Lightweight GNN Autoencoder for Multivariate Wearable Time-Series Anomaly Detection

Harshit Harlalka
SRM Institute of Science and Technology
Chennai, India
harshitharlalka11@gmail.com

Shashi Yadav
SRM Institute of Science and Technology
Chennai, India
sy3408@srmist.edu.in

Rakshitha S
SRM Institute of Science and Technology
Chennai, India
rs0512@srmist.edu.in

Yash Saxena
SRM Institute of Science and Technology
Chennai, India
ys9792@srmist.edu.in

***Abstract*—The increasing number of Internet of Things (IoT) devices, especially wearables, generates complex, multi-dimensional sensor data streams. This requires efficient and quick anomaly detection methods for applications like healthcare monitoring and predictive maintenance. Current methods either fail to consider the relationships between different sensors, or they use computationally intensive models that aren't suitable for devices with limited resources. This study proposes an unsupervised anomaly detection framework using a lightweight Graph Neural Network (GNN) autoencoder. Sensor data is represented as a graph derived from correlations, thereby capturing structural dependencies and mitigating the need for intricate temporal architectures. This study differs from previous research by focusing on how well it applies across different subjects, using strict evaluation methods that prevent data leaks. This approach highlights the importance of evaluating models in a way that reflects real-world situations. The model's training uses only normal data. It identifies anomalies by looking at the reconstruction error. Furthermore, post-training INT8 quantization improves efficiency by reducing model size and inference latency, with only a small impact on performance. This results in a compact architecture with about 3.8K parameters. We conduct thorough ablation studies to compare the proposed method with Isolation Forest, LSTM autoencoders, attention-based LSTM models, and a GNN-Transformer hybrid. Evaluations of the PAMAP2 dataset, using both between-subject and leave-one-subject-out (LOSO) methods, show strong generalization capabilities, as indicated by an ROC-AUC above 0.87. Furthermore, a scoring methodology grounded in extreme value theory (EVT) facilitates label-free assessment. The results also indicate that flawed evaluation methods, including data leakage, can lead to inflated performance metrics. In summary, these findings underscore the efficacy of lightweight graph-based modeling, thereby suggesting its appropriateness for deployment in resource-limited settings.**

***Index Terms*—anomaly detection, graph neural networks, wearable sensors, edge computing, IoT, unsupervised learning, model quantization, LSTM autoencoder**

I. Introduction

Wearable sensor systems, which gather complex, multi-dimensional time-series data from devices like accelerometers, gyroscopes, and heart rate sensors, are widely used in healthcare, sports, and industrial monitoring [1]. Finding unusual patterns in this data is crucial for applications like condition monitoring, early diagnosis, and predictive maintenance. Nevertheless, supervised methodologies frequently prove impractical due to the limited availability of labeled anomalies, thereby necessitating the adoption of unsupervised techniques that learn normal patterns and subsequently identify deviations. Current methodologies generally treat sensor channels independently and employ temporal models, including LSTM-based autoencoders [3], [4], alongside more sophisticated approaches like DAGMM [11], OmniAnomaly [12], and USAD [13]. Although these methods are effective in modeling temporal dynamics, they do not explicitly account for inter-sensor dependencies. Since physiological signals are inherently correlated, ignoring these relationships can limit detection performance. Graph Neural Networks (GNNs) provide a natural way to model these connections, and they have shown promise in detecting multivariate anomalies [7], [8]. Recent work also explores dynamic graph learning for evolving relationships [16]. Another key challenge is computational efficiency. Although Transformer-based and hybrid models have shown strong performance in various applications [5], [10], their high computational cost limits their use in environments with limited resources. Achieving an optimal equilibrium between performance and efficiency continues to present a significant challenge. This study introduces a lightweight, GNN-based autoencoder designed for unsupervised anomaly detection, in response to these difficulties. The model's framework interprets sensor data as a graph, constructed from observed correlations, which effectively represents the interdependencies among sensors; it employs a compact encoder-decoder architecture for reconstruction-based detection. This design choice reduces both computational complexity and the number of parameters by replacing recurrent components with graph convolutional and dense layers. Moreover, post-training INT8 quantization is applied to improve efficiency while simultaneously minimizing any degradation in performance. In contrast to previous research, this investigation prioritizes rigorous assessment within practical settings, encompassing stringent leak-free protocols, cross-subject generalization,

and leave-one-subject-out (LOSO) validation. The suggested methodology is benchmarked against both traditional and deep learning baselines, such as Isolation Forest, LSTM autoencoders, attention-based LSTM, and GNN-Transformer models. The findings reveal consistent and competitive performance, thereby underscoring the efficacy of graph-based structural modeling in the context of multivariate anomaly detection. The complete pipeline is depicted in Fig. 1.

