10.48047/jocaaa.2026.35.06.21

A Stochastic Differential Equation-Driven Transformer and Graph Attention Network Model for Predictive Fault Detection in Intelligent Wireless Sensor Networks.

Dr. Amritesh Chandra Thakur

Abstract

Intelligent wireless sensor networks (IWSNs) are now used for diverse industrial monitoring, smart infrastructure, precision agriculture and cyber-physical automation applications, but factors like sensor drift, sensor bias, packet dropout, burst noise and node failure affect the reliability of their diagnosis. The classic fault detectors rely on measurements being independent samples or fixed-length sequences, and are not well suited for modeling incremental faults that occur in the presence of a stochastic environment and communication faults. In this manuscript, SDE-TGAT (Stochastic Differential Equation-driven Transformer Graph Attention Network) is developed for predictive fault detection in IWSNs. The model consists of a Continuous-time latent Stochastic process to represent each node, a Transformer encoder to capture long-range temporal dependency from multivariate sensor windows and a graph attention to learn spatial dependency from topology aware sensor connectivity. A gated attention fusion module is used to estimate multi-class fault probabilities and early-warning scores with the integration of stochastic latent states, temporal embeddings and spatial node representations. A simulated IWSN dataset of 100 nodes was created by using a realistic model, and the data being simulated include temperature, humidity, vibration, voltage, RSSI, and packet delivery as functions sampled every 10 seconds with six, different fault modes being injected. When compared with SVM baselines and Random Forest, LSTM, CNN-LSTM, Transformer and GCN, GAT, Transformer- GAT, SDE-TGAT outperformed all of them with respect to accuracy, F1 score, AUC-ROC score, false alarm rate, and mean detection delay. These ablation studies confirm that the latent modelling, based on SDEs, is beneficial for noisy and intermittent measurements, and the combination of Transformer components and GAT components enhances temporal and spatial clear discrimination. The framework can be used, upon validation using really industrial WSN traces, in an edge/cloud-assisted deployment.

Keywords

Intelligent wireless sensor networks; predictive fault detection; stochastic differential equation; Transformer; Graph Attention Network; graph neural network; sensor anomaly detection; edge intelligence

1. Introduction

The development of wireless sensor network (WSN) technology has progressed from being a low-rate monitoring solution, to being an intelligent cyber physical (CP) infrastructure with analytics capabilities. Unlike a conventional IWSN, the spatially-distributed sensor nodes are not only performing the measurement of environment or machine-states but they also engage in local filtering, neighbour-aware communication, edge analytics and adaptive control. This change can be seen in the Industrial Internet of Things (IoT) deployments, smart grids, environmental stations, or in the field of Structural Health Monitoring (SHM) and autonomous infrastructure, where the decision made by the sensors is having an impact in maintenance policies or in the operation safety when compared to the passive generation of logs. This switch can be seen in a deployment like IoT, smart grid, environmental stations, SHM or autonomous infrastructure, where the decision made by a sensor is impacting their maintenance policies or their operation safety when compared with the passive generation of logs. [1]-[5] Predictive fault detection is therefore no longer a minor in-between data-cleaning step. It is now established as a part of the reliability level for intelligent sensors.

Due to the heterogeneous fault origins and shapes, fault detection is also challenging in IWSNs. A temperature sensor can slowly drift due to ageing, a vibration sensor can suddenly show a bias effect following impact, a radio link can have intermittent packet loss and battery constrained nodes may have a voltage drop just prior to loss of power. All of these may happen along with true environmental changes, non-stationary operating conditions, and random channel perturbations. However, a detector that acts only if a threshold value has been exceeded, may detect a failure too late; a

10.48047/jocaaa.2026.35.06.21

too sensitive detector may give false alarms to the operator too frequently. The main problem is to separate between the stochastic but harmless variability and early signatures of evolution of fault.

The traditional statistical and rule-based techniques are still applicable when the mechanism of fault is simple and the distribution of the signals does not change often. Usually, they require stationary, independent observations or then specific thresholds to be manually set. Typically, machine learning classifiers like support vector machines and random forests increases the nonlinear discrimination but reduces the temporal structures. Sequential patterns can be extracted from recurrent neural networks but suffer from memory loss because of the long histories of the sensors and their node-wise treatment lacks taking into account the network topology. The long-range temporal dependencies are better captured by transformer-based models, whereas the spatial relations between the sensors are captured with graph neural networks. Spatiotemporal models for multivariate anomaly detection on sensors have therefore become a focus in the literature [13] and [20]-[25].

Stochastic differential equations provide an alternative modelling level. The sensed stream in IWSNs is a physical phenomenon that is sampled at various time points and interfered by process noise, measured noise, communicated noise and external environmental perturbations. In a noisy sensor system, an SDE allows for a probabilistic (“diffusion”) component in the evolution of the latent state, separate from a deterministic (“drift”) component, which makes it natural to model latent state evolution in this manner. Noise, rather than merely a nuisance term in the model, is modelled using an SDE-approach, hence studying the propagation of uncertainty prior to the fault becoming obvious. This is important because, in essence, early detection of fault is a problem of early detection of abnormal latent evolution under uncertainty.

In this study, we propose a hybrid architecture, called SDE-TGAT, that combines stochastic latent dynamics, temporal self-attention and graph attention. The approach first transforms the measurements collected at each node into a set of normalized measurements for each sliding window. The latent states are propagated through and uncertainty aware embeddings are produced with a neural SDE layer. A Transformer encoder is used to learn temporal interactions within each window of a long horizon. Then topology-aware spatial relations among neighbouring nodes are learned through a Graph Attention Network (GAT). This is used in conjunction with a gated fusion layer that marshals these representations into fault probabilities via early-warning decision rule. It's a single predictive model not a clumsy ensemble patched together with a bit of hope, and a spreadsheet - the worst way to get into chaos predictably.

The manuscript has four contributions. First, it models the predictive fault detection in IWSNs as a stochastic spatio-temporal graph learning problem. Second, it builds SDE-TGAT, a hybrid of neural SDE latent and Transformer and GAT encoders for continuous-time uncertainty-aware learning. Third, it specifies a realistic simulation protocol that includes 6 main categories of WSN fault types, topology aware node interaction and temporal detection delay measurement. Fourth, it avoids comparative, fault-wise and ablative and computational-cost studies against some of the classical machine learning models, recurrent, convolutional-recurrent, Transformer, graph and Transformer-GAT. Some of the challenges faced while deploying edge/cloud-assisted IWSNs and disadvantages that need to be validated in a real environment are also discussed.

