Detection of multi-vector attacks in IoT networks: a graph attention network-based approach*

Mykola Stetsiuk[1,*,†], Yurii Klots[1,†], Dmytro Tymoshchuk[2,*,†], Mikolaj Karpinski[3,†] and Nataliia Petliak[1,†]

[1] Khmelnytskyi National University, 11, Instytuts'ka str., Khmelnytskyi, 29016, Ukraine
[2] Ternopil Ivan Puluj National Technical University, 56, Ruska str. Ternopil, 46001, Ukraine
[3] University of the National Education Commission, 2 Podchorążych str, Krakow, 30084, Poland

Abstract

This paper addresses the challenge of detecting multi-vector attacks in Internet of Things (IoT) networks by leveraging Graph Attention Networks (GAT). A novel method is proposed for modeling IoT infrastructure as a weighted directed graph that captures not only the physical topology but also logical relationships and dynamic interactions between devices. Nodes represent heterogeneous IoT elements — including end devices, routers, and gateways — while edges denote communication channels with assigned weights based on telemetry features. The method introduces a structured representation of telemetry in the form of feature vectors extracted from time-series data, enabling the detection of behavioral deviations at both node and network levels. Multi-vector attacks are formalized as transformations of the graph's structure and dynamics, allowing for the analysis of complex propagation chains involving devices, links, and control nodes. A GAT-based architecture is developed to process these representations, applying attention mechanisms to identify relevant contextual dependencies in the graph. The proposed detection pipeline consists of a telemetry converter, an interpreter for node classification, and a training module based on binary cross-entropy loss. Experimental validation was conducted using real-world datasets, covering a wide range of attack scenarios and device types. Comparative analysis with other graph-based methods (GCN, GraphSAGE, GAE) demonstrates the competitiveness of GAT in terms of accuracy, F1 score, and ROC-AUC — achieving a ROC-AUC of 0.955 while maintaining interpretability and scalability.

Keywords

IoT security, anomaly detection, network traffic analysis, machine learning, GAT, GNN, intrusion detection, cybersecurity

1. Introduction

Internet of Things (IoT) devices are widely used in smart homes [1,2], agriculture [3,4], logistics [5,6], healthcare [7,8], city infrastructure [9,10], and for monitoring environmental parameters [11-13], enabling real-time interaction between the physical and digital levels. The rapid growth of the IoT has led to the formation of heterogeneous, dynamic environments with limited-resource devices, open interfaces, and minimal protection. This makes such systems vulnerable to multi-vector attacks that exploit various layers — from end nodes to routing and control logic. Traditional detection systems based on signatures or anomalies often prove ineffective due to delays in creating new signatures, a lack of labeled data, and frequent changes in topology. To overcome these limitations, machine learning-based approaches are a promising method. Machine learning methods have become widespread in various fields, including materials science [14–18], medicine [19–21], financial analytics [22–24], engineering [25–27], and cybersecurity [28–30].

* ITTAP'2025: 5h International Workshop on Information Technologies: Theoretical and Applied Problems, October 24-25, 2025, Ternopil, Ukraine, Opole, Poland
[1,*] Corresponding author.
[†] These authors contributed equally.
mykola.stetsiuk@khmnu.edu.ua (M. Stetsiuk); klots@khmnu.edu.ua (Y. Klots); dmytro.tymoshchuk@gmail.com (D. Tymoshchuk); mikolaj.karpinski@uken.krakow.pl (M. Karpinski); npetlyak@khmnu.edu.ua (N. Petliak)
0000-0003-3875-0416 (M. Stetsiuk); 0000-0002-3914-0989 (Y. Klots); 0000-0003-0246-2236 (D. Tymoshchuk); 0000-0002-8846-332X (M. Karpinski); 0000-0001-5971-4428 (N. Petliak);

Approaches to detecting intrusions in IoT networks increasingly rely on deep learning. A promising direction is the use of graph neural networks (GNNs). Among them, graph attention networks (GAT) provide selective aggregation of information from neighboring nodes. This paper proposes a GAT-based model for detecting multi-vector attacks in IoT networks by combining graph representations with device telemetry. The method formalizes both physical and logical topologies to detect anomalies in nodes and interactions between devices using real IoT data with attack scenarios.

2. Overview of detection and protection methods

To develop an effective approach for detecting multivector attacks in IoT networks, existing research in this field has been analyzed.

