Industrial Fault Detection Using Spatio-Temporal Variational Graph Attention Autoencoder

Shuyuan Xu

Chengdu University of Technology, Chengdu 610059, Sichuan, China

Abstract: ***In critical domains such as industrial IoT, data-center operations, and financial risk control, accurate anomaly detection on multivariate time series (MTS) is essential for ensuring system reliability and security. Anomalies in MTS are rarely isolated numerical deviations; instead, they manifest as subtle disruptions of the normal cooperative patterns among interrelated variables. Specifically, while the encoder can perceive graph structure, the decoder is often reduced to a structure-agnostic multilayer perceptron (MLP), causing valuable structural information to be lost during reconstruction and thus limiting the precision of "relationship anomaly" detection.***

Keywords: Industrial fault detection; Multivariate time series detection; Encoder.

1. INTRODUCTION

1.1 Research Background and Problem Significance

We are living in a data-driven era. From industrial production lines that power the global economy, to massive data centers that sustain the modern information society, to spacecraft that safeguard humanity's exploration of deep space, the stable operation of these complex systems depends on real-time monitoring by vast sensor networks. These networks continuously generate high-dimensional multivariate time series (MTS) data, which act as the "vital signs" of the systems, accurately recording dynamic changes in key indicators ranging from temperature and pressure to network traffic and CPU load. Therefore, developing technologies that can automatically, in real time, and accurately detect anomalies from such MTS data is no longer a purely academic issue; it has become a core technological requirement for safeguarding modern infrastructure, preventing catastrophic failures, and avoiding major economic losses. Neural network optimization is comprehensively addressed by Gong et al. (2023) through their review of neural network lightweighting techniques [1], while manufacturing automation progresses through Xie and Chen's (2025) Maestro system for multi-agent task recognition and optimization [2]. Digital advertising technologies show substantial innovation with Zhu's (2025) RAID system for reliability automation in large-scale ad platforms [3], Zhang's (2025) CrossPlatformStack enabling high-availability deployment across meta services [4], Hu's (2025) GenPlayAds for procedural playable 3D ad creation [5], and Li, Lin, and Zhang's (2025) privacy-preserving framework incorporating federated learning and differential privacy [6]. Recommendation systems further evolve through Li, Wang, and Lin's (2025) graph neural network enhanced sequential recommendation method for cross-platform ad campaigns [7]. Urban computing and public infrastructure benefit from Xu's (2025) CivicMorph for generative public space modeling [8], while communication systems advance through Tu's (2025) AutoNetTest for intelligent 5G network automation [9]. Data analytics is enhanced by Xie and Liu's (2025) DataFuse for multimodal interview analytics [10], and platform stability is strengthened through Zhu's (2025) ReliBridge as a scalable LLM-based backbone for small businesses [11]. Content creation is revolutionized by Hu's (2025) few-shot neural editors for 3D animation [12], and industrial applications include Tan et al.'s (2024) damage detection using deep transfer learning [13]. Digital transformation extends to Zhuang's (2025) theoretical construction of real estate marketing strategies [14], and recommendation systems advance through Han and Dou's (2025) hierarchical graph attention networks with multimodal knowledge graphs [15]. Healthcare applications include Yang's (2025) Prompt-Biomrc model for intelligent consultation [16], while conversational AI progresses through Yang et al.'s (2025) RLHF fine-tuning for alignment with implicit user feedback [17], complemented by parallel optimization methods for LLM-based recommendation systems [18]. Business intelligence features Zhang et al.'s (2025) AI-driven sales forecasting in gaming [19], while cloud infrastructure benefits from Yang's (2025) high-availability architecture design [20]. Urban planning advances through Xu's (2025) UrbanMod for accelerated city architecture planning [21], healthcare through Hsu et al.'s (2025) MEDPLAN for personalized medical plans [22], and cross-media analytics through Yuan and Xue's (2025) multimodal integration framework using graph neural networks [23].

1.2 Existing Methods and Their Limitations

To address this challenge, both academia and industry have invested tremendous effort. Traditional statistical methods, such as Principal Component Analysis (PCA) or Vector Autoregression (VAR), struggle to accommodate the highly nonlinear and dynamic nature of modern MTS data due to their strict distributional assumptions and limitations in capturing only linear relationships. In recent years, data-driven approaches led by deep learning have become the mainstream of research in this field, thanks to their powerful nonlinear modeling capabilities. Most of these methods follow an unsupervised learning paradigm: by training on large volumes of normal data, they learn a model that precisely characterizes the "normal pattern," and then flag data points that the model cannot adequately explain as anomalies [1].