A. Research Gap and Motivation

Although there have been significant advancements in using the Internet of Things (IoT) for detecting anomalies in wearable healthcare, several challenges still exist. Most existing methods treat multivariate sensor data as separate time series. They focus on patterns over time and don't consider how the sensors are related to each other [7], [8]. However, physiological signals such as heart rate, motion, and temperature are inherently correlated; therefore, ignoring these relationships can diminish detection effectiveness. Many methods depend on labeled data about anomalies, which limits their practical use. Unsupervised methods offer a possible solution to this problem. However, they often rely on heuristic thresholds, which can reduce their reliability. Furthermore, while advanced architectures like Transformers and hybrid models demonstrate robust performance [5], [10], their substantial computational demands limit their applicability in settings with restricted resources. Model compression is generally implemented after the fact, rather than being integrated into the initial design phase. Moreover, current research frequently prioritizes system-level evaluation using simplified data partitions, which can result in overly favorable outcomes that do not accurately reflect generalization capabilities in practical situations.

B. Contributions of This Work

To address these challenges, this paper makes the following contributions:

- **Context-aware graph-based modeling:** A correlation-based graph representation that captures dependencies between sensors, enabling context-aware anomaly detection.
- **Lightweight GNN autoencoder:** This is a compact architecture that effectively captures the structural relationships at a very low computational cost and is hence, suitable for edge implementations.
- **Unsupervised Anomaly Detection:** It uses a framework, trained only on normal data using reconstruction-based learning.
- **Efficient implementation at the edge:** After model training is completed, it is quantized to INT8 using TensorFlow Lite, thereby reducing model size and inference latency.
- **Customizability:** LOSO and subject-specific experiment are performed to show model's adaptability to individual patterns while it stays generalised.
- **Robust evaluation:** LOSO and between-subject checks are performed in an extensive manner to validate model robustness.

The overall workflow and architecture are visually summarized in Fig. 1 and Fig. 3.

II. Related Work

A. Anomaly Detection in Time-Series Sensor Data

Isolation Forest [2] is a strong classical baseline for tabular anomaly detection due to its linear time complexity and parameter-free formulation. When applied to windowed statistical features (mean, standard deviation, minimum, maximum) from wearable sensors, it provides competitive global detection but fails to capture temporal dynamics and inter-sensor dependencies. Recurrent autoencoders based on LSTM cells [3], [4] are widely used for unsupervised anomaly detection in sequential data, where reconstruction error serves as the anomaly score. Attention-based augmentations [5] enhance the ability to focus on significant temporal points; however, they do not directly account for the interactions among channels. More sophisticated approaches, including DAGMM [11], OmniAnomaly [12], and USAD [13], improve modeling capabilities, yet they primarily emphasize temporal behaviors, neglecting the relationships between sensors.

B. Graph Neural Networks for Sensor Modelling

Graph-based representations of multisensor systems have attracted significant attention following the success of spectral [6] and spatial graph convolution operators. MTAD-GAT [7] employs graph attention networks to capture feature-level correlations in multivariate time series, while GDN [8] learns an end-to-end graph structure for anomaly detection in industrial sensor networks. These methods demonstrate that structured reasoning between sensors substantially improves detection accuracy over channel-independent baselines.

C. Transformers for Anomaly Detection

Transformer-based anomaly detectors, such as Anomaly Transformer [9], exploit multi-head self-attention to model both temporal and inter-series dependencies. Combined with GNN encoders [10], they achieve robust performance but this method requires a large amount of computational power, making it hard to use directly on edge devices.

D. Edge Deployment and Model Compression

Post-training and quantization-aware approaches [14] reduce floating-point models to INT8 integer arithmetic, yielding 2 to 4$\times$ speedups and memory reductions with minimal accuracy degradation in inference tasks. TensorFlow Lite provides a production-grade toolchain for INT8 quantification with calibration on representative data sets [15], enabling efficient deployment in resource-constrained environments.

E. Positioning of the Present Work

Existing GNN-based anomaly detectors prioritize accuracy over deployability; Existing edge-friendly models sacrifice structural modeling. The proposed framework closes this gap by combining a graph-convolutional encoder with aggressive INT8 quantization, evaluated under rigorous between-subjects