In the article [31], the authors propose the Top-K Similarity Graph Framework (TKSGF) for detecting intrusions in IoT networks. Unlike traditional graph methods based on physical connections, TKSGF constructs graphs based on Top-K attribute similarity, which improves node representation. GraphSAGE is used as a GNN model for scalable training. The influence of graph directionality, K value, and GNN architectures is investigated. Experiments on the NF-ToN IoT and NF-BoT IoT datasets show that TKSGF outperforms traditional ML methods and other graph-based approaches in terms of accuracy and robustness. In the paper [32], the authors propose a new approach to detecting intrusions in IoT networks by combining Self-Supervised Learning (SSL) and Markov Graph Convolutional Network (MarkovGCN). This approach allows for effective work with unbalanced data and new attacks with a small number of samples. SSL reduces dependence on large labeled sets, while MarkovGCN detects network structures and enriches node and edge features with context. Experiments on the EdgeIIoT-set dataset showed high performance (Accuracy – 98.68%, Precision – 98.18%, Recall – 98.35%, F1 – 98.40%), outperforming traditional ML methods. In the article [33], the authors analyze current methodologies for labeling network data in the field of cybersecurity, emphasizing that most open datasets quickly become obsolete due to changes in attacker behavior and the complexity of the labeling process. Common automated methods generate synthetic traffic, hiding key features for distinguishing between normal and malicious activity, while approaches involving non-experts are limited by data quality and volume. The authors emphasize the need to develop a consistent labeling methodology for the continuous creation of representative datasets, which is critical for the implementation of new ML and statistical threat detection methods. In the study [34], the authors propose a new approach to IoT security, using a graph algorithm to build a network model and then evaluate it using a GAT-based intrusion detection system (IDS). This approach allows for the complex interrelationships between IoT nodes to be taken into account. The NSL-KDD dataset was used to evaluate effectiveness, and the analysis was performed using key metrics: F1-score, Recall, Accuracy, and Precision. The results confirmed the high accuracy, scalability, and stability of GNN-IDS in countering modern threats in IoT systems.

The article [35] creates an optimized Long Short-Term Memory (LSTM) model for detecting anomalies in network traffic using three hyperparameter optimization methods: Particle Swarm Optimization (PSO), JAYA, and Salp Swarm Algorithm (SSA). The NSL-KDD, CICIDS, and BoT-IoT datasets were used for the study. Performance was evaluated using the metrics Accuracy, Precision, Recall, F-score, TPR, FPR, and ROC. A comparative analysis showed that SSA-LSTMIDS outperforms other models on all three datasets, demonstrating the best accuracy and the lowest false positive rate. The article [36] proposes an improved system for detecting intrusions in IoT networks based on Self-Attention Progressive Generative Adversarial Network (SAPGAN). The proposed approach involves collecting IoT data, preprocessing it with local least squares to recover missing values, and selecting optimal features using a modified War Strategy Optimization Algorithm (WSOA). Based on the selected features, traffic is classified as “Anomalous” or “Normal.” Testing on various types of attacks (flood, DDoS, brute force, etc.) showed that SAPGAN outperforms traditional models in accuracy by 18–27% and reduces computation time by 13–26%. In

the article [37], the authors propose a hybrid deep Attention-CNN-LSTM model for detecting intrusions in networks, which combines Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and a self-attention mechanism to extract the most informative features. CNN is used to extract spatial features, while LSTM is used to model temporal dependencies. Testing on the NSL-KDD and Bot-IoT datasets showed an accuracy of 94.8–97.5%, improving MCC and F1-score. The study confirmed the contribution of each component, especially the attention layer. The model provides processing of over 1,200 records/s with a delay of less than 35 ms, making it suitable for high-traffic environments. In the paper [38], the authors propose an IoT-enabled Cyberattack Detection System (IoT-E-CADS) for detecting cyber threats in smart energy metering infrastructure using machine learning methods. The two-level system first applies the Isolation Forest algorithm to detect anomalies and attacks in real time, and then the Decision Tree algorithm to identify cyberattacks and false data injections. The developed device was tested at Quantanics TechServ Pvt. Ltd. (India) with 10 smart meters, where it successfully detected two simulated attacks. The system achieved 95% accuracy, demonstrating its effectiveness for commercial use.

The article [39] presents a new network intrusion detection system (NIDS) based on graph neural networks, specifically the E-GraphSAGE model. The authors justify the use of GNN in this field by the fact that network traffic flows are naturally represented in the form of a graph, which allows for effective consideration of both the structural connections between nodes and the characteristics of the connections themselves. The proposed approach allows for simultaneous consideration of network topology and the characteristics of connections between devices for more accurate detection of intrusions in IoT environments. Based on the results of experiments on four relevant datasets, the model demonstrated superiority over modern methods in terms of key classification metrics. The results confirm the effectiveness of GNN in anomaly detection and highlight the promise of further research in this area.

The article [40] discusses the problem of detecting network intrusions in Edge Computing environments, where traditional methods often prove ineffective due to the complexity and dynamism of network data. Although graph neural networks show potential in this area, most existing solutions are based on simplified graph construction methods that do not reflect the real behavioral relationships between nodes. This leads to overfitting, limited extraction of graph information, and reduced classification accuracy, especially in multi-class scenarios. To address these shortcomings, this paper proposes the BS-GAT method, a graph neural network with attention that takes into account the behavioral similarity between nodes. A new approach to graph construction is proposed, which integrates the weights of behavioral connections into the attention mechanism, allowing for more effective consideration of both structural and contextual information. The results of experiments on modern datasets confirm the effectiveness of the model: in binary classification, all key metrics exceed 99%, and in the multi-class case, the accuracy exceeds 93%, which indicates a significant advantage of the proposed approach over existing solutions. The study [41] addresses the issue of intrusion detection in Internet of Things (IoT) networks, where the high complexity and heterogeneity of the environment pose significant challenges for building effective protection systems. Although graph-based deep learning methods are already showing promise in the field of cybersecurity, most existing approaches form graphs based on physical connections, which does not always adequately reflect the relationships between nodes. This paper proposes a new concept, the Top-K Similarity Graph Framework (TKSGF), in which the graph is constructed based on the similarity of node attributes rather than physical connections. The GraphSAGE model is used to extract node representations, which allows for scalability. Study [42] investigates the problem of automatically detecting malicious outbound traffic from IoT devices. The authors tested several combinations of neural networks (CNN, LSTM, CNN-LSTM) on KDDCup99, NSL-KDD, UNSW-NB15, WSN-DS, and CICIoT2023 datasets. The CNN-LSTM configuration performed best — achieving up to 96% accuracy and 0.94 F1 score with low false positive rates. The authors note that combined spatial and temporal feature extraction ensures stability even in the presence of data imbalance, though highly resource-intensive models may be difficult to deploy in edge scenarios.