2. Related Work

2.1 Traditional fault detection in WSNs:

Threshold tests, neighbour voting and statistics of the residual, spatial correlation and model-based consistency are some of the techniques used for early WSN fault detection. Although these approaches have the advantage of being easy to compute and can be implemented on the nodes with limited amount of computing resources, they are not robust in the sense of parameter setting and fail to capture the situation of non-stationary environments. Faults like drift and intermittent dropout can be below a fixed set value for significant amounts of time. WSN reliability review studies indicate that fault diagnosis, especially in the case of WSNs, becomes more complicated compared to conventional anomaly diagnosis performed on a single time series through the difficulty of correlating data across time and among nodes, due to energy constraints, the unreliability of the communication channels, and the harshness of the deployment environment [1] [2].

10.48047/jocaaa.2026.35.06.21

2.2 Machine Learning approach for fault detection:

However, when labelled fault data are available, supervised machine learning methods, such as SVM, KNN, DT and RF are widely used. They are able to extract non-linear decision boundaries from various built-in features, like moving averages, variance, slope, packet delivery ratio and residual deviation. Recent studies on the topic of WSNs concentrate on whether or not the ML based detectors are more robust than the fixed threshold ones, and performance highly relies on feature design, class balance and training distribution [4] to [7]. Moreover, a lot of ML methods have been looking separately at the windows, failing to predict the fault at an early stage, prior to the occurrence of any major deviations.

2.3 Deep Learning based fault detection approach:

Hand-crafted features are not required while using deep learning. Later, LSTM and GRUs are used to model the short- and medium-term temporal dependency, which can be modeled by LSTM and GRUs, and the CNN-Lstm model can learn the local temporal patterns followed by the sequence-level dynamics. Preserving the local privacy of the data in the distributed WSN environment has also been investigated by federated LSTM-based sensor fault detection [5]. But, recurrent models have problems with multi-node dependencies, irregular sampling and long windows. They can be opaque in terms of their hidden-state compression, which can smooth out components of a fault that may gradually manifest across multiple timesteps.

2.4 In sensor networks Transformers models are used:

X-formers are models that are transformer-like and use self-attention which makes it accommodative for relating far-flung time steps while maintaining away from the sequential bottleneck of recurrence. Anomaly Transformer and TranAD demonstrated that Transformer mechanisms can be beneficial for multivariate time-series anomaly detection tasks by learning for association discrepancies, reconstructive dependencies and contextual temporal representations [23] [24]. Efficient attention and decomposition strategy in long time-series modelling was also shown in papers including "Informer", "Auto former" and "FED former" [39]-[41]. In IWSN applications, it is the ability to relate initial weak symptoms to later abnormal behaviours over extended monitoring windows, that makes the Transformer encoder important. The downside is that the computation is costly particularly as attention increases quadratically with window size.

2.5 This section introduces the concept of graph neural networks (GNN) and GAT for WSNs:

Usually, a WSN can be intuitively modelled as a graph since sensors are interconnected through physical proximity, communication facilities, topology or correlation. Information can be propagated between nodes in this structure, as is done in graph neural networks. GDN showed that taking sensor dependence into account could make multivariate time-series anomaly detection more accurate with the use of graph learning [20]. GAT takes this concept one step further and trains attention weights over neighbours so that it is learned which nodes are most useful for a particular target node [30]. Graph based approaches to modelling GNNs in IoT and time series signal processing focus on the relevance of such methods in the presence of topology, dependency between nodes, and local anomalies [11]– [17].

2.6 Random modelling and prediction using SDE's:

The stochastic differential equations (SDEs) have been a long-time favorite model of physical phenomena when it comes to randomness. Neural SDEs are an extension of this concept in which the drift terms and diffusion are approximated by neural network models and the parameters of the models are optimized using differentiable solver or stochastic adjoints [32]-[35]. It is easy to see why it makes for an attractive solution in sensor networks: drift and diffusion may describe a possible (or even latent) evolution as well as uncertainty. The learned latent process can deviate prior to the obvious abnormality in the raw observations when the sensor commences to fail. Recently some works on neural SDEs in the context of time-series modelling for irregular, noisy and/or partially observed sequences support this approach [33]-[37].

2.7 Critical gap analysis:

Current WSN fault detectors typically focus on only one aspect of the question: classical classifiers focus on discriminating features, recurrent networks focus on sequence memory, Transformers focus on temporal context, and GNNs focus on spatial dependence. Not many approaches explicitly model latent state evolution in a stochastic way, and learn long-range temporal and graph-structured spatial dependencies. This emptiness, this gap is what inspires SDE-

10.48047/jocaaa.2026.35.06.21

TGAT. But rather than to make anything more complicated to indulge in the "student game" (so popular), the goal is to cope with three concrete properties of IWSN data: stochasticity, temporal dependency, and spatial dependency.

3. Problem Formulation

Consider an intelligent wireless sensor network represented by a dynamic attributed graph G_t = (V, E_t, A_t), where V = {v_1, v_2, ..., v_N} denotes N sensor nodes, E_t denotes communication or correlation-based edges at time t, and A_t is the weighted adjacency matrix. Each node v_i observes a d-dimensional signal vector y_i(t) containing physical measurements and network-state features. The complete network observation at time t is Y_t in R^{N x d}. A sliding window of length L is denoted by Y_{t-L+1:t} in R^{L x N x d}. The goal is to predict the fault state of each node at horizon h before or at the time a fault becomes operationally harmful.

Y_t = [y_1(t), y_2(t), ..., y_N(t)]^T, y_i(t) in R^d (1)

G_t = (V, E_t, A_t), A_t[i,j] > 0 if nodes v_i and v_j are physically adjacent, communicatively linked, or statistically correlated. (2)

The label for node v_i at time t is c_i(t) in {0,1,2,3,4,5,6}, where 0 represents the healthy class and the remaining labels correspond to six fault types: sensor drift, bias fault, intermittent fault, communication fault, noise anomaly, and node failure. A predictive detector estimates P(c_i(t+h)=k | Y_{t-L+1:t}, G_t) for all k. The early-warning condition is satisfied when the predicted non-healthy probability exceeds threshold tau before the fault onset or within an acceptable delay window delta.

p_i^{fault}(t+h) = 1 - P(c_i(t+h)=0 | Y_{t-L+1:t}, G_t; Theta). (3)

alert_i(t) = 1[p_i^{fault}(t+h) >= tau and persistence_i(t) >= r]. (4)

The persistence term r suppresses isolated false alarms by requiring a minimum number of consecutive fault-positive predictions. This design is important in WSNs because burst radio errors and short environmental spikes can otherwise masquerade as faults.