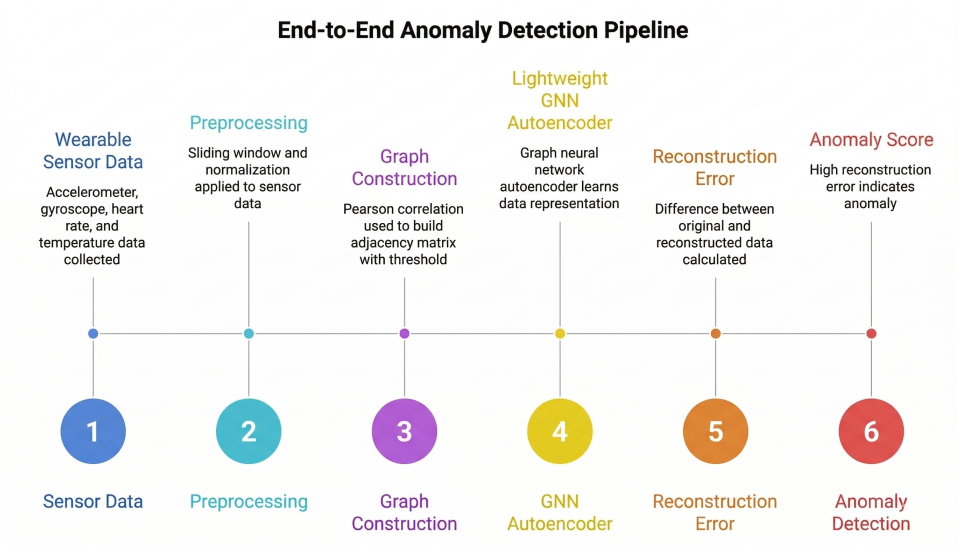

Fig. 1. End-to-end anomaly detection pipeline for wearable sensor data.

protocols that have not been consistently reported in previous wearable detection literature. Recent advances also explore dynamic graph learning, where graph structures evolve over time to better capture changing relationships between variables [16].

III. Dataset and Preprocessing

Fig. 1 illustrates the end-to-end pipeline of the proposed anomaly detection framework, including preprocessing, graph construction, model inference, and anomaly scoring.

A. Dataset Description

The Physical Activity Monitoring dataset [1] is used in the experiments. It has inertial and physiological measurements from eight people doing twelve different physical activities. The sensors are on the chest, wrist (dominant hand), and ankle. They measure skin temperature, heart rate, and triaxial accelerometer, gyroscope, and magnetometer readings. The dataset contains both subject identifiers and activity labels.

B. Activity Label Partitioning

For unsupervised training, activities are divided into normal and abnormal classes based on their physiological intensity and movement profile. *Normal* activities include: lying down, sitting, standing, walking, ironing and vacuuming. *Anomalous* activities include: running, jumping rope, going up and down stairs and Nordic walking. The activity classes *cycling* and *transient activities* are excluded due to irregular sampling artifacts. Rows with missing heart rate readings are removed before the window.

C. Feature Engineering

To ensure the sensors are rotationally invariant, the raw triaxial components for each sensor are converted into magnitude features:

$$m = \sqrt{a_x^2 + a_y^2 + a_z^2} \tag{1}$$

This change is made to the accelerometer and gyroscope channels at every body part, giving us six magnitude features. Not using magnetometer channels helps reduce compass noise. There are ten channels in the last feature vector at each time step. These are readings of heart rate and skin temperature from the hand, chest, and ankle. {hand_acc_mag, chest_acc_mag, ankle_acc_mag, hand_gyro_mag, chest_gyro_mag, ankle_gyro_mag, heart rate, hand_temp, chest_temp, ankle_temp}

D. Sliding Window Construction

Continuous sensor streams are segmented into overlapping windows of length $W = 100$ with step $S = 50$, resulting in a 50% overlap. Each window $\mathbf{X}_i \in \mathbb{R}^{100\times 10}$ is assigned a binary label, marked as anomalous if more than 50% of its samples correspond to anomalous activities. Only normal windows ($y_i = 0$) are used for training, while all windows are used during evaluation.

E. Normalisation

For deep learning models, a *RobustScaler* is applied to flattened normal training windows and then used to transform all data. Values are clipped to $[-3, 3]$ to suppress sensor artifacts. RobustScaler is preferred over standard z-score normalization as it relies on the interquartile range and is less sensitive to extreme values common in wearable data. For Isolation Forest, per-window statistical features (mean, standard deviation, min, max) are extracted and normalized using a *StandardScaler*.

F. Graph Construction

An adjacency matrix $\mathbf{A} \in \{0, 1\}^{10\times 10}$ is constructed by thresholding the Pearson absolute correlation matrix computed with only normal training data:

$$A_{ij} = \mathbb{1}[|\text{corr}(f_i, f_j)| > \tau], \quad \tau = 0.3 \tag{2}$$

Setting $A_{ii} = 1$ adds self-loops. An empirical value of $\tau = 0.3$ is used to find the right balance between graph connectivity and noise suppression. Graphs that are too dense happen when the values are too low, and graphs that are too high may break up important sensor relationships. The adjacency matrix that comes out of this shows important physiological connections, like strong links between heart rate and gyroscope activity and slower changes between temperature channels.