The authors of [43] proposed a fuzzy inference-based network traffic analysis system that uses only packet headers for classification (no payload). They defined nine linguistic variables and a set of if-then rules for TCP-SYN flood and other attacks. Results show high classification effectiveness: outperforming traditional binary fuzzy methods while reducing the load on network infrastructure. However, the solution has scalability limitations for complex attacks and requires enhancement of the rule knowledge base. In [44], the issue of network attack detection in cyber-physical systems is addressed using rule-based logical neural networks. The authors implemented an IDS solution that analyzes multivariate time series from sensors directly through a combination of logical rules and neural elements. The model is adapted to heterogeneous cyber-physical environments and showed higher classification accuracy compared to traditional statistical and rule-based approaches. In [45], a method for anomaly detection in IoT device traffic is proposed based on a modified Z-index, which does not require model training or labeled data. The approach builds a profile of normal device behavior, uses median and MAD for noise and outlier resistance, and applies exponential smoothing and cumulative deviation indices to detect both short- and long-term anomalies. The practical implementation with generated logs (4804 devices) showed up to 91% accuracy for interval-based anomalies, with an average F1 score ≈ 0.86. The method is attack-type independent, operates in real-time, and is suitable for resource-limited devices. A weak point remains its sensitivity to recurring patterns, which are difficult to detect without additional contextual analysis.

3. Formalization of IoT network structure and attack vectors in the form of a graph

Present a system for detecting multi-vector attacks on an IoT network in the form of a graph that formalizes its topology, functional relationships, and the dynamics of interactions between devices. This approach will allow for the effective use of graph attention networks (GAT) for analyzing traffic structure, detecting anomalies, and modeling potential attack vectors. Figure 1 shows a general scheme of the mesh IoT network and attacks on the network.

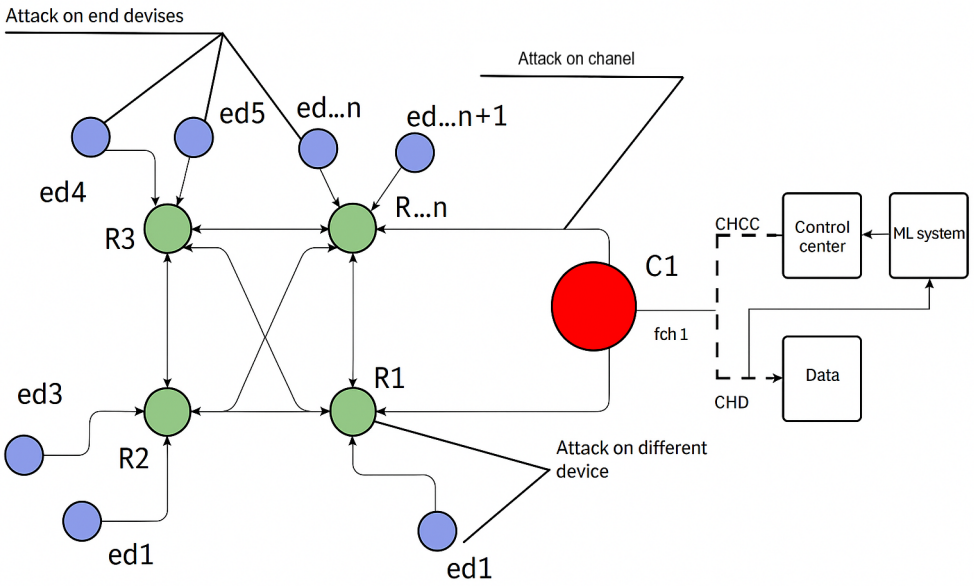

Figure 1: Presentation of the general scheme of the mesh IoT network and attacks on the network.

Based on the presented mesh topology, the IoT network is described as a directed weighted graph:

$$G=(V,E,A,X,W) \quad (1)$$

where each component performs its function in the graph structure. The set of nodes is denoted as *V*, the set of edges as *E*.

For analytical modeling, the following additional components are added:

- $X \in R^{|V| \times d}$ – a matrix of node features, where is the number of descriptors or parameters describing each node
- $A \in \{0,1\}^{|V| \times |V|}$ – a binary adjacency matrix indicating the presence or absence of a connection between nodes
- $W: E \to R^{+}$ – a weight function on the edges that specifies the strength or priority of the connection between nodes

The set of nodes *V* in the graph model of the IoT network includes three categories of elements: end devices V_{ED}, routers V_R and critical network nodes V_R. The V_{ED} group includes sensors, actuators and other devices that generate or consume data – they are the most numerous and vulnerable to attacks. V_R nodes are responsible for routing traffic between end devices and higher levels of the network, forming the main framework of the mesh structure. V_C nodes, in particular *C1*, act as data concentration points or gateways connecting the local network with analysis and management systems. This structure allows you to model functional relationships and risk areas within the framework of detecting multi-vector attacks.