Existing deep learning methods can generally evolve along two technical routes. The first route focuses on capturing complex temporal dependencies in MTS. Pioneered by Malhotra et al. [cite], these approaches use recurrent neural networks (RNNs) and their variants (such as LSTM and GRU) to learn and reconstruct time series through LSTM autoencoders (LSTM-AE). Subsequent work, such as the classic OmniAnomaly [cite Su, et al.], significantly enhances the model's ability to model the probability distribution of time series by combining RNNs with variational autoencoders (VAE) and more sophisticated stochastic processes like normalizing flows.

The second route keenly grasps the importance of inter-variable relationships and innovatively introduces Graph Neural Networks (GNN). Representative works such as GDN [cite Zhao, et al.] and STG-AE [cite Deng & Hooi] model the N variables in MTS as N nodes of a graph, pre-compute or learn the relationships among variables from data as edges, and then leverage GNN to simultaneously aggregate information across both temporal and spatial (structural) dimensions.

2. BACKGROUND

2.1 Multivariate Time Series Anomaly Detection

Multivariate Time Series (MTS) $X = \{x_1, x_2, \cdots, x\_T\}$ is a data sequence composed of T timestamps, where each timestamp $x\_t \in R\^N$ is a N -dimensional vector representing the observed values of N distinct variables (or sensors) at time t. The goal of MTS anomaly detection is to identify time points or intervals in the sequence that deviate from normal behavior patterns. Depending on the type of anomaly, they can be categorized into point anomalies, contextual anomalies, and collective (relational) anomalies. Our work primarily focuses on the latter two, especially collective anomalies caused by the disruption of inter-variable collaborative relationships.

2.2 Variational Autoencoder

The Variational Autoencoder (VAE), introduced by Kingma and Welling [cite], is a deep generative model that incorporates a probabilistic perspective into the standard Autoencoder (AE), aiming to learn the latent probability distribution of the data. A VAE consists of two components:

Encoder: Also called the inference or recognition network $q\_\phi(z|x)$. It takes input data x and outputs the parameters of a latent distribution, typically the mean μ and log-variance $\log(\sigma^2)$ of a Gaussian. It learns the mapping from data space to latent space.

Decoder: Also called the generative network $p\_\theta(x|z)$. It samples a latent vector z from the latent distribution parameterized by μ and $\log(\sigma^2)$ (via the reparameterization trick), then attempts to reconstruct the original input data x′ from z. It learns the mapping from latent space to data space [2].

The optimization objective of a VAE is to maximize the Evidence Lower Bound (ELBO), which is equivalent to minimizing a loss function composed of two terms:

Loss_VAE = Loss_reconstruction + β * Loss_KL

Here, Loss_reconstruction (e.g., MSE) measures the similarity between the reconstructed data x′ and the original data x. Loss_KL (KL divergence) measures the discrepancy between the latent distribution produced by the encoder $q\_\phi(z|x)$ and a preset prior distribution $\mathrm{p(z)}$ (usually a standard normal $\mathrm{N(0, I)}$). β is a hyperparameter balancing the two. The KL term acts as a regularizer, forcing the encoder to learn a well-structured, continuous, and complete latent space, which not only improves generalization but also enables the generation of new samples similar to the training data.