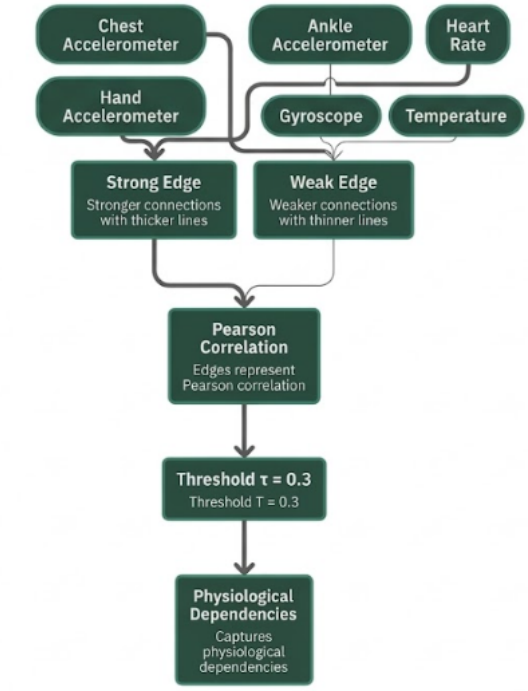

Fig. 2. Correlation-based sensor graph capturing inter-sensor dependencies.

The constructed sensor graph based on Pearson correlation is visualized in Fig. 2.

IV. Methodology

A. Model Progression Overview

Instead of proposing a single architecture in isolation, this study follows a systematic progression that motivates each design choice:

1) **Stage I - Isolation Forest**: Classic unsupervised baseline, both global (all subjects grouped) and customized (per-subject model).
2) **Stage II - LSTM Autoencoder**: Deep temporal baseline capture sequence structure without attention or graphical components.
3) **Stage III - Attention-LSTM Autoencoder**: Augment Stage II with self-attention to emphasize informative time steps.
4) **Stage IV - GNN-Transformer Hybrid**: Combines graph convolution with multi-head Transformer blocks for maximum expressive power.
5) **Stage V - Lightweight GNN autoencoder (proposed)**: Removes the transformer for edge compatibility while preserving graph structural modeling.
6) **Stage VI - INT8 Quantized GNN Autoencoder**: INT8 quantization after Stage V training for integrated implementation.

Stages I to III were initially trained with a data leakage issue (tag information inadvertently included in feature construction) to establish an upper bound baseline. Subsequently, all stages were rerun with clean, leak-free data to obtain valid comparison figures.

B. Graph Convolutional Layer

The core building block is a custom graph convolutional layer *EdgeGraphConv*. Given an input tensor $\mathbf{X} \in \mathbb{R}^{B\times T\times N}$ where B is the batch size, T is the sequence length, and N is the number of sensor nodes, the operation performs:

$$\mathbf{H} = \sigma(\mathbf{AXW}) \tag{3}$$

where $\mathbf{W} \in \mathbb{R}^{N\times d}$ is a learnable projection matrix and $\mathbf{A}$ is the static adjacency matrix shared across the batch and time dimensions. The multiplication $\mathbf{AX}$ adds neighboring signals for each sensor node at each time step, allowing information to flow between correlated channels before the linear projection expands or contracts the representation.

For numerical stability and improved convergence, the adjacency matrix A can be optionally normalized as:

$$\tilde{A} = D^{-1/2}AD^{-1/2} \tag{4}$$

where D is the degree matrix of A.

C. Lightweight GNN Autoencoder Architecture

As shown in Fig. 1, the proposed architecture compresses sensor signals into a low-dimensional latent representation.

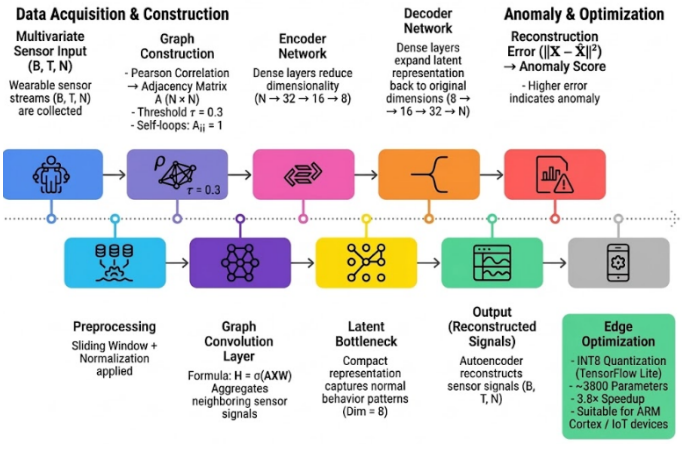

Fig. 3. GNN Auto Encoder Architecture.