The nodes are divided into three functional groups:

- V_{ED} – end devices $ed_1, ed_2, \ldots, ed_n$
- V_R – routers $R_1, R_2, \ldots, R_n$
- V_C – controllers (control units / gateways) C1

Let's write the total set of nodes:

$$V = V_{ED} \cup V_R \cup V_C \quad (2)$$

The set of edges *E* represents all possible connections between network nodes and reflects both the physical and logical topology of interactions. Each edge $e_{ij}=(\upsilon_i, \upsilon_j) \in E$ reflects the transfer of information or control commands from node υ_i to node υ_j To describe different directions of interaction in the IoT system, the set of edges is divided into two disjoint subsets - internal and external connections. E_{int} – internal connections between nodes within the IoT segment E_{ext} – external logical connections from V_C controllers to external systems, such as a control center or ML module. Physical edges within E_{int} can be implemented via wireless technologies such as ZigBee, BLE, LoRa or other low-power protocols. E_{ext} edges, on the other hand, operate over IP-oriented protocols (e.g., MQTT or HTTPS) and connect controllers to cloud infrastructure or higher-level systems. This separation allows us to separate local (network) interaction channels from channels leading to external objects, including interfaces to cloud systems, gateways, or control controllers. Formally, this is specified as follows:

$$E = E_{\text{int}} \cup E_{\text{ext}}, E_{\text{int}} \cap E_{\text{ext}} = R \quad (3)$$

Thus, the set *E* includes both local routes that provide telemetry and command transmission, and global channels through which integration with the outside world is implemented.

The interaction between the nodes of the set *V* is implemented through directed edges of the set *E*, which form routes for transmitting data, commands, and control logic within the network and outside. The end devices V_{ED} generate telemetry, which is transmitted to the routers V_R through the edges:

$$e_{ij} \in E_{\text{int}} \subseteq V_{ED} \times V_R \quad (4)$$

To display the logical connections of the IoT network with external information systems, a set of logical (non-physical) nodes is introduced:

$$V_{ext} = \{ \upsilon_{ext}^{1}, \upsilon_{ext}^{2}, \ldots, \upsilon_{ext}^{n} \} \quad (5)$$

where υ_{ext}^{n} are nodes such as a ML engine for data processing, a security event control center (CHCC), a cloud storage, or a central telemetry collection server. They are not part of the physical topology of the IoT segment, but act as endpoints for logical communication channels. Thus, external edges are defined as:

$$E_{\text{int}} \subseteq V_C V_{\text{ext}} \quad (6)$$

That implement the transition from IoT protocols to IP-oriented technologies. In the reverse direction, control commands can flow through V_C and V_R to the actuators in V_{ED}. Thus, the edges in the graph not only reflect physical or logical connections, but also determine the permissible paths for data circulation between network subsystems, taking into account the limitations of protocols, topology and security policies (Figure 2).

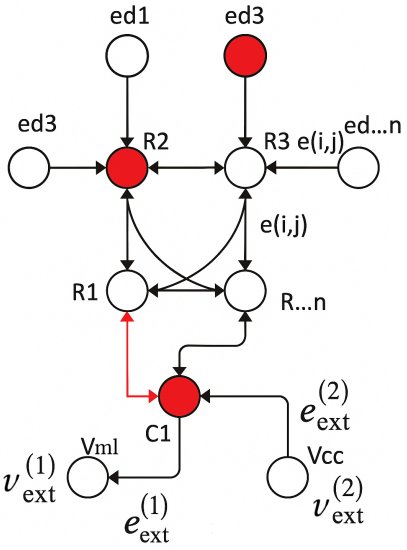

Figure 2: Graph representation of a mesh IoT network.

Based on the formalized model of the IoT network, a directed graph was constructed that reflects both the functional structure of the system and potential attack vectors.

The graph defines three main categories of nodes: end devices, routers, a central controller, as well as external components — a machine learning (ML) module and a control center (Control Center). The connections between nodes model data transmission channels, routing, and control commands.

The nodes under attack are marked in red in the graph:

- *ed2* — a compromised end device from which false data can be injected or an attempt to spread malicious traffic is made
- *R2* — a router that is the target of a DoS attack or route manipulation (redirecting traffic through itself for further control)
- *C1* — a network controller potentially affected by a policy change or infiltration

In addition to the nodes, the edge between nodes *R1* and *C1* is also highlighted in red, which symbolizes the attack vector on the communication channel. Such an attack can be implemented as traffic interception, injection of forged packets, or delay/blocking of messages transmitted from the router to the controller.

Within the graph model of the IoT network, each attack vector is considered as a transformation of the basic structure of the graph G that describes the current state of the system. In this graph, the node set V includes all devices in the network, the edge set E represents the directed links between them, the matrix $X \in R^{|V| \times F}$ contains the numerical attributes of each node, the adjacency matrix $A \in R^{|V| \times |V|}$ reflects the presence of links, and the vector $W \in R^{F}$ (specifies the weights or importance of the edges (for example, bandwidth or channel reliability). Each attack vector α_k is a targeted action that modifies one or more of these sets, leading to a new state of the network G, potentially unstable or vulnerable.