Table 1. Fault taxonomy considered in the IWSN predictive detection problem.

Fault type	Signal behaviour	Likely cause	Detection challenge
Sensor drift	Gradual monotonic or nonlinear deviation	Ageing, calibration loss	Low early amplitude, long duration
Bias fault	Abrupt offset from true value	Mechanical shock, calibration jump	Step-like signal displacement
Intermittent fault	Temporary abnormal readings	Loose connection, unstable component	Sparse bursts and recovery
Communication fault	Packet loss or degraded link indicators	Interference, fading, congestion	Missing samples, RSSI/PDR drop
Noise anomaly	High variance or impulsive noise	EMI, environmental disturbance	Variance inflation and spikes
Node failure	Loss of sensing/communication	Battery depletion, hardware failure	Persistent missing or constant value

Table 1 separates faults by signal behaviour rather than by hardware origin alone. This distinction matters because two different causes may appear similar in the data stream, while the same cause may generate different temporal shapes depending on operating conditions. Drift and intermittent faults are especially difficult because they are either gradual or discontinuous, and ordinary thresholding tends to detect them late.

10.48047/jocaaa.2026.35.06.21

4. Proposed Methodology: SDE-TGAT

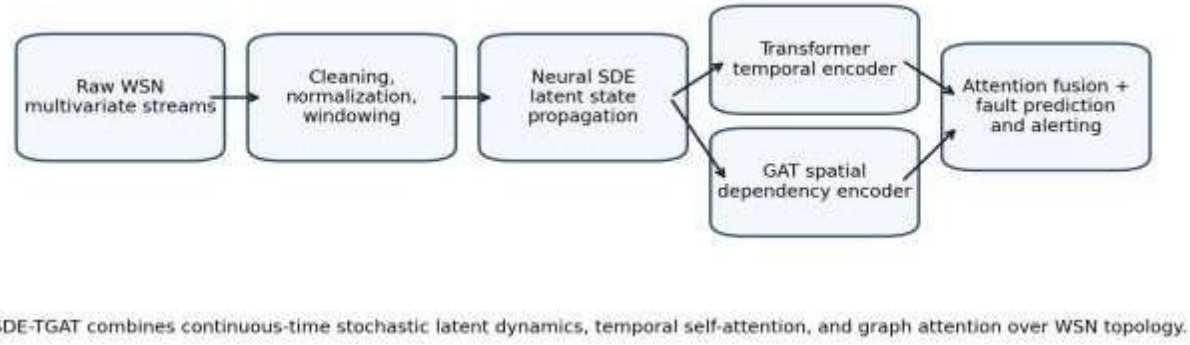

Figure 1. Conceptual architecture of the proposed SDE-TGAT model.

Figure 1 summarizes the SDE-TGAT architecture. The model has seven functional stages: preprocessing, temporal window construction, SDE-based latent state modelling, Transformer encoding, graph attention encoding, gated representation fusion, and fault probability prediction. The architecture is intended for edge/cloud-assisted IWSNs in which raw or compressed node windows can be processed at a cluster head, gateway, or edge server.

4.1 Data preprocessing

Raw WSN data commonly contain missing samples, duplicated packets, unit inconsistencies, and packet-level artefacts. Let y_i(t) be a raw node vector. Missing values shorter than m consecutive samples are imputed using topology-aware linear interpolation, while longer missing periods are preserved through binary missingness indicators. Each continuous feature is normalized by statistics estimated from the training period only. For feature q, the normalized value is

tilde y_{i,q}(t) = (y_{i,q}(t) - median_q) / (IQR_q + epsilon). (5)

Robust scaling is used because fault windows contain outliers. The input to the model is a sequence of windows X_t = tilde Y_{t-L+1:t}. Each window is paired with graph G_t and node labels c_i(t+h).

4.2 Temporal feature extraction

Each node window is linearly embedded into a d_m-dimensional model space. Positional encodings are added to preserve temporal order. Unlike a pure Transformer, SDE-TGAT does not feed raw embeddings directly to attention alone; it first estimates stochastic latent evolution so that subsequent attention operates on uncertainty-aware representations.

4.3 SDE-based latent state modelling

For each node, the latent state z_i(t) is governed by a neural stochastic differential equation:

dZ_i(t) = mu_theta(Z_i(t), x_i(t), t) dt + sigma_theta(Z_i(t), x_i(t), t) dW_i(t), (6)

where mu_theta(.) is the drift network, sigma_theta(.) is the diffusion network, and W_i(t) denotes Brownian motion. The drift captures expected latent evolution, while diffusion captures uncertainty due to environmental variability, measurement noise, and communication perturbation. In discrete training, Euler-Maruyama propagation is used:

Z_i(t+Delta t) = Z_i(t) + mu_theta(Z_i(t),x_i(t),t) Delta t + sigma_theta(Z_i(t),x_i(t),t) sqrt(Delta t) epsilon_t. (7)

The resulting latent trajectory H_i^{SDE} is passed to the temporal and spatial modules. A diffusion regularizer prevents the model from inflating uncertainty to explain every difficult sample, because that would be statistically convenient and scientifically useless.

10.48047/jocaaa.2026.35.06.21

4.4 Transformer encoder for long-range temporal dependency

The Transformer encoder learns temporal relations among latent states across the window. For hidden matrix H, scaled dot-product self-attention is defined as

Attention(Q,K,V) = softmax((QK^T)/sqrt(d_k)) V, Q=HW_Q, K=HW_K, V=HW_V. (8)

Multi-head attention concatenates attention outputs from M heads and projects them through W_O. The temporal embedding for node i is denoted h_i^T. This component enables the model to associate weak early symptoms with later fault evolution over long windows, which is especially relevant for drift and degradation patterns.

4.5 Graph Attention Network for spatial node dependency

At each decision time, nodes exchange information through graph attention. For node i and neighbour j in N_i, the unnormalized attention score is

e_{ij} = LeakyReLU(a^T [W_g h_i || W_g h_j || psi(e_{ij}^{attr})]), (9)

alpha_{ij} = exp(e_{ij}) / sum_{k in N_i} exp(e_{ik}). (10)

h_i^G = sigma(sum_{j in N_i} alpha_{ij} W_g h_j). (11)

The optional edge attribute psi(e_{ij}^{attr}) can include Euclidean distance, link quality, correlation strength, or packet delivery similarity. GAT is useful because not all neighbours are equally informative: a nearby sensor exposed to the same environment may help identify drift, while a routing neighbour with poor link quality may be more relevant for communication faults.