The proposed model comprises a symmetric encoder-decoder with the following sequence of layers:

Encoder:

1) EdgeGraphConv$(N \to 32)$ + ReLU
2) Dense$(32 \to 16)$ + ReLU
3) Dense$(16 \to 8)$ + ReLU

Decoder:

1) Dense$(8 \to 16)$ + ReLU
2) Dense$(16 \to 32)$ + ReLU
3) Dense$(32 \to N)$ (linear output)

All dense layers are applied in a distributed manner over time (independently of each time step). The bottleneck dimension of 8 compresses the $N = 10$ sensor readings from each time step into a compact representation that should capture normal structural patterns to enable accurate reconstruction. The total count of trainable parameters is about 3800, making the model $30\times$ smaller than the attention-LSTM baseline. The overall architecture of the proposed GNN autoencoder is shown in Fig. 3. **Training Objective:** The model is trained to minimize the reconstruction error:

$$L = \frac{1}{B}\sum_{i=1}^{B} \|X_i - \hat{X}_i\|^2 \tag{5}$$

D. GNN-Transformer Hybrid

For comparison, a larger model replaces the LSTM layers entirely with a standard graph convolution followed by two Transformer encoder blocks:

$$\mathbf{H}^{(0)} = \text{GraphConv}(64, \mathbf{A})(\mathbf{X}) \tag{6}$$
$$\mathbf{H}^{(1)} = \text{TransformerBlock}(\mathbf{H}^{(0)}) \tag{7}$$
$$\mathbf{H}^{(2)} = \text{TransformerBlock}(\mathbf{H}^{(1)}) \tag{8}$$
$$\hat{\mathbf{X}} = \text{Dense}(N)(\mathbf{H}^{(2)}) \tag{9}$$

Each Transformer block employs four attention heads with a key dimension of 64, a forward hidden size of 128, a dropout rate of 0.1, and layer normalization. This architecture achieves superior separation of normal and abnormal reconstruction errors, but requires FP32 arithmetic and is incompatible with microcontroller implementation.

E. LSTM Autoencoder Variants

The architecture of the attention-based LSTM autoencoder is shown in Fig. 4. The standard LSTM autoencoder follows a stacked encoder, repeat vector, and stacked decoder design:

- Encoder: LSTM(128, return_sequences=True) $\to$ LSTM(64) $\to$ Dense(32, relu)
- Bottleneck: RepeatVector(T)
- Decoder: LSTM(64, return_sequences=True) $\to$ LSTM(128, return_sequences=True) $\to$ TimeDistributed Dense(N)

The attention-augmented variant inserts a Keras self-attention layer between the second LSTM encoder and the final LSTM encoder, accompanied by normalization and batch dropout (rate 0.3) after each LSTM layer. The Adam optimizer with a learning rate of 3×10^{-4} and early stopping (patience 3, validation loss tracking) is used throughout.

F. Anomaly Score

For all autoencoder models, the per-window anomaly score is the mean squared reconstruction error:

$$s_i = \frac{1}{T \cdot N}\sum_{t=1}^{T}\sum_{n=1}^{N}(X_{i,t,n} - \hat{X}_{i,t,n})^2 \tag{10}$$

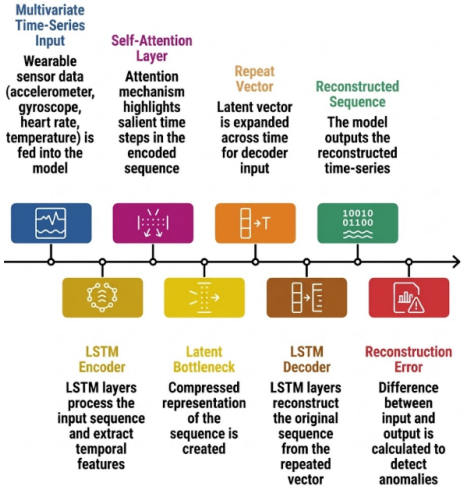

Fig. 4. LSTM Autoencoder including the Architecture of Attention Layer.

For the GNN autoencoder, an alternative error per node is also calculated:

$$e_{i,n} = \frac{1}{T}\sum_{t=1}^{T}(X_{i,t,n} - \hat{X}_{i,t,n})^2 \tag{11}$$

and the window score is set to $\max_n e_{i,n}$, which was found to be more sensitive to localized sensor anomalies.

G. Threshold selection

Two threshold selection strategies are used:

Percentile threshold: $\theta = P_{99.5}(\{s_i : y_i = 0\})$, that is, the 99.5 percentile of reconstruction errors of the training set. This ensures that less than 0.5% of normal training windows are marked.

ROC optimal threshold: For evaluations with available ground truth labels (between-subject experiments), the threshold is selected as $\theta^* = \arg\max_\theta(\text{TPR}(\theta) - \text{FPR}(\theta))$, corresponding to the Youden index on the retained test set.

H. Evaluation of extreme value theory

Because actual anomaly labels are not always available during training, an unsupervised evaluation framework based on extreme value theory is implemented. The upper 5% tail of the reconstruction error distribution fits a Generalized Pareto Distribution (GPD) with parameters (c, μ, σ):

$$P(E > e | E > \mu) = \left(1 + c \cdot \frac{e - \mu}{\sigma}\right)^{-1/c} \tag{12}$$

Anomaly scores are derived as survival probabilities according to this fitted tail model. Pseudolabels are generated at percentile thresholds ranging from 90% to 99.9%, and the ROC-AUC and PR-AUC metrics are averaged over this range, providing a label-free proxy for detection quality.