In IoT systems, attacks are almost never limited to a single vector of influence. Due to the complexity of the topology, the branched network structure, the presence of diverse device types, and the multi-level control model, security breaches often have a combined nature. Attackers employ multiple strategies – either simultaneously or sequentially – by targeting end devices, routers, communication channels, and control nodes. This defines the category of multi-vector attacks.

A multi-vector attack involves several directions of influence that reinforce each other. For instance, compromising a sensor may enable influence over a router, which in turn may provide access to the gateway. Such chained effects are extremely difficult to detect when events are analyzed in isolation. Therefore, an integrated model is required that can formalize both individual vectors of influence and their combinations. With in the proposed model, three fundamental types of attack vectors are distinguished:

$$\alpha_{multi} = \alpha_{ED} + \alpha_{ch} + \alpha_{ctrl} \tag{7}$$

Where α_{ED} denotes the compromise of an end device, α_{ch} represents an attack on the data transmission channel, and α_{ctrl} refers to interference with the operation of a controller or gateway. In this expression, each component corresponds to a specific functional domain of the IoT network. Collectively, they model a chain attack, which may begin, for example, with the compromise of an end device and culminate in the takeover of the system's control logic.

A matrix representation was used to analyze impact scenarios. This approach made it possible to formalize the relationships between all network components and assess the intensity and direction of impact. It also made it possible to identify critical escalation paths.

$$M_a = \left(m_j\right), m_{i,j} \in \Omega_{ED,ch,ctl}\, i, j = 1 \ldots n \tag{8}$$

where M_α is the mutual influence matrix within the IoT system.

Each element of this matrix, m_(i,j), reflects the presence and type of influence between component F_i and F_j. If influence exists, it is defined by a type from a predefined set that includes the three main categories of attack vectors. Thus, the matrix not only captures the existence of interactions but also classifies them according to the source of the attack.

$$\Omega = \Omega_{ed}, \Omega_{ch}, \Omega_{ctrl} \tag{9}$$

The Ω set represents three main categories of attack vectors within an IoT network. Ω_{ed} refers to compromises targeting end devices (sensor data tampering, data injection, malware installation). Ω_{ch} refers to attacks on communication channels (packet interception, traffic redirection). Ω_{ch} covers manipulations targeting control blocks or gateways. Each subset reflects a separate level of impact. This allows for the formalization of escalation scenarios at different levels of the IoT system.

Each subset covers typical attack operations at the corresponding level – from data interception to manipulation of control logic. To provide a more detailed view of the components, a block structure of the IoT system was introduced.

$$M_{IT} = \begin{cases} F_0 \mid F_0 = \bigcup_{i=1}^{N_{IT}} F_{0,i} \\ F_1 \mid F_1 = \bigcup_{i=1}^{N_{IT}} F_{1,i} \\ \cdots \\ F_{N_{IT}} \mid F_{N_{IT}} = \bigcup_{i=1}^{N_{IT}} N_{IT} \\ F_{N_{IT}} \mid F_{N_{IT}} = \bigcup_{i=1}^{N_{IT}} N_{IT,i} \\ A_{IT} \mid A_{IT} = \bigcup_{i=1}^{N_{IT}} A_{IT,i} \end{cases} \quad (10)$$

The matrix M_{IT} describes the modular structure of the IoT system, where each logical subsystem F_i consists of a set of subcomponents $F_{i,k}$, where $i=0..N_{\text{IT}}$. The number of such subsystems is defined by the parameter N_{IT}, which represents the total number of functional blocks in the network. In turn, $A_{\text{IT},\text{i}}$ denotes the set of compromised areas, formed as a union of all locally affected components that belong to the corresponding F_i. The resulting impact matrix can be expressed as:

$$M_r = \begin{pmatrix} m_{r,1,1} & \cdots & m_{r,1,N_{\text{VP}}} \\ \vdots & \ddots & \vdots \\ m_{r,1,N_{\text{IT}},1} & \cdots & m_{r,N_{\text{IT}},N_{\text{VP}}} \end{pmatrix} \quad (11)$$

where $m_{r,i,j}$ is a numerical or logical estimate of the impact from the i-th element on the j-th object, N_{IT} is the number of logical subsystems, and N_{VP} is the number of vulnerable objects in the system.

This formalization reflects the structure of risks and allows building an automated analysis system that identifies nodes with the highest number of incoming and outgoing connections. It also assesses critical attack propagation paths and supports scenario testing for protection mechanisms. Representing a multi-vector attack as a graph allows us to move from an intuitive description to a formalized model. This model covers the structural and behavioral characteristics of an IoT network. Such formalization is a basic prerequisite for creating an effective mechanism for detecting and responding to complex compromise scenarios.