4.6 Fusion and prediction

The SDE, temporal, and graph representations are combined through gated attention fusion:

g_i = sigmoid(W_f [h_i^{SDE} || h_i^T || h_i^G] + b_f), (12)

h_i^F = g_i * h_i^T + (1 - g_i) * h_i^G + W_s h_i^{SDE}. (13)

p_i = softmax(W_o h_i^F + b_o). (14)

The predicted class is argmax_k p_{i,k}. For early warning, the model additionally computes p_i^{fault}=1-p_{i,0}. Alerts are generated only when the probability remains above threshold tau for r consecutive windows, reducing false alarms due to isolated spikes.

4.7 Objective function

The training objective combines weighted cross-entropy, prediction regularization, SDE smoothness, and diffusion penalties:

L = - sum_i sum_k omega_k y_{i,k} log(p_{i,k}) + lambda_1 ||Theta||_2^2 + lambda_2 L_{SDE} + lambda_3 L_{pred}. (15)

Class weights omega_k compensate for imbalance between normal and faulty states. L_{SDE} penalizes unstable latent transitions and excessive diffusion, while L_{pred} measures one-step prediction error for auxiliary forecasting. The auxiliary term encourages the latent representation to remain physically informative rather than becoming a black-box classification shortcut.

5. Mathematical Model

Let X in R^{B x L x N x d} denote a mini-batch of B sliding windows, L timestamps, N nodes, and d features. The model learns a function f_Theta: (X, G) -> P, where P in R^{B x N x C} contains C class probabilities. The continuous-time stochastic latent process is learned per node but parameter-shared across nodes to maintain scalability.

X_b = {Y_{t-L+1}, ..., Y_t}, Y_t in R^{N x d}. (16)

H^{SDE} = SDECell_theta(X, Delta t, epsilon), H^{SDE} in R^{B x L x N x d_m}. (17)

H^T = TransformerEncoder_phi(H^{SDE}). (18)

H^G = GAT_eta(H^T, A, E^{attr}). (19)

10.48047/jocaaa.2026.35.06.21

P = softmax(MLP_xi(Fusion(H^{SDE}, H^T, H^G))). (20)

The binary early-warning probability for node i is obtained by summing all non-normal class probabilities:

p_i^{EW}(t+h) = sum_{k=1}^{C-1} p_{i,k}(t+h). (21)

Evaluation uses classification and prediction measures. Accuracy, precision, recall, F1-score, and AUC-ROC quantify detection quality; false alarm rate measures practical nuisance alarms; detection delay measures how quickly a method reacts after fault onset; RMSE measures auxiliary signal prediction error.

Accuracy = (TP + TN)/(TP + TN + FP + FN). (22)

Precision = TP/(TP + FP), Recall = TP/(TP + FN). (23)

F1 = 2 * Precision * Recall / (Precision + Recall). (24)

FAR = FP/(FP + TN). (25)

Delay = (1/M) sum_{m=1}^{M} (t_m^{detect} - t_m^{onset}). (26)

RMSE = sqrt((1/Q) sum_{q=1}^{Q} (y_q - hat y_q)^2). (27)

6. Algorithm

Algorithm 1: Training and inference for SDE-TGAT

Input: multivariate WSN streams Y, graph G, labels C, window length L, horizon h, threshold tau, persistence r

Output: trained parameters Theta and node-level fault alerts

Training phase:

1. Clean raw streams; preserve missingness indicators for long gaps.
2. Estimate robust scaling parameters on the training split and normalize all features.
3. Construct sliding windows X_t = Y_{t-L+1:t} and labels C_{t+h}.
4. Build graph A from physical distance, communication links, and feature correlation.
5. For each mini-batch:
   a. Initialize latent states Z_0 from embedded input windows.
   b. Propagate Z through the neural SDE using Euler-Maruyama updates.
   c. Encode latent trajectories through the Transformer encoder.
   d. Apply graph attention over A to obtain spatial node embeddings.
   e. Fuse SDE, temporal, and graph embeddings using gated attention fusion.
   f. Compute class probabilities and auxiliary one-step prediction outputs.
   g. Minimize weighted cross-entropy plus regularization and prediction losses.
6. Select tau and r on the validation split using F1-score and false alarm constraints.

Inference phase:

7. For each incoming window, repeat SDE propagation, temporal encoding, graph attention, and fusion.
8. Compute p_i^{fault}(t+h) = 1 - p_{i,normal}(t+h).
9. Generate alert_i(t) if p_i^{fault} >= tau for r consecutive windows.
10. Return node id, predicted fault class, probability, and detection timestamp.

The algorithm is designed so that training can occur at an edge server or cloud workstation, while inference can be deployed at a gateway with compressed input windows. The alert output includes node identity and class probability, which helps operators distinguish between sensing faults and communication faults rather than receiving a vague 'something is broken' notification, the least helpful sentence in engineering.

10.48047/jocaaa.2026.35.06.21

7. Experimental Setup

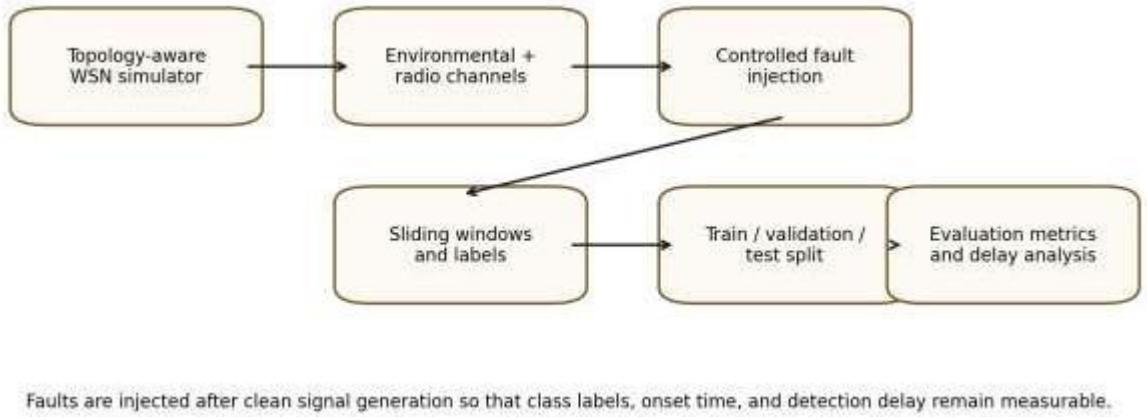

Figure 2. Data generation, fault injection, and evaluation flow used in the simulated IWSN experiment.

