5$, demonstrating that a moderate balance between temporal consistency and anomaly amplification is beneficial. Overall, the graph

Table 8
Parameters Analysis: scale hyperparameter δ in Eq.30. AUROC for different δ on the SWaT dataset.

δ	1.0	2.0	3.0	4.0	5.0	6.0
AUROC	91.5(±0.8)	91.2(±1.5)	91.5(±1.3)	91.9(±0.9)	**92.1(±1.1)**	91.5(±1.8)

Table 9
Parameters Analysis: hyperparameter γ, λ in Eq.31 and Eq.32. AUROC for different δ on the SWaT dataset.

γ \ λ	0.1	0.2	0.3	0.4	0.5	0.6
0.1	88.9(±1.8)	88.6(±0.9)	91.6(±2.2)	91.1(±2.0)	91.1(±1.7)	90.4(±1.5)
0.2	88.7(±2.3)	90.0(±2.5)	91.5(±0.9)	92.0(±0.9)	**92.1(±1.1)**	90.4(±1.8)
0.3	89.0(±2.0)	89.4(±1.3)	88.5(±2.9)	91.1(±2.6)	91.9(±2.3)	88.6(±0.5)
0.4	91.0(±2.4)	91.5(±2.6)	90.1(±0.8)	90.6(±2.8)	90.1(±1.0)	89.5(±0.9)
0.5	91.9(±3.0)	91.8(±2.1)	90.0(±0.8)	89.5(±3.2)	91.3(±1.1)	89.0(±2.0)
0.6	88.3(±2.5)	91.3(±3.1)	90.2(±0.8)	90.7(±3.1)	88.7(±1.9)	90.5(±3.1)

loss parameters have a controllable influence, and STF²-DGL achieves stable performance across a broad range of settings.

6. CONCLUSION

In this work, we have presented **STF²-DGL**, a flow-based framework for unsupervised MTS anomaly detection that integrates spatio-temporal feature fusion with adaptive dynamic graph generation. Extensive experiments conducted on five public datasets demonstrate the superiority of STF²-DGL over ten baseline methods. The strong empirical performance of STF²-DGL can be attributed to its sophisticated modeling of spatio-temporal correlations, adaptive structural graph learning, and its ability to precisely distinguish anomalies through enhanced graph-informed representations. To address our model's high computational complexity and improve robustness, future work will integrate graph similarity assessment directly into the GNN, exploring graph kernels and learnable function mappings, which can evaluate graph similarity more efficiently. The ultimate goal is to develop a simpler, more scalable framework without compromising the effective capture of these crucial correlations, enhancing its utility in real-world applications.

References

[1] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, D. Pei, Robust anomaly detection for multivariate time series through stochastic recurrent neural network, Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (2019).

[2] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, M. Long, Timesnet: Temporal 2d-variation modeling for general time series analysis, ArXiv abs/2210.02186 (2022).

[3] K.-H. Lai, L. Wang, H. Chen, K. Zhou, F. Wang, H. Yang, X. Hu, Context-aware domain adaptation for time series anomaly detection, in: Proceedings of the 2023 siam international conference on data mining (SDM), SIAM, 2023, pp. 676–684.

[4] Y. Nam, S. Yoon, Y. Shin, M. Bae, H. Song, J.-G. Lee, B. S. Lee, Breaking the time-frequency granularity discrepancy in time-series anomaly detection, in: Proceedings of the ACM Web Conference (WWW), 2024, pp. 4204–4215.

[5] E. Dai, J. Chen, Graph-augmented normalizing flows for anomaly detection of multiple time series, arXiv preprint arXiv:2202.07857 (2022).

[6] Q. Zhou, S. He, H. Liu, J. Chen, W. Meng, Label-free multivariate time series anomaly detection, IEEE Transactions on Knowledge and Data Engineering 36 (2023) 3166–3179.

[7] X. Huang, W. Chen, B. Hu, Z. Mao, Graph mixture of experts and memory-augmented routers for multivariate time series anomaly detection, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 39, 2025, pp. 17476–17484.

[8] D. Salinas, V. Flunkert, J. Gasthaus, T. Januschowski, Deepar: Probabilistic forecasting with autoregressive recurrent networks, International journal of forecasting 36 (3) (2020) 1181–1191.

[9] K. Rasul, C. Seward, I. Schuster, R. Vollgraf, Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting, in: International Conference on Machine Learning, PMLR, 2021, pp. 8857–8868.

[10] S. Feng, C. Miao, K. Xu, J. Wu, P. Wu, Y. Zhang, P. Zhao, Multi-scale attention flow for probabilistic time series forecasting, IEEE Transactions on Knowledge and Data Engineering 36 (5) (2023) 2056–2068.