4. Telemetry modeling and feature extraction

To link the formal topology with the actual dynamics of events in the network, we introduce the set T, which represents telemetry data describing the current parameter values of devices in the network over a given time period. This telemetry serves as the primary source of information for constructing feature vectors x_i, evaluating node states, and identifying behavioral anomalies. We define the set T as follows:

$$T = \left\{ T_i \middle| \upsilon_i \in V \right\} \quad (12)$$

where each T_i is a time series of measurements for node υ_i and has the following form:

$$T_i = \left\{ t_i^1, t_i^2, \ldots, t_i^\tau \right\}, t_i^k \in R^d \quad (13)$$

where τ is the number of time points, and d is the number of parameters (features) recorded by the device at each time step k; for example, RSSI, CPU load, latency, battery level, etc.

These time series reflect the behavior of devices in the network and form the foundation for constructing node feature vectors x_i, analyzing network state, and detecting anomalies using statistical methods and the model. Since the set T contains dynamic elements, we do not use the entire series but only selected fragments within a fixed time interval $[t_{a,} t_b]$, introducing a temporally localized subset of telemetry:

$$T^{[t_a,t_b]} = \bigcup_{v_i \in V} T^{[t_a,t_b]}, T_i^{[t_a,t_b]} = \{t_i^k \in T_i \mid t_{a,} \leq k \leq t_b\} \tag{14}$$

The description of the set of telemetry vectors $T^{[t_{a,}t_b]}$ provides information about the behavior of each node in the IoT network within the specified time interval. It allows for the assessment of both local and global changes in network state, detection of deviations in device parameters, and construction of a structured model $G_{[t_{a,}t_b]}$ used for anomaly detection. The feature vector x_i summarizes the telemetry data of node $v_i \in V$ over the interval $[t_{a,} t_b]$, characterizing its functional state based on metrics t_i^k. It serves as input to the GAT model used for analyzing and detecting anomalies.

$$x_i = \begin{bmatrix} RSSI \\ LATENSY \\ POWER\ LIMIT \\ CPU \\ \dots \end{bmatrix} \in R^M \tag{15}$$

The content of the feature vector x_i depends on the type of node and the set of available telemetry metrics. It may include parameters such as RSSI, latency, CPU usage, packet loss, power consumption, battery level, queue size, or other device state indicators. If needed, the feature set may also include latent characteristics obtained using an autoencoder or other dimensionality reduction techniques. The features are formed based on aggregated telemetry values over the time interval $[t_{a,} t_b]$, for example, by calculating the mean, variance, maximum, or trend.

The structure of the graph $G_{[t_{a,}t_b]}$, which represents the relationships between nodes of the IoT system within the specified time interval, is constructed by forming the edge set $E \subseteq V \times V$, where each edge $e_{i,j} = (v_i, v_j)$ denotes the presence of a data transmission channel or logical interaction between devices v_i and v_j. Using the adjacency matrix A_{ij}, we represent the structure of the graph required for computations in the GAT model, allowing us to define the connections between nodes and control the flow of information within the neural network.

$$A_{ij} = \begin{cases} 1, \text{ if there exists an edge } e_{i,j} \in E \text{ between nodes } v_i \text{ and } v_j \\ 0, \text{otherwise} \end{cases} \tag{16}$$

$A_{ij} = 1$ indicates that node v_i transmits information and v_j receives it during neural network computations. To provide a more accurate description of node interactions, a weight matrix

$W \in R^{n \times n}$ is introduced, where the weight W_{ij} represents the degree of influence from node v_i to v_j:

$$W_{ij} = \begin{cases} \text{link metric if } (v_i, v_j) \in E) \\ 0, \text{ otherwise} \end{cases} \tag{17}$$

The weight value is based on the characteristics of the data transmission channels and the behavioral parameters of the node.

For the generalized representation of node υ_i, obtained after passing through n layers of the GAT, we define the vector $h_i x_{ij}$. It integrates the local behavior of the node (telemetry, features) and the structured context (interactions with neighboring nodes and their features).

$$h_i^{(n+1)} = \sigma\left(\sum_{j \in N(i)} a_{ij}^{(n)} W^{(n)} h_j^{(n)} \right) \tag{18}$$

At each GAT layer, the new representation of node υ_i at level *n+1* is computed as a nonlinear representation of the weighted sum of its neighbors' features. Aggregation weights are determined using the attention mechanism, which dynamically defines the importance of each node $\upsilon_j \in N(i)$ in the context of υ_i. The evaluation mechanism is activated by a nonlinear function σ (ReLU або ELU), and $a_{ij}^{(n)}$ defines the attention coefficient of node υ_j with respect to υ_i.

The threshold value θ for anomaly decision-making can be either fixed or adaptive. In certain scenarios, it is appropriate to use statistical or quantile-based approaches that take into account the current distribution of scoring values in the network. This enables better adaptation to changing system conditions. The interpreter makes the anomaly decision for the representation of node υ_i according to the algorithm described below.