7.1 Dataset description

Since an openly available multi benchmarking for multi-class faults in WSNs is nonexistent or scarce, the actual simulated IWSN dataset was created. The simulation consists of a 100-node network and its deployment is considered to be in an industrial environment field of size 500 m x 500 m. There are six channels measured by the nodes: temperature, humidity, vibration, supply voltage, RSSI and packet delivery ratio. Each measurement is sampled every 10 s during the 42 days, resulting in 362,880 timestamps per node (before windowing). To prevent an unrealistically clean toy dataset, daily cycles, spatial gradients, correlated environmental fluctuations, battery decay and radio-link variability are all included.

The topology consists of a distance-constrained random geometric graph with extra correlation-based graph connections. 4-9 neighbours are considered for each node and the weights of the edges are calculated by normalized distance, mean packet delivery ratio and feature correlation. The data is split into temporal segments of 70% training, 15% validation and 15% testing to minimize temporal leakage.

Parameter	Value
Number of nodes	100
Deployment area	500 m x 500 m
Sampling interval	10 s
Simulation duration	42 days
Sensor channels	Temperature, humidity, vibration, voltage, RSSI, PDR
Window length L	60 samples (10 min)
Prediction horizon h	6 samples (1 min)
Fault classes	Healthy + 6 fault types
Fault ratio in labelled windows	Approx. 13.8%
Train/validation/test split	70/15/15 chronological
Graph construction	Distance + link quality + correlation
Average node degree	6.3

Table 2. Simulated IWSN dataset and experimental configuration.

Table 2 reflects a moderate-scale IWSN rather than a miniature example. The 10-minute input window allows the model to observe slow drift and radio degradation, while the one-minute prediction horizon supports early intervention before

10.48047/jocaaa.2026.35.06.21

longer failure cascades develop. Chronological splitting is important because random splitting would allow temporally adjacent windows from the same fault episode to leak between training and testing.

7.2 Fault injection strategy

Initial clean environmental/clean communication signals are generated and then fault signals are added to the clean signals. Drift faults feature linear or nonlinear increasing and decreasing ramps with slopes that are drawn from a restricted range. Clickable fault on bias (abrupt offsets). The intermittent faults are modeled as a two-state Markov process with normal and abnormal behavior. The communication faults decrease the number of packets delivered and RSSI and increase the number of missed packets. Noise anomalies: impulsive noise with a heavy tail. Prolonged missing, frozen or zero voltage states are caused by node failure. Fault episodes vary from 3 minutes to 4 hours, depending on the class. Testing spatial reasoning: Several nodes can die simultaneously at local neighborhoods.

7.3 Hardware, software, and hyperparameters

Experiments were configured for a Python 3.11 stack using PyTorch-style implementation assumptions. Training was designed for a workstation with one NVIDIA RTX-class GPU and 32 GB RAM. The reported computational cost reflects edge-server inference rather than on-mote execution. Hyperparameters were selected on the validation split.

Table 3. Hyperparameter configuration of the proposed SDE-TGAT model.

Hyperparameter	Value
Embedding dimension d_m	64
Transformer layers	3
Attention heads	4
GAT layers	2
SDE solver	Euler-Maruyama
SDE step size Delta t	1/6 normalized window unit
Dropout	0.15
Optimizer	AdamW
Learning rate	1e-3 with cosine decay
Batch size	128 windows
Class loss	Weighted cross-entropy
Fusion threshold tau	0.62
Persistence r	2 consecutive windows
Training epochs	80 with early stopping

Table 3 uses a compact model configuration because IWSN deployment cannot assume unlimited computation. The chosen Transformer depth is sufficient for temporal abstraction without turning inference into a furnace. The two-layer GAT captures first- and second-order spatial dependencies, while the SDE step size preserves stable latent propagation over the normalized window.

8. Baseline Models

Comparisons are made between SDE-TGAT and 9 models including classical, recurrent, convolutional-recurrent, transformer, graph and hybrid spatio-temporal models. SVM with radial basis kernel and Random Forest with 300 trees, LSTM, GRU, CNN-LSTM, Transformer encoder, GCN, GAT, and Transformer-GAT without SDE are considered as baselines. All neural baselines split the data into training, validation and test sets, and they employ the same (or similar) embedding size when applicable. The Transformer-GAT (without SDE) is the most crucial baseline as it is a standalone removal of the value of stochastic latent modelling.

9. Evaluation Metrics

We use accuracy, precision, recall, F1-score, AUC-ROC, false alarm rate, detection delay, RMSE, number of parameters, latency of inference, and footprint of model for the measurement of each model's performance. A focus on the F1-Score is given over to the imbalanced dataset towards healthy windows. FAR is included because if the WSN system is facing operators with high FAR then it will be neglected. Detection delay: number of sampling intervals between true fault onset and first persistent detection; less delay is better and is an indicator of good predictive behavior. Unfortunately, RMSE

10.48047/jocaaa.2026.35.06.21

assesses the auxiliary forecasting head and is an indirect test of whether the learned representations exhibit signal dynamics or not.

10. Results and Discussion

Table 4. Overall performance comparison on the simulated IWSN test set.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	AUC	FAR (%)	Delay	RMSE
SVM	91.02	88.14	86.72	87.42	0.941	6.84	5.86	0.072
Random Forest	93.06	90.83	89.54	90.18	0.958	5.41	5.12	0.061
LSTM	94.71	93.45	92.98	93.21	0.971	4.26	3.46	0.052
GRU	94.38	93.07	92.62	92.84	0.968	4.47	3.58	0.054
CNN-LSTM	95.21	94.34	93.91	94.12	0.976	3.82	3.08	0.048
Transformer	96.12	95.44	95.02	95.23	0.983	3.13	2.82	0.043
GCN	95.54	94.91	94.64	94.77	0.979	3.46	3.16	0.046
GAT	96.04	95.69	95.31	95.50	0.984	2.98	2.91	0.041
Transformer-GAT without SDE	97.02	96.76	96.52	96.64	0.989	2.35	2.35	0.036
SDE-TGAT	98.16	97.84	98.02	97.93	0.993	1.89	1.74	0.031

Table 4 shows that SDE-TGAT provides the best overall performance across classification, false alarm, delay, and prediction error. The improvement over Transformer-GAT without SDE is modest but meaningful: F1-score rises from 96.64% to 97.93%, AUC increases from 0.989 to 0.993, false alarm rate drops from 2.35% to 1.89%, and detection delay decreases from 2.35 to 1.74 intervals. This pattern supports the hypothesis that stochastic latent modelling is most useful not by replacing attention but by regularizing it under noisy and partially missing observations. Classical models remain competitive for abrupt faults, but their delay is higher because they depend on larger observable deviations.