3. MODEL AND METHOD

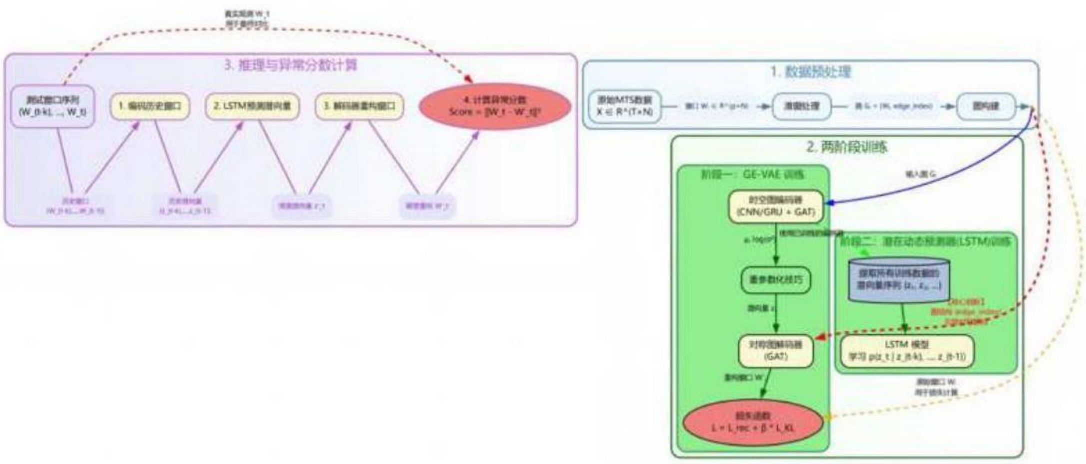
3.1 Motivation

The core motivation of our work stems from observing a pervasive "asymmetry" bottleneck in existing autoencoder-based MTS anomaly detection methods. Many advanced approaches have recognized the importance of using GNNs to encode spatial dependencies among variables; however, their decoding process is often simplified into a direct mapping from a global latent vector to high-dimensional raw data, typically performed by a structure-agnostic MLP. We argue that this design has a fundamental flaw: the fine-grained, node-level structural information painstakingly learned during encoding is compressed into a global vector after graph pooling, yet the decoder cannot leverage the original graph topology to guide this "decompression" process.

3.2 Overall Model Architecture

Our GE-VAE framework follows a two-stage, "predictive reconstruction"-based anomaly detection paradigm. The overall pipeline is illustrated below:

Symmetric Graph-Embedded Variational Autoencoder (GE-VAE) Framework Diagram for Robust Multivariate Time-Series Anomaly Detection



Symmetric Graph-Embedded Variational Autoencoder (GE-VAE) Framework Diagram for Robust Anomaly Detection in Multivariate Time Series

Data preprocessing: the raw MTS data $X \in R^n(T \times N)$ is first subjected to a sliding-window procedure, being split into a series of overlapping windows $W\_i \in R^n(p \times N)$, where p is the window length. Each window W_i is treated as a graph G_i, in which N variables serve as nodes; the relationships between nodes are determined by a predefined graph-construction method (e.g., correlation) to form the edge index edge_index.

Phase 1: GE-VAE Training:

Encoding: Each graph window G_i is fed into a spatio-temporal graph encoder, which compresses it into parameters of a latent distribution via CNN/GRU and GAT layers.

Number μ_i and $\log(\sigma \cdot 2)\ \_i$.

Decoding: Sample a latent vector z_i from this distribution and feed it into a symmetric graph decoder. The decoder leverages edge_index to perform structured information expansion via GAT layers, ultimately reconstructing the window W'_i.

Optimization: The model is trained by minimizing node reconstruction loss and KL divergence loss, learning to precisely compress and decompress "normal" graph windows.

Phase Two: LSTM Training:

All training-set graph windows are encoded by the trained GE-VAE encoder to produce a latent-vector time series $\{z_1, z_2, \cdots, z\_M\}$.

The latent dynamics predictor (LSTM) is trained on this z sequence to learn to predict the latent vector at the next time step, i.e., to learn $p(z\_t|z\_\{t-1\}, \cdots, z\_\{t-1\})$.

Inference and Anomaly Score Calculation:

For a new sequence $\{W\_\{t-k\}, \cdots, W\_t\}$ in the test set, we first feed its first k windows $\{W\_\{t-k\}, \cdots, W\_\{t-1\}\}$ into the $\mathrm{GE-VAE}$ encoder to obtain the latent vector $\{z\_\{t-k\}, \cdots, z\_\{t-1\}\}$.

Feed these k historical z vectors into the trained LSTM to predict the expected latent vector $z' - t$ for the next time step.

Send this predicted z^1_t into the GE - VAE graph decoder to reconstruct the desired window $W' - t$.

Compute the difference (e.g., squared error) between the expected reconstructed window $W' - t$ and the true observed window $W - t$; this error is the anomaly score at time t. If the true observation deviates strongly from the history-based expectation, the anomaly score will be high [3].

3.3 Spatio-Temporal Graph Encoder

The encoder's goal is to map a high-dimensional, structured graph window $G\_i = (X\_i, A)$, where $X\_i \in R^{\sim}(N \times p)$ is the node feature matrix (a p-dimensional time series for each node) and A is the adjacency matrix (represented by edge_index), into a low-dimensional latent distribution $q\_\phi(z\_i|G\_i)$. This process is divided into two steps:

Node-level temporal feature extraction: For each node (variable), its p-dimensional time series must first be encoded into a fixed-dimensional feature vector. We explore two approaches:

CNN: A 1-D convolutional network (1D-CNN) is used to capture local patterns and shape features in the time series.

GRU: A gated recurrent unit (GRU) is used to capture sequential dependencies.

After this step, each node's p -dimensional time series is transformed into a d_t-dimensional initial node embedding $h\_j\hat{0} \in R(\hat{d}\_t)$.