Algorithm 1 Workflow inference-based anomaly detection using GAT-interpreter

Require:
Graph representation
$G = (V, E, X), GAT - generated\ node\ embeddings\ h_i \in R^d$
Ensure:
Anomaly detection result $y_i \in \{0,1\}$ for each node $\upsilon_i \in V$
1: Begin
2: Input data preprocessing
3:Extract telemetry $T^{[t_a, t_b]}$ and compute embedding h_i via trained GAT model
4: Score calculation
5: For each node $\upsilon_i \in V$, compute anomaly score:
6: $s_i = h_i, C_i$
7: Where C_i – local or global context (e.g., neighborhood deviation)
8: **Threshold estimation**
9: Determine adaptive or static threshold θ_i using statistical rules:
10: e.g. $\theta_i = \mu_s + \lambda \cdot \sigma_s$
11: **Decision rule**
12: if $s_i > \theta_i$ classify node υ_i as anomalous:
13: $y_i = 1$ (anomaly)
14: Else $y_i = 0$ (normaly)
15: **Output classification vector** $\vec{y} = [y_1, y_1, \ldots, y_{1n}]$
16: **End**

The decision-making algorithm implements the classification rule for the behavior of network node υ_i, based on the feature vector h_i generated after passing through the GAT block. A scoring function is used to produce a numerical value s_i that reflects the degree of deviation in the node's behavior in the context of its neighbors. The decision is made by comparing s_i with an adaptive or fixed threshold θ_i, calculated using statistical rules as $\theta_i = \mu_s + \lambda \cdot \sigma_s$. As a result, a classification label $y_i \in \{0,1\}$ is assigned, where 1 indicates an anomaly and 0 corresponds to a normal node state.

The GAT model is refined by adjusting its parameters based on the observed behavior of nodes in the IoT network. Training is performed iteratively as follows: at each step, the network receives input feature vectors x_i and generates corresponding representations h_i, which are then analyzed by the interpretation module. The obtained results are compared against expected labels or statistical references.

Algorithm 2: Workflow training module for GAT-based anomaly detection

Require:
Graph structure $G = (V, E, X)$, telemetry dataset $T^{[t_a, t_b]}$
initialized GAT model
Ensure:
Trained GAT model with optimized parameters W, a_{ij}
1: **Begin**
2: Preprocess telemetry data $T^{[t_a, t_b]}$ to generate node features $X = \{x_1, x_2, \ldots, x_n\}$
3: Construct graph $G = (V, E, X)$
4: Initialize GAT model parameters W and attention weights a_{ij}
5: for each $e \in \{1, \ldots, E\}$ do
6: for each node $\upsilon_i \in$ do
7: Compute node embedding h_i using GAT
8: Infer anomaly score or label y_i
9: end for
10: Compute loss *L(h,y)* (binary cross-entropy)
11: Back propagate gradients and update W, a_{ij}
12: end for
13: Output: Trained GAT model
14: **End**

To optimize the model parameters, a binary cross-entropy loss function *L(h,y)* is used, which allows for an accurate assessment of the discrepancy between the predicted and expected anomaly values at each training step.

Figure 3 presents the proposed architecture of the GAT-based anomaly detection model for IoT networks. It is built upon the coordinated interaction of three core components: the converter, the interpreter, and the training module. Input data in the form of telemetry $T^{[t_a, t_b]}$ enters the system through a data collection pipeline. The converter generates a feature vector x_i that reflects the functional state of the node within the specified time interval. The interpreter then constructs the

representation h_i using the attention mechanism, which takes into account both the node's local features and the structure of its neighborhood in the graph. This representation is passed to the training module, which estimates the probability of anomalous behavior and optimizes the model using the loss function.

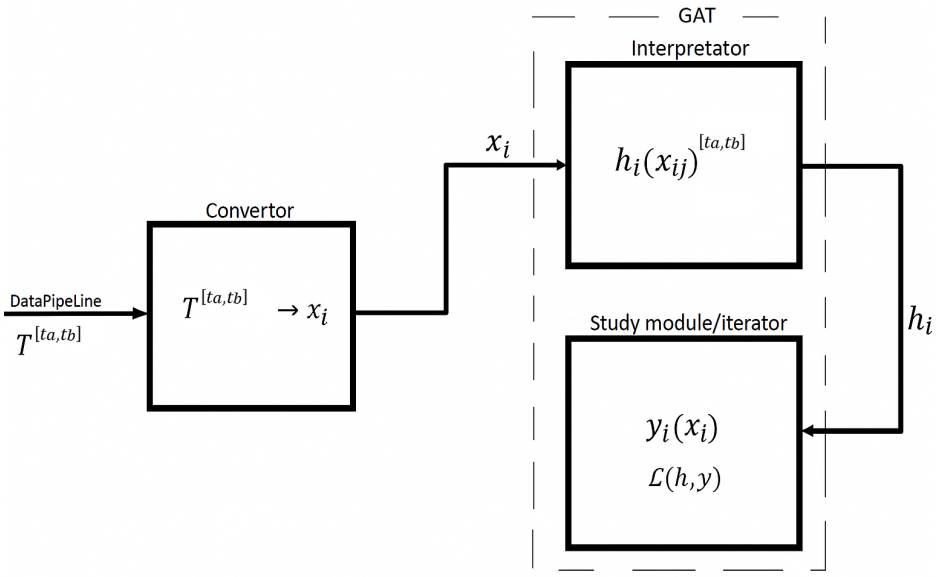

Figure 3: Architecture of the GAT model.

The simulation results demonstrate that combining the structural representation of the IoT network with telemetry interpretation via a GAT model enables the construction of a flexible mechanism for detecting complex, multi-vector attacks.