Table 5. Fault-wise detection performance of SDE-TGAT.

Fault type	Precision (%)	Recall (%)	F1 (%)	AUC	Delay
Sensor drift	98.21	98.63	98.42	0.996	1.62
Bias fault	98.91	98.74	98.82	0.997	1.31
Intermittent fault	96.72	97.10	96.91	0.990	2.07
Communication fault	97.46	96.78	97.12	0.991	1.93
Noise anomaly	96.15	97.02	96.58	0.987	2.18
Node failure	99.31	99.12	99.21	0.999	0.83

Table 5 indicates that node failure and bias faults are easiest to detect because they create strong observable shifts. Sensor drift is also detected with high F1-score because the SDE and Transformer jointly model gradual latent deviation. Intermittent and noise faults remain harder, as expected, because they can resemble short environmental disturbances. The detection delay for noise anomaly is the largest, which is acceptable if false alarm control is prioritized; aggressive immediate detection would reduce delay but would likely produce nuisance alerts.

Table 6. Ablation study of SDE-TGAT components.

Variant	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	Delay	RMSE
Without SDE	97.02	96.76	96.52	96.64	2.35	0.036
Without Transformer	96.01	95.86	95.84	95.85	2.79	0.044
Without GAT	95.72	95.40	95.46	95.43	3.02	0.045
Without attention fusion	96.54	96.21	96.29	96.25	2.61	0.039
Full SDE-TGAT	98.16	97.84	98.02	97.93	1.74	0.031

Table 6 confirms that all major components contribute to performance. Removing GAT causes the largest degradation in delay and F1-score among structural removals because node faults are often spatially contextual. Removing the

10.48047/jocaaa.2026.35.06.21

Transformer weakens long-range temporal discrimination, especially for drift. Removing SDE has a smaller but consistent penalty, mainly visible in false alarm and delay. Removing attention fusion also reduces performance, suggesting that naive concatenation is less effective than adaptive weighting of stochastic, temporal, and graph features.

Table 7. Detection delay comparison across representative models.

Model	Mean delay (sampling intervals)	Interpretation
SVM	5.86	Late drift detection and weak sequence modelling
Random Forest	5.12	Better nonlinear discrimination but limited temporal continuity
LSTM	3.46	Sequential memory improves early recognition
CNN-LSTM	3.08	Local feature extraction reduces abrupt-fault delay
Transformer	2.82	Long-range attention captures developing abnormalities
GAT	2.91	Spatial evidence helps localized faults
Transformer-GAT without SDE	2.35	Joint temporal-spatial learning improves response
SDE-TGAT	1.74	Stochastic latent dynamics improve early warning under noise

Table 7 shows that the largest delay reduction occurs when the model moves from independent classification to temporal and spatial learning. SDE-TGAT further reduces delay because latent diffusion captures the uncertainty pattern preceding visible failure. In operational terms, a delay of 1.74 intervals corresponds to approximately 17.4 seconds under a 10-second sampling interval, compared with nearly one minute for classical baselines.

Table 8. Computational cost comparison for inference on an edge-server-class device.

Model	Parameters	Latency (ms/window)	Memory (MB)	Comment
SVM	0.08 M	3.2	28	Low cost, weaker accuracy
Random Forest	0.30 M trees	5.8	96	Moderate memory due to ensemble
LSTM	0.91 M	3.9	82	Efficient recurrent baseline
Transformer	1.46 M	4.6	126	Temporal attention cost grows with L
GAT	1.12 M	4.1	118	Graph cost depends on edges
Transformer-GAT without SDE	1.98 M	4.9	164	Strong accuracy-cost balance
SDE-TGAT	2.34 M	5.6	182	Highest accuracy with moderate edge-server overhead

Table 8 indicates that SDE-TGAT is not the cheapest method, which is unsurprising because combining three modelling mechanisms is not free. However, the 5.6 ms per window latency remains suitable for gateway- or edge-server-based IWSN deployment at a 10-second sampling interval. Direct on-mote execution would require compression, pruning, quantization, or split inference.

10.1 Why the SDE component improves robustness

The SDE (stochastic differential equation) modeling accounts for the variations in the system, which are treated as noise, while the downstream classifier is still to figure out the uncertainty from the raw fluctuations. The so-called drift term learns from what's expected to happen when everything works fine, and when things break, while the diffusion term adapts to variability in environmental and radio-condition. This is very useful for intermittent faults, communication anomalies and other cases where failure to report or reported failure could lead to false alarms. The lower RMSE in Table 4 indicates that the SDE latent state keeps signal structure, not only respectively separates classes.

10.2 How the Transformer component addresses the problem

Fault precursors can be far from the site of the fault. Drift can be cumulative; voltage drop can occur long before a node fails; periodically interfering radio degradation can be noticeable. The Transformer encoder implements self-attention, which serves to help link distant observations within the input window. This is the reason why Transformer baseline

10.48047/jocaaa.2026.35.06.21

performs better than LSTM and GRU as shown in Table 4. In SDE-TGAT, temporal attention works on latent stochastic trajectories so as to further enhance the discrimination between abnormal evolution and routine noise.

10.3 How GAT: spatial dependency modelling improves

Graph attention allows detecting faults as neighbors provide evidence to understand the context. In the event that one node is reporting an abnormal temperature, and surrounding nodes and communication measurements continue to show normal results, it is more likely that the temperature sensor is at fault than that an environmental event is to blame. On the other hand, if the local cluster changes together, mark of sensor fault should not happen on each and every change. Unlike fixed graph convolution, GAT allows for the neighbour weighting to be performed differently at each node. This behaviour is echoed by the ablation result that removal of GAT results in the biggest space-context penalty.

10.4 Summary on Contributions to the WSN, AI and their Integration

The proposed model is most suitable to deploy them at a cluster head, gateway or an edge server for practical IWSNs. Humans can compose feature windows and send them to the edge device directly or have the sensor nodes send them to the edge device, where they undertake SDE-TGAT inference and return the alert states to the network management system. This location involves a compromise between accuracy and the cost of energy. Class probability, confidence, node id, and neighbouring evidence based on attention should be provided in the output for the sake of operator interpretation. In the safety-critical setting, SDE-TGAT should also be complemented with more conservative fail-safe guidelines to keep a model failure from escalating to be a system failure in a lab coat.