[11] K. Rasul, A.-S. Sheikh, I. Schuster, U. Bergmann, R. Vollgraf, Multivariate probabilistic time series forecasting via conditioned normalizing flows, arXiv preprint arXiv:2002.06103 (2020).

[12] A. Deng, B. Hooi, Graph neural network-based anomaly detection in multivariate time series, in: Proceedings of the AAAI conference on artificial intelligence, Vol. 35, 2021, pp. 4027–4035.

[13] Z. Chen, D. Chen, X. Zhang, Z. Yuan, X. Cheng, Learning graph structures with transformer for multivariate time-series anomaly detection in iot, IEEE Internet of Things Journal 9 (12) (2021) 9179–9189.

[14] H. Zhao, Y. Wang, J. Duan, C. Huang, D. Cao, Y. Tong, B. Xu, J. Bai, J. Tong, Q. Zhang, Multivariate time-series anomaly detection via graph attention network, in: 2020 IEEE International Conference on Data Mining (ICDM), IEEE, 2020, pp. 841–850.

[15] H. Liu, X. Qiu, Y. Shi, Z. Zang, Usd: Unsupervised soft contrastive learning for fault detection in multivariate time series, in: International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2025, pp. 1–5.

[16] Y. Wang, H. Sun, C. Wang, M. Zhu, J. Wang, W. Tang, Q. Qi, Z. Zhuang, J. Liao, Interdependency matters: Graph alignment for multivariate time series anomaly detection, in: 2024 IEEE International Conference on Data Mining (ICDM), IEEE Computer Society, Los Alamitos, CA, USA, 2024, pp. 869–874.

[17] N. Zucchet, A. Orvieto, Recurrent neural networks: vanishing and exploding gradients are not the end of the story, in: Advances in Neural Information Processing Systems, Vol. 37, Curran Associates, Inc., 2024, pp. 139402–139443.

[18] C. Ding, S. Sun, J. Zhao, Mst-gat: A multimodal spatial–temporal graph attention network for time series anomaly detection, Information Fusion 89 (2023) 527–536.

[19] J. Audibert, P. Michiardi, F. Guyard, S. Marti, M. A. Zuluaga, Usad: Unsupervised anomaly detection on multivariate time series, Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (2020) 3395–3404.

[20] Q. Zhou, J. Chen, H. Liu, S. He, W. Meng, Detecting multivariate time series anomalies with zero known label, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 37, 2023, pp. 4963–4971.

[21] K.-H. Lai, D. Zha, J. Xu, Y. Zhao, G. Wang, X. Hu, Revisiting time series outlier detection: Definitions and benchmarks, in: Thirty-fifth conference on neural information processing systems datasets and benchmarks track, 2021.

[22] Z. Wu, S. Pan, G. Long, J. Jiang, C. Zhang, Graph wavenet for deep spatial-temporal graph modeling, arXiv preprint arXiv:1906.00121 (2019).

[23] S. Lan, Y. Ma, W. Huang, W. Wang, H. Yang, P. Li, Dstagnn: Dynamic spatial-temporal aware graph neural network for traffic flow forecasting, in: International conference on machine learning, PMLR, 2022, pp. 11906–11917.

[24] V. M. Panaretos, Y. Zemel, Statistical aspects of wasserstein distances, Annual Review of Statistics and its Application 6 (2019) 405–431.

[25] X. Huang, W. Chen, B. Hu, Z. Mao, Graph mixture of experts and memory-augmented routers for multivariate time series anomaly detection, arXiv preprint arXiv:2412.19108 (2024).

[26] Y. Tian, R. Gao, L. Yan, D. Liu, Z. Ye, Dynamic graph learning with long and short-term for multivariate time series anomaly detection, in: 2023 IEEE 12th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Vol. 1, IEEE, 2023, pp. 1065–1070.

[27] Z. Zamanzadeh Darban, G. I. Webb, S. Pan, C. Aggarwal, M. Salehi, Deep learning for time series anomaly detection: A survey, ACM Computing Surveys 57 (1) (2024) 1–42.

[28] W. Wu, L. He, W. Lin, Y. Su, Y. Cui, C. Maple, S. Jarvis, Developing an unsupervised real-time anomaly detection scheme for time series with multi-seasonality, IEEE Transactions on Knowledge and Data Engineering 34 (9) (2020) 4147–4160.

[29] N. Ding, H. Gao, H. Bu, H. Ma, H. Si, Multivariate-time-series-driven real-time anomaly detection based on bayesian network, Sensors 18 (2018).