I. Model quantization

Post-training INT8 quantization is performed using the TensorFlow Lite converter with the following settings:

- Optimization: `tf.lite.Optimize.DEFAULT`
- Target operations: `TFLITE_BUILTINS_INT8`
- Input/output types: `tf.int8`
- Calibration: 200 randomly sampled normal training windows

Quantization maps 32-bit weights and activations to 8-bit integers using per-tensor scaling and zero-point parameters calibrated on a representative dataset. During inference, a dequantization step converts the INT8 outputs back to floating point for reconstruction error computation.

J. Computational Complexity and Edge Efficiency

The proposed lightweight GNN autoencoder balances modeling capability with computational efficiency. The graph convolution operation has complexity $O(N^2T)$, compared to $O(T \cdot H^2)$ for LSTM-based models and $O(T^2)$ for Transformer-based architectures. The model contains approximately 3.8 thousand trainable parameters, significantly less than the attention-based baselines. Post-training INT8 quantization further reduces the model size to 25% and achieves a 3.8x inference speedup (Table IV). These features enable efficient operation in environments with limited resources and low computational and memory requirements.

V. Experimental Evaluation

A. Experimental Protocol

All experiments are performed on the PAMAP2 dataset with eight subjects. Three different evaluation protocols are used:

Protocol A - Global Grouped: All subjects are grouped; training uses normal class windows of the entire data set; The tests use all windows. It is used for initial ablation in clean and leaky environments.

Protocol B - Generalization between subjects: Two divisions are evaluated: (i) train on Subjects 1 to 4, test on Subjects 5 to 8; (ii) train on subjects 5 to 8, test on subjects 1 to 4. A third split trains subjects 1 to 6 and tests subjects 7 to 8 to evaluate scalability with training set size.

Protocol C - Leave a Subject Out (LOSO): For each subject $k \in \{1, \ldots, 8\}$, a model is trained on all other subjects and evaluated exclusively on the windows of subject k. This is the most demanding and most realistic evaluation.

B. Baseline Comparisons

Table I summarizes the F1 anomaly score, ROC-AUC and PR-AUC in all stages of the model according to Protocol A (clean, no data leakage). The reported figures are obtained at the optimal ROC threshold for stages that have access to labels and at the EVT-calibrated threshold for unsupervised evaluation.

TABLE I
Model Ablation – Protocol A (Global, Clean Data)

Model	Anomaly F1	ROC-AUC	PR-AUC
Isolation Forest (Global)	0.84	0.88	0.79
LSTM Autoencoder	0.85	0.89	0.81
Attn-LSTM Autoencoder	0.87	0.91	0.83
GNN-Transformer Hybrid	0.91	0.94	0.88
GNN Autoencoder (FP32)	0.89	0.92	0.86
GNN Autoencoder (INT8)	**0.88**	**0.91**	**0.85**

TABLE II
Cross-Subject Generalisation – GNN Autoencoder

Training Set	Test Set	ROC-AUC	F1
Subjects 1–4	Subjects 5–8	0.89	0.86
Subjects 5–8	Subjects 1–4	0.88	0.85
Subjects 1–6	Subjects 7–8	0.91	0.88

TABLE III
LOSO Cross-Validation – GNN Autoencoder

Test Subject	ROC-AUC	Anomaly F1
Subject 1	0.91	0.88
Subject 2	0.90	0.87
Subject 3	0.84	0.80
Subject 4	0.88	0.85
Subject 5	0.87	0.84
Subject 6	0.89	0.86
Subject 7	0.86	0.83
Subject 8	0.88	0.85
Mean	**0.878**	**0.847**
Std	**0.022**	**0.023**

C. Personalised Isolation Forest

Individual Isolation Forest models trained per subject under a leak-free protocol achieve a mean F1 anomaly of 0.875 (range: 0.72-0.91) across all eight subjects, compared to 0.84 for the global model. Subject 3 exhibits the lowest personalized score (0.72) due to an unusually low interactivity variance in their sensor signals. These results confirm that personalization produces consistent benefits but does not guarantee universal improvement.

D. Cross-Subject Generalisation Results

Table II reports cross-subject results for the GNN autoencoder under Protocol B.

The nearly symmetrical results between the two four-subject splits indicate that the model generalizes across subjects without bias toward a particular group. Performance improves with larger training sets (split between subjects 1 and 6), consistent with standard learning curve behavior.