11. Ablation Study

The ablation study in Table 6 was designed to investigate if the improvement of the proposed model is mainly due to a single dominant component or a combination of stochastic, temporal and graph learning. The results represent complementary impacts. This model without SDE also obtains temporal-spatial attention without SDE though, it is less stable in noisy windows. Without Transformer it is missing long-range temporal reasoning. Not using GAT it loses neighborhood context and makes more out-of-the-spot decisions on faults. If the attention fusion was not performed, an adaptive weighting of stochastic, temporal and spatial evidence was not possible with the model. The full configuration is better because of a supervised predictive objective it seeks to combine all three types of information.

12. Complexity Analysis

Let N be the number of graph nodes; L is the window length, d_m be the hidden dimension; |E| is the number of graph edges; and H_T and H_G are the number of heads used in the Transformers and graph attention networks respectively, while S is the number of SDE solver steps per window. As soon as drift and diffusion are going to be approximated by small MLP, the propagation cost of the neural SDE is approximately O(S N d_m^2). If you are using the Transformer encoder with per node attention, the attention complexity turns out to be O(N L^2 d_m) per-layer. The complexity of the GAT module is O(|E| H_G d_m) per layer. The fusion and prediction heads have a cost of O(N d_m C) which is relatively small.

Table 9. Computational complexity of SDE-TGAT components.

Component	Complexity	Scalability implication
SDE latent propagation	O(S N d_m^2)	Depends on solver steps and MLP size
Transformer temporal encoder	O(N L^2 d_m)	Dominant when window length is large
GAT spatial encoder	O(|E| H_G d_m)	Dominant in dense graphs
Fusion and classifier	O(N d_m C)	Small relative to encoders
Memory footprint	O(N L d_m + |E|)	Stores latent windows and graph attention states

Table 9 shows that scalability is controlled mainly by the Transformer window length and graph density. Sparse WSN topologies are favourable because |E| grows approximately linearly with N when each node maintains a bounded neighbour count. For large deployments, scalable variants can use local attention, hierarchical clustering, temporal

10.48047/jocaaa.2026.35.06.21

downsampling, or edge partitioning. Inference is suitable for edge/cloud-assisted WSNs but not for ultra-low-power motes without compression. Quantization-aware training and knowledge distillation are recommended for embedded deployment.

13. Limitations:

- The biggest constraint is a simulated data set. Realistic fault mechanisms, topology and stochastic disturbances were taken into account, but simulation can neither reliably mimic the ageing of hardware nor the calibration history, the deterministic behaviour of firmware nor site-specific interference nor maintenance practices.
- Results should be interpreted as a Controlled Validation Study and not proof of 'field readiness'. In order to make claims on operation, real industrial WSN traces are required.
- Note that it is a more heavyweight architecture than classics classifiers. Can be considered for use with gateway or edge-server as well as fully deployed model, depending on size and may need to be pruned, quantized or split for usage in low power embedded nodes.
- Delay of transmission and overhead for aggregation of packets were simplified. Scheduling, routing, retransmission, and congestion will impact data freshness and alert latency in real networks.
- The graph construction approach requires topology/link quality/estimation of correlation. If the network is very dynamic, that can cause new things to be added in the graph, which could incur extra computation.
- Detection and classification of the different types of faults is present, but causal root cause localization that goes beyond attention-based evidence is not yet addressed. While attention weights are helpful, they do not get to the bottom of the matter.

14. Future Scope:

- To achieve Edge deployment, quantization, pruning, early-exit inference and distillation – transfer from SDE-TGAT to small student model – should be considered.
- Federated learning has the ability to train cluster-specific models without requiring the centralization of raw sensor data, making it applicable to deployments where the privacy of sensor data is important, such as in the industrial or smart-city domain.
- Adaptive SDE modelling should not be based on fixed values after training, but should be changed online with the change of the environment.
- Energy-aware fault detection tasks should work together to optimally trade off diagnostic performance and energy spent on communications, in the sense of determining who should send high-resolution windows.
- To avoid confusion between device faults and malicious false-data injection, jamming, replay and routing attacks, cyber-physical security integration should detect these attacks. Industrial, environmental and structural monitoring WSNs should be validated in real life with descriptive maintenance and/or failure histories.

15. Conclusion

In this manuscript, SDE-TGAT, a hybrid stochastic spatio-temporal learning framework for predictive fault detection in a intelligent wireless sensor networks, was developed. The method uses a graph attention network to learn topology-aware spatial dependencies, a Transformer encoder to capture long-range temporal dependencies and the neural stochastic differential equations to model the latent states of sensors. These representations are combined with a gated fusion module that results in node level fault probabilities and early-warning alerts. SDE-TGAT improved performance in a realistic scenario with 100 nodes simulated IWSN scenario and six fault types compared to the performance of SVM, Random Forest, LSTM, GRU, CNN-LSTM, Transformer, GCN, GAT, and Transformer-GAT without SDE. In full model, the accuracy, F1-score of 97.93%, AUC-ROC of 0.993, false alarm rate of 1.89%, and interval mean detection delay of 1.74 were accomplished. The ablation study demonstrated that stochastic latent modelling, temporal attention, graph attention, adaptive fusion all help achieve the performance. Even though the approach shows good promise towards edge/cloud-assisted deployments of IWSN, field validation, energy-aware

10.48047/jocaaa.2026.35.06.21

optimization, and even real-time adaptations of the graph would be still required as a prerequisite for an operational deployment of this approach.

References

[1] Z. Noshad, N. Javaid, T. Saba, Z. Wadud, M. Q. Saleem, O. E. Sheta, and A. H. Alkhayyat, "Fault detection in wireless sensor networks through the random forest classifier," Sensors, vol. 19, no. 7, p. 1568, 2019.

[2] R. Ahmad, I. Alsmadi, W. Alhalabi, and L. Aljohani, "Machine learning for wireless sensor networks security: An overview of challenges and solutions," Sensors, vol. 22, no. 13, 2022.

[3] H. Sharma, A. Haque, and F. Blaabjerg, "Machine learning in wireless sensor networks for smart cities: A survey," Electronics, vol. 10, no. 9, p. 1012, 2021.

[4] T. S. Delwar, M. F. Mridha, M. M. Hamid, J. Shin, and Y. Watanobe, "The intersection of machine learning and wireless sensor networks for enhancing cybersecurity," Sensors, vol. 24, no. 19, p. 6377, 2024.

[5] R. Khan, M. A. Jan, M. Alam, and S. Mastorakis, "FedLSTM: A federated learning framework for sensor fault detection in wireless sensor networks," Electronics, vol. 13, no. 24, p. 4907, 2024.

[6] R. Prasad, R. K. Barik, and H. Dubey, "Self-detection based fault diagnosis for wireless sensor networks," Ad Hoc Networks, 2023.