Graph-level spatial information aggregation: Next, all initial node embeddings $H^{\sim}0 = \{h\_1\hat{0}, \cdots, h\_N\hat{0}\}$ and the graph's edge_index are fed into a two-layer graph attention network (GAT):

$$H(\hat{1} + 1) = GAT(H\hat{1}, edge\_index)$$

Via message passing and self-attention, GAT allows each node's representation to incorporate information from its neighbors. After two GAT layers, we obtain the final node representations H^2L, which jointly encode each variable's own temporal dynamics and the spatial dependencies among variables.

Graph pooling and latent distribution output: To obtain a single vector representing the entire graph window, we apply a global mean pooling operation to all final node representations $H^{\sim}L$:

$h\_G = GlobalMeanPool\left(H\hat{L}\right)$

Finally, this graph-level representation h_G is passed through two separate linear layers to output the mean μ and log-variance $\log(\sigma^{2}{^\circ}2)$ of the latent distribution:

$\mu = FC\_\mu(h\_G)$

$\log(\sigma^2) = FC\_logvar(h\_G)$

3.4 Symmetric Graph Decoder

This is the core of our methodology. The decoder's task is to perform the encoder's "inverse operation": given a global vector $z \in R^2(d\_z)$ sampled from the latent distribution and the graph structure edge_index, reconstruct the original node features $X \in R(N \times p)$.

Broadcast/Un-pooling: This is the first step of graph decoding and the most fundamental difference from an MLP decoder. We replicate the latent vector z that encodes global information N times, assigning an initial representation to every node in the graph. More formally, for all graphs in a batch, we broadcast z to the corresponding nodes via the batch_map index, yielding the initial node representations $H\_dec\hat{0}$.

$H\_dec\hat{0} = FC\_initial(z[batch\_map])$

where FC_initial is a linear layer that projects z to the hidden dimension of the GNN.

Structured information expansion: Next, we feed $H\_dec\hat{0}$ and edge_index into a two-layer GAT network [4] that is structurally symmetric to the encoder.

$H\_dec\left(\hat{1} + 1\right) = GAT\_dec(H\_dec\hat{1}, edge\_index)$

During this process, the GNN acts as both an "information diffuser" and a "relationship coordinator." Each node's representation is no longer isolated; instead, through message passing it continuously "negotiates" with its neighbors, adjusting and refining its representation according to the graph structure, ensuring that the finally reconstructed node features satisfy the required cooperative relationships.

4. EXPERIMENTS

4.1 Datasets and Evaluation Metrics

Dataset: We validate our method on the publicly available server-monitoring dataset SMD (Server Machine Dataset) released by Su et al. [cite]. We use the machine-1-1 subset, which contains 38 sensor variables and records 28,479 training timestamps and 28,479 test timestamps. It is a typical high-dimensional, complex MTS dataset.

Evaluation metrics: We adopt the standard metrics in time-series anomaly detection: Precision, Recall, and F1-Score. We follow the point-adjust evaluation strategy [cite Su, et al.], i.e., if any point within a ground-truth anomalous segment is detected, all points in that segment are considered correctly detected [5].

4.2 Implementation Details and Hyperparameters

Our model is implemented entirely in PyTorch and PyTorch Geometric. Key hyperparameters: sliding-window length l_win=48, LSTM sequence length l_seq=12, latent-space dimension code_size=32, batch size batch_size=32. Both GE-VAE and LSTM are trained for 100 epochs with learning rates 0.0006 and 0.0003, respectively. Graph construction uses Pearson correlation with threshold 0.8. All experiments are run on a single NVIDIA GeForce RTX 3080 GPU.

Seed	Precision	Recall	F1 Score
42	0.6601	0.9165	0.7674
123	0.6325	0.8781	0.7353

1024	0.6268	0.8703	0.7288
Mean	0.6398	0.8883	0.7438
Std	0.0146	0.0205	0.0171

Ultimately, our GE-VAE model can be scientifically reported to achieve an average F1 score of 0.744 ± 0.017. The low standard deviation demonstrates the model's robustness. Even in the worst random run (F1=0.7288), its performance remains very high, while its best performance reaches F1=0.7674, showcasing its great potential.

5. CONCLUSION

This paper proposes a symmetric Graph-Embedded Variational Auto-Encoder (GE-VAE) to address the insufficient modeling of inter-variable synergistic relationships and the asymmetry of encoder-decoder architectures in existing multivariate time-series anomaly detection methods. By combining LSTM-based prediction of latent-space dynamics, our framework effectively identifies anomalies through predictive reconstruction errors.

Extensive experiments on the public SMD dataset demonstrate the effectiveness of our approach. In particular, we conduct rigorous multiple independent trials and systematic post-processing tuning. Our ablation studies clearly validate the superiority of the symmetric graph decoder, while the failed exploration of introducing edge-reconstruction loss offers valuable practical insights for the field.