[30] K.-H. Lai, L. Wang, H. Chen, K. Zhou, F. Wang, H. Yang, X. Hu, Context-aware domain adaptation for time series anomaly detection, in: Proceedings of the 2023 siam international conference on data mining (SDM), SIAM, 2023, pp. 676–684.

[31] L. Li, J. Yan, Q. Wen, Y. Jin, X. Yang, Learning robust deep state space for unsupervised anomaly detection in contaminated time-series, IEEE Transactions on Knowledge and Data Engineering 35 (2023) 6058–6072.

[32] Y. Nam, S. Yoon, Y. Shin, M. Bae, H. Song, J.-G. Lee, B. S. Lee, Breaking the time-frequency granularity discrepancy in time-series anomaly detection, Proceedings of the ACM on Web Conference 2024 (2024).

[33] J. Xu, H. Wu, J. Wang, M. Long, Anomaly transformer: Time series anomaly detection with association discrepancy, arXiv preprint arXiv:2110.02642 (2021).

[34] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, T. Soderstrom, Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding, in: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, 2018, pp. 387–395.

[35] D. Li, D. Chen, L. Shi, B. Jin, J. Goh, 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.

[36] Y. Yang, C. Zhang, T. Zhou, Q. Wen, L. Sun, Dcdetector: Dual attention contrastive representation learning for time series anomaly detection, in: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2023, pp. 3033–3045.

[37] Y. Fang, J. Xie, Y. Zhao, L. Chen, Y. Gao, K. Zheng, Temporal-frequency masked autoencoders for time series anomaly detection, in: 2024 IEEE 40th International Conference on Data Engineering (ICDE), IEEE, 2024, pp. 1228–1241.

[38] H. Zhou, K. Yu, X. Zhang, G. Wu, A. Yazidi, Contrastive autoencoder for anomaly detection in multivariate time series, Information Sciences 610 (2022) 266–280.

[39] G. Papamakarios, T. Pavlakou, I. Murray, Masked autoregressive flow for density estimation, Advances in neural information processing systems 30 (2017).

[40] L. Dinh, J. Sohl-Dickstein, S. Bengio, Density estimation using real nvp, arXiv preprint arXiv:1605.08803 (2016).

[41] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, M. Long, itransformer: Inverted transformers are effective for time series forecasting, arXiv preprint arXiv:2310.06625 (2023).

[42] S. Brody, U. Alon, E. Yahav, How attentive are graph attention networks?, arXiv preprint arXiv:2105.14491 (2021).

[43] Z. Zhang, Z. Geng, Y. Han, Graph structure change-based anomaly detection in multivariate time series of industrial processes, IEEE Transactions on Industrial Informatics 20 (4) (2024) 6457–6466.

[44] J. Goh, S. Adepu, K. N. Junejo, A. Mathur, A dataset to support research in the design of secure water treatment systems, in: Critical Information Infrastructures Security: 11th International Conference, Springer, 2017, pp. 88–99.

[45] C. M. Ahmed, V. R. Palleti, A. P. Mathur, Wadi: a water distribution testbed for research in the design of secure cyber physical systems, in: Proceedings of the 3rd international workshop on cyber-physical systems for smart water networks, 2017, pp. 25–28.

[46] A. Abdulaal, Z. Liu, T. Lancewicki, Practical approach to asynchronous multivariate time series anomaly detection and localization, in: Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining, 2021, pp. 2485–2494.

[47] L. Ruff, R. A. Vandermeulen, N. Görnitz, A. Binder, E. Müller, K.-R. Müller, M. Kloft, Deep semi-supervised anomaly detection, arXiv preprint arXiv:1906.02694 (2019).

[48] S. Goyal, A. Raghunathan, M. Jain, H. V. Simhadri, P. Jain, Drocc: Deep robust one-class classification, in: International conference on machine learning, PMLR, 2020, pp. 3711–3721.

[49] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, H. Chen, Deep autoencoding gaussian mixture model for unsupervised anomaly detection, in: International conference on learning representations (ICLR), 2018.

[50] O. Gungor, A. Rios, P. Mudgal, N. Ahuja, T. Rosing, A robust framework for evaluation of unsupervised time-series anomaly detection, in: Pattern Recognition, Springer Nature Switzerland, 2025, pp. 48–64.

[51] M. S. Sarfraz, M.-Y. Chen, L. Layer, K. Peng, M. Koulakis, Position: quo vadis, unsupervised time series anomaly detection?, arXiv preprint arXiv:2405.02678 (2024).

[52] R. Wu, E. J. Keogh, Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress, IEEE transactions on knowledge and data engineering 35 (3) (2021) 2421–2429.

[53] B. Yu, H. Yin, Z. Zhu, Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, International Joint Conferences on Artificial Intelligence Organization, 2018, p. 3634–3640.