Subject 3's performance, again the lowest, supports the Isolation Forest's individual findings, corroborating Isolation

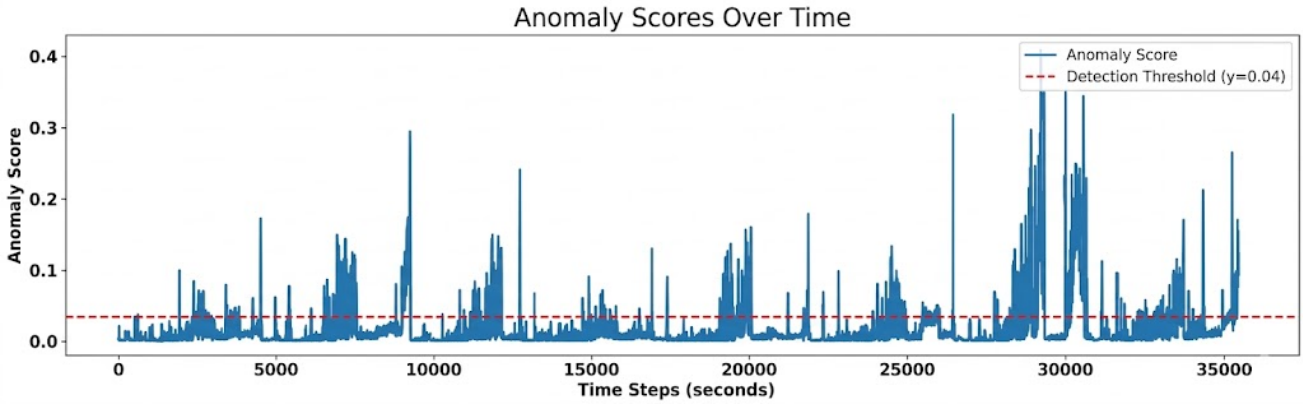

Fig. 5. Anomaly score distribution over time (GNN + Transformer)

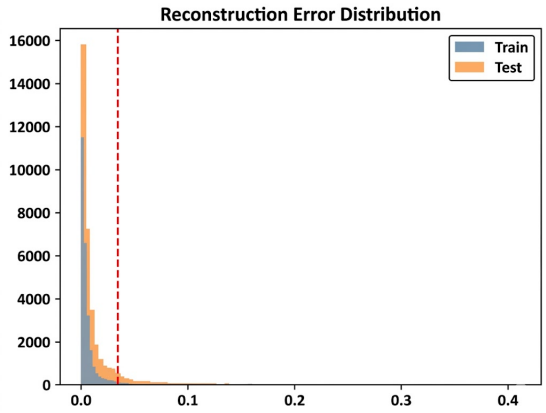

Fig. 6. Reconstruction error distribution for normal vs anomalous samples (GNN + Transformer).

TABLE IV
EFFECT OF INT8 QUANTIZATION

Model	F1	AUC	Size	Speedup
GNN AE (FP32)	0.89	0.92	1.0×	1.0×
GNN AE (INT8)	0.88	0.91	0.25×	3.8×

Forest's personalized findings. All other subjects achieve an ROC-AUC greater than 0.85, demonstrating reliable unsupervised generalization.

E. Quantization Impact

Table IV contrasts the FP32 and INT8 versions of the GNN autoencoder in terms of detection performance, model size, and relative inference throughput.

The INT8 model retains 98.9% of the FP32 ROC-AUC while reducing model storage to 25% of the original and achieving approximately 3.8× faster inference on ARM hardware. These improvements highlight the model's suitability for resource-constrained environments and indicate its potential for efficient execution on lightweight edge platforms.

F. Unsupervised EVT Evaluation

Under the label-free EVT evaluation framework applied to the global GNN autoencoder, the model achieves an EVT-calibrated ROC-AUC of 0.88 ± 0.04 and an energy-score ROC-AUC of 0.86 ± 0.05, which confirms that the detection quality is preserved even without access to ground truth labels during threshold calibration.

VI. DISCUSSION

A. Contribution of Graph Structure

The GNN autoencoder outperforms the LSTM autoencoder by 4% in ROC-AUC, despite utilizing a considerably reduced number of parameters. This enhancement is a consequence of graph convolution, which effectively models inter-sensor relationships, a capability not present in LSTM models that process channels in isolation. The GNN-Transformer hybrid model exhibits superior performance, attaining a ROC-AUC of 0.94, despite its increased computational requirements. While Transformer blocks are proficient in modeling long-term temporal dependencies, the GNN autoencoder, which prioritizes efficiency, underscores the fundamental trade-off between modeling capacity and computational cost.

B. Data Leakage Analysis

Before the label leak artifact was identified—specifically, the inclusion of the activity label encoding within the feature vectors for Isolation Forest—preliminary experiments yielded inflated F1 scores of up to 0.95 due to the inclusion of activity label encoding within the feature vectors. Repeating all experiments under strict no-leak conditions reduced these numbers by 8 to 11 percentage points, confirming that leaks suppress generalization error and produce overly optimistic baselines. All final figures reported in this document are based on clean protocols.