REFERENCES

[1] Gong, Z., Zhang, H., Yang, H., Liu, F., & Luo, F. (2023). A Review of Neural Network Lightweighting Techniques. Innovation & Technology Advances, 1(2), 1–24. https://doi.org/10.61187/ita.v1i2.36
[2] Xie, Minhui, and Shujian Chen. "Maestro: Multi-Agent Enhanced System for Task Recognition and Optimization in Manufacturing Lines." Authorea Preprints (2025).
[3] Zhu, Bingxin. "RAID: Reliability Automation through Intelligent Detection in Large-Scale Ad Systems." (2025).
[4] Zhang, Yuhan. "CrossPlatformStack: Enabling High Availability and Safe Deployment for Products Across Meta Services." (2025).
[5] Hu, Xiao. "GenPlayAds: Procedural Playable 3D Ad Creation via Generative Model." (2025).
[6] Li, X., Lin, Y., & Zhang, Y. (2025). A Privacy-Preserving Framework for Advertising Personalization Incorporating Federated Learning and Differential Privacy. arXiv preprint arXiv:2507.12098.
[7] Li, X., Wang, X., & Lin, Y. (2025). Graph Neural Network Enhanced Sequential Recommendation Method for Cross-Platform Ad Campaign. arXiv preprint arXiv:2507.08959.
[8] Xu, Haoran. "CivicMorph: Generative Modeling for Public Space Form Development." (2025).
[9] Tu, Tongwei. "AutoNetTest: A Platform-Aware Framework for Intelligent 5G Network Test Automation and Issue Diagnosis." (2025).
[10] Xie, Minhui, and Boyan Liu. "DataFuse: Optimizing Interview Analytics Through Multimodal Data Integration and Real-Time Visualization." (2025).
[11] Zhu, Bingxin. "ReliBridge: Scalable LLM-Based Backbone for Small Business Platform Stability." (2025).
[12] Hu, Xiao. "Learning to Animate: Few-Shot Neural Editors for 3D SMEs." (2025).
[13] Tan, C., Gao, F., Song, C., Xu, M., Li, Y., & Ma, H. (2024). Proposed Damage Detection and Isolation from Limited Experimental Data Based on a Deep Transfer Learning and an Ensemble Learning Classifier.
[14] Zhuang, R. (2025). Evolutionary Logic and Theoretical Construction of Real Estate Marketing Strategies under Digital Transformation. Economics and Management Innovation, 2(2), 117-124.
[15] Han, X., & Dou, X. (2025). User recommendation method integrating hierarchical graph attention network with multimodal knowledge graph. Frontiers in Neurorobotics, 19, 1587973.
[16] Yang, J. (2025, July). Identification Based on Prompt-Biomrc Model and Its Application in Intelligent Consultation. In Innovative Computing 2025, Volume 1: International Conference on Innovative Computing (Vol. 1440, p. 149). Springer Nature.
[17] Yang, Zhongheng, Aijia Sun, Yushang Zhao, Yinuo Yang, Dannier Li, and Chengrui Zhou. "RLHF Fine-Tuning of LLMs for Alignment with Implicit User Feedback in Conversational Recommenders." arXiv preprint arXiv:2508.05289 (2025).

[18] Yang, Haowei, Yu Tian, Zhongheng Yang, Zhao Wang, Chengrui Zhou, and Dannier Li. "Research on Model Parallelism and Data Parallelism Optimization Methods in Large Language Model-Based Recommendation Systems." arXiv preprint arXiv:2506.17551 (2025).
[19] Zhang, Jingbo, et al. "AI-Driven Sales Forecasting in the Gaming Industry: Machine Learning-Based Advertising Market Trend Analysis and Key Feature Mining." (2025).
[20] Yang, Yifan. "High Availability Architecture Design and Optimization Practice of Cloud Computing Platform." European Journal of AI, Computing & Informatics 1.1 (2025): 107-113.
[21] Xu, Haoran. "UrbanMod: Text-to-3D Modeling for Accelerated City Architecture Planning." Authorea Preprints (2025).
[22] Hsu, Hsin-Ling, et al. "MEDPLAN: A Two-Stage RAG-Based System for Personalized Medical Plan Generation." arXiv preprint arXiv:2503.17900 (2025).
[23] Yuan, Yuping, and Haozhong Xue. "Multimodal Information Integration and Retrieval Framework Based on Graph Neural Networks." Proceedings of the 2025 4th International Conference on Big Data, Information and Computer Network. 2025.