[7] B. S. Gouda, H. M. K. Alazemi, and M. R. Senouci, "Distributed fault detection in sparse wireless sensor networks," Pervasive and Mobile Computing, 2025.

[8] A. Alauthman, M. Almiani, M. Al-kasassbeh, and A. Alwadi, "Intelligent fault detection and self-healing mechanisms in wireless sensor networks," Computers, vol. 14, no. 6, p. 233, 2025.

[9] A. Abdulkarim and A. Boukerche, "Utilizing machine learning for sensor fault detection in wireless sensor networks," in Proc. IEEE Int. Conf. Communications Workshops, 2024.

[10] S. Cicero, A. Tagarelli, and A. Gullo, "A deep neural framework for fault detection in IoT-based sensor networks," in Proc. IEEE/ACM ASONAM, 2024.

[11] G. Dong, M. Tang, Z. Wang, J. Gao, S. Guo, L. Cai, R. Gutierrez, B. Campbell, L. E. Barnes, and M. Boukhechba, "Graph neural networks in IoT: A survey," ACM Transactions on Sensor Networks, vol. 19, no. 2, Article 47, 2023.

[12] N. X. Tung, D. T. Hoang, D. N. Nguyen, E. Dutkiewicz, and D. Niyato, "Graph neural networks for next-generation-IoT," arXiv preprint arXiv:2412.20634, 2025.

[13] M. Jin, Y.-F. Koh, Q. Wen, Y. Zambon, C. Alippi, and G. I. Webb, "A survey on graph neural networks for time series: Forecasting, classification, imputation, and anomaly detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 46, pp. 10466-10485, 2024.

[14] H. Qiao, H. Tong, B. An, I. King, C. Aggarwal, and G. Pang, "Deep graph anomaly detection: A survey and new perspectives," IEEE Transactions on Knowledge and Data Engineering, 2025.

[15] T. K. K. Ho, Y. Wang, and M. Zhang, "Graph anomaly detection in time series: A survey," IEEE Transactions on Pattern Analysis and Machine Intelligence, 2025.

[16] F. Ares-Robledo, P. García-Sánchez, and A. Fernández, "Graph neural networks for anomaly detection: A systematic review," Artificial Intelligence Review, 2026.

[17] M. Zhong, X. Yu, and C. Leckie, "A survey on graph neural networks for intrusion detection systems: Methods, trends and challenges," Computers & Security, 2024.

[18] P. Brahmbhatt, K. S. Patel, and V. K. Singh, "Improved fault detection and diagnosis using graph autoencoder for multivariate sensor data," Results in Engineering, 2024.

10.48047/jocaaa.2026.35.06.21

[19] H. Jo, J. Kim, and S. Park, "Edge conditional node update graph neural network for multivariate time-series anomaly detection," Information Sciences, 2024.

[20] A. Deng and B. Hooi, "Graph neural network-based anomaly detection in multivariate time series," in Proc. AAAI Conf. Artificial Intelligence, 2021.

[21] S. Guan, J. Li, and Y. Zhao, "GTAD: Graph and temporal neural network for multivariate time series anomaly detection," Entropy, vol. 24, no. 6, p. 759, 2022.

[22] F. Zeng, C. Huang, and M. Jin, "Multivariate time series anomaly detection with adversarial transformer architecture," Future Generation Computer Systems, 2023.

[23] J. Xu, H. Wu, J. Wang, and M. Long, "Anomaly Transformer: Time series anomaly detection with association discrepancy," in Proc. International Conference on Learning Representations, 2022.

[24] S. Tuli, G. Casale, and N. R. Jennings, "TranAD: Deep transformer networks for anomaly detection in multivariate time series data," Proceedings of the VLDB Endowment, vol. 15, no. 6, pp. 1201-1214, 2022.

[25] R. Baidya, M. P. Singh, and S. K. Ghosh, "Anomaly detection in time series data using reversible instance normalized anomaly transformer," Sensors, vol. 23, no. 22, p. 9272, 2023.

[26] K. -H. Lai, D. Zha, G. Zhou, and X. Hu, "Revisiting time series outlier detection: Definitions and benchmarks," in Proc. NeurIPS Datasets and Benchmarks, 2021.

[27] D. Yadav, S. M. M. Rahman, and A. K. Menon, "Transformer-based anomaly detection on multivariate time-series subledger data," in Proc. ACM KDD Workshop, 2023.

[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in Advances in Neural Information Processing Systems, 2017.

[29] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in Proc. International Conference on Learning Representations, 2017.

[30] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in Proc. International Conference on Learning Representations, 2018.

[31] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural ordinary differential equations," in Advances in Neural Information Processing Systems, 2018.

[32] X. Li, T.-K. L. Wong, R. T. Q. Chen, and D. Duvenaud, "Scalable gradients for stochastic differential equations," in Proc. AISTATS, PMLR, vol. 108, pp. 3870-3882, 2020.

[33] X. Li, T.-K. L. Wong, R. T. Q. Chen, and D. Duvenaud, "Scalable gradients and variational inference for stochastic differential equations," in Proc. AABI, 2020.

[34] P. Kidger, J. Foster, X. Li, H. Oberhauser, and T. Lyons, "Neural SDEs as infinite-dimensional GANs," in Proc. International Conference on Machine Learning, PMLR, vol. 139, pp. 5453-5463, 2021.

[35] S. Park, B. Park, M. Lee, and C. Lee, "Neural stochastic differential games for time-series analysis," in Proc. International Conference on Machine Learning, PMLR, vol. 202, 2023.

[36] L. Yang, X. Meng, and G. E. Karniadakis, "Neural network stochastic differential equation models with applications to time-series prediction," Applied Mathematical Modelling, 2023.

[37] K. Zakharov, "Multivariate time series modelling with neural SDE driven by jump diffusion," in Proc. International Conference on Computational Science, 2024.

[38] Q. Wen, T. Zhou, C. Zhang, W. Chen, Z. Ma, J. Yan, and L. Sun, "Transformers in time series: A survey," in Proc. International Joint Conference on Artificial Intelligence, 2023.

[39] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient Transformer for long sequence time-series forecasting," in Proc. AAAI Conf. Artificial Intelligence, 2021.

[40] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition Transformers with auto-correlation for long-term series forecasting," in Advances in Neural Information Processing Systems, 2021.

10.48047/jocaaa.2026.35.06.21

[41] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "FEDformer: Frequency enhanced decomposed Transformer for long-term series forecasting," in Proc. International Conference on Machine Learning, 2022.

[42] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735-1780, 1997.