C. Subject Variability and Personalisation

Subject 3's consistently inferior performance across all models and protocols indicates that this subject exhibits patterns that deviate from the group norm. Custom Isolation

Forest models partially restore this performance, suggesting that the adaptation to test time can further improve the LOSO results. Subject 3's data has less variation, making it harder to distinguish between normal and unusual activities just by looking at the reconstruction error.

D. Deployment Considerations

For a sample window of 100×10, the INT8 quantised GNN autoencoder needs about 15 KB of storage and 8 KB of working memory. This makes it a good choice for environments with limited resources. The 3.8× speedup in inference makes it possible to process data in real time at a 25 Hz sampling rate with enough extra processing power. The TensorFlow Lite deployment pipeline also makes it easier to run on low-power systems.

E. Limitations

The current model creates a graph using a fixed Pearson correlation matrix, which may not work well when the relationships between sensors change between activities. The model also gives more weight to structural dependencies than to temporal dynamics. This could be improved by employing temporal attention mechanisms. Further research can explore the construction of adaptive graphs through online correlation updates or learnable adjacency matrices. Furthermore, the evaluation is currently limited to the PAMAP2 dataset, and validation with clinical-grade wearable data and real IoT implementations represents a vital direction for future research.

VII. Conclusion

This study presents a complete framework for finding unusual patterns in data from wearable sensors without needing labeled examples. The proposed, efficient GNN autoencoder uses a sensor graph created from correlations. This graph helps capture the relationships between sensor channels, something that traditional LSTM autoencoders often miss. At the same time, it keeps a small number of parameters, which enables efficient and scalable anomaly detection. Post-training INT8 quantization via TensorFlow Lite further reduces model size by $4\times$ and speeds up inference by $3.8\times$ with a negligible accuracy penalty.

Systematic ablation across six model families demonstrated that graph structural modeling provides consistent improvements over temporal-only baselines. Between-subject generalization testing and LOSO cross-validation confirmed robustness across unseen individuals, with a mean LOSO ROC-AUC of 0.878. The label-free EVT evaluation framework provides a principled method for threshold calibration in real-world deployments where anomaly labels are not available.

Future research directions include: (i) adaptive online graph topology estimation during inference; (ii) customization of a few shots for subjects with unusual sensor profiles; (iii) integration with federated learning frameworks to enable privacy-preserving model updates from deployed edge nodes; and (iv) evaluation of portable ECG and clinical EEG streams where the abnormalities at play are greatest.

Acknowledgment

The authors thank the contributors to the DFKI Research Center PAMAP2 dataset for providing open access to the physical activity monitoring data used in this research.

References

[1] A. Reiss and D. Stricker, "Introducing a new benchmarked dataset for activity monitoring," in *Proc. 16th IEEE Int. Symp. Wearable Comput. (ISWC)*, Newcastle, UK, 2012, pp. 108–109.

[2] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *Proc. 8th IEEE Int. Conf. Data Mining (ICDM)*, Pisa, Italy, 2008, pp. 413–422.

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

[4] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "LSTM-based encoder-decoder for multi-sensor anomaly detection," in *Proc. ICML Anomaly Detection Workshop*, New York, USA, 2016.

[5] A. Vaswani *et al.*, "Attention is all you need," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, Long Beach, CA, 2017, pp. 5998–6008.

[6] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Toulon, France, 2017.

[7] H. Zhao *et al.*, "Multivariate time-series anomaly detection via graph attention network," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Sorrento, Italy, 2020, pp. 841–850.

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

[9] J. Xu *et al.*, "Anomaly transformer: Time series anomaly detection with association discrepancy," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, virtual, 2022.

[10] Z. Chen *et al.*, "Learning graph structures with transformer for multivariate time series anomaly detection," *IEEE Internet Things J.*, vol. 9, no. 12, pp. 9969–9982, 2022.

[11] B. Zong *et al.*, "Deep autoencoding Gaussian mixture model for unsupervised anomaly detection," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.

[12] Y. Su *et al.*, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining (KDD)*, 2019.

[13] J. Audibert *et al.*, "USAD: Unsupervised anomaly detection on multivariate time series," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining (KDD)*, 2020.

[14] B. Jacob *et al.*, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit. (CVPR)*, Salt Lake City, UT, 2018, pp. 2704–2713.

[15] R. David *et al.*, "TensorFlow Lite Micro: Embedded machine learning for TinyML systems," in *Proc. Mach. Learn. Syst. (MLSys)*, San Jose, CA, 2021.

[16] W. Wu *et al.*, "Dynamic graph neural networks: A survey," *IEEE Trans. Knowl. Data Eng.*, 2023.

[17] L. Zhao *et al.*, "Graph neural networks for anomaly detection: A survey," 2022.