5. Evaluation of the effectiveness of the method

In the course of the experimental modeling, both publicly available and synthetically generated IoT datasets were utilized. Public datasets such as UNSW BoT-IoT were used at the initial stage for baseline evaluation, while a custom-generated dataset was employed to simulate ZigBee-based telemetry from low-power IoT devices. The synthetic data included frame-level attributes (e.g., RSSI, LQI, sequence number) and was created using a controlled Python-based simulation pipeline. The datasets cover a wide range of parameters including wireless signal characteristics, device-level metrics, behavioral anomalies, and logical communication patterns, and include labeled instances of various attacks such as DoS, spoofing, scanning, privilege escalation, and data poisoning.

For the purposes of this study, each dataset was preprocessed and transformed into a graph-based format, where nodes represent IoT devices and edges represent communication channels (physical or logical). Node attributes include both telemetry data and signature-based features, which serve as input to the graph neural network.

Figure 4 illustrates the sequential transformation of raw IoT telemetry into feature vectors suitable for processing by graph neural networks. The first block contains raw dataset values such as CPU, RAM, network traffic, protocol, and class labels.

The second block shows the normalized form of the feature vectors x_i, corresponding to the first level of preprocessing: percentage values of CPU and RAM were converted to numerical form (from 0 to 1), and packet count values were scaled.

The third block illustrates the formation of hidden-space vectors h_i, obtained through a linear transformation of the features x_i (emulating the weighted aggregation mechanism in GAT). Each vector becomes part of the subsequent aggregation process within the device's subgraph.

timestamp	src_addr	dst_addr	rame_type	seq_no	RSSI	LQI	PAN_ID	class
2025-02-2	0xD4	0xC3	ACK	158	-74	94	0xABCD	Normal
2025-02-2	0xA1	0xB2	Beacon	140	-51	85	0xABCD	Attack
2025-02-2	0xD4	0xA1	Beacon	189	-46	117	0xABCD	Normal
2025-02-2	0xC3	0xA1	ACK	43	-56	122	0xABCD	Normal
2025-02-2	0xC3	0xFF	Data	125	-65	105	0xABCD	Normal
2025-02-2	0xA1	0xB2	ACK	97	-51	85	0xABCD	Normal
2025-02-2	0xB2	0xC3	Beacon	18	-63	88	0xABCD	Normal
2025-02-2	0xB2	0xC3	Beacon	157	-74	112	0xABCD	Normal
2025-02-2	0xA1	0xFF	Data	48	-55	123	0xABCD	Attack

timestamp	x_seq_no	x_RSSI	x_LQI	class
2025-02-2	0,62963	0,275	0,28	Normal
2025-02-2	0,555556	0,85	0,1	Attack
2025-02-2	0,757202	0,975	0,74	Normal
2025-02-2	0,156379	0,725	0,84	Normal
2025-02-2	0,493827	0,5	0,5	Normal

timestamp	h1	h2	class
2025-02-21 10:0	0,468722	0,051074	Normal
2025-02-21 10:0	0,071667	0,338889	Attack
2025-02-21 10:0	0,352401	0,62956	Normal
2025-02-21 10:0	0,056784	0,639724	Normal
2025-02-21 10:0	0,32037	0,301235	Normal

Figure 4: Sample transformation of raw data into the unified node feature representation.

This output represents the final result of the model's forward pass on a single data batch, enabling the computation of the loss function and subsequent training.

Figure 5 shows a visualization of the transition from a time series of network traffic to the graph representation of IoT devices.

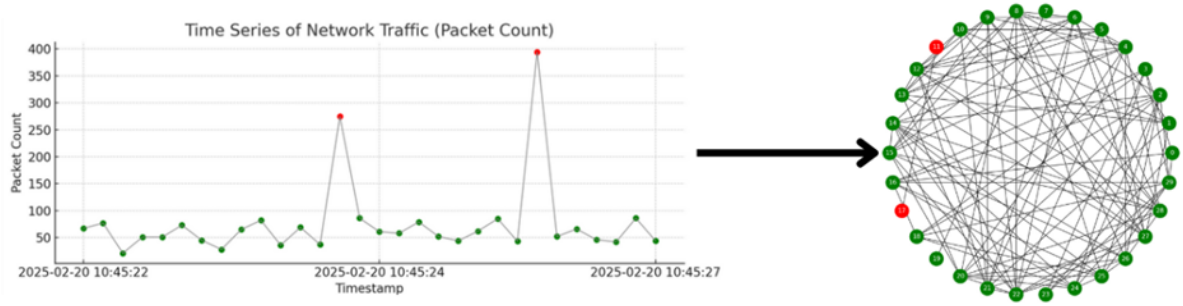

Figure 5: Visualization of the transition from network traffic time series to the graph representation of IoT devices.

On the left, the time series of the feature pkt_count (packet count) for individual devices is shown; red dots indicate anomalous nodes. On the right, a constructed graph is presented, where each vertex corresponds to a device and edges reflect feature similarity in the hidden space. This transformation serves as the basis for further processing by the graph neural network.

Figure 6 illustrates the processing results of IoT devices within the graph neural network across multiple training epochs. Each row originally corresponded to a single network device at a specific point in time. The vectors h_1, h_2, h_3 represent the hidden features of the node, formed based on its local neighborhood in the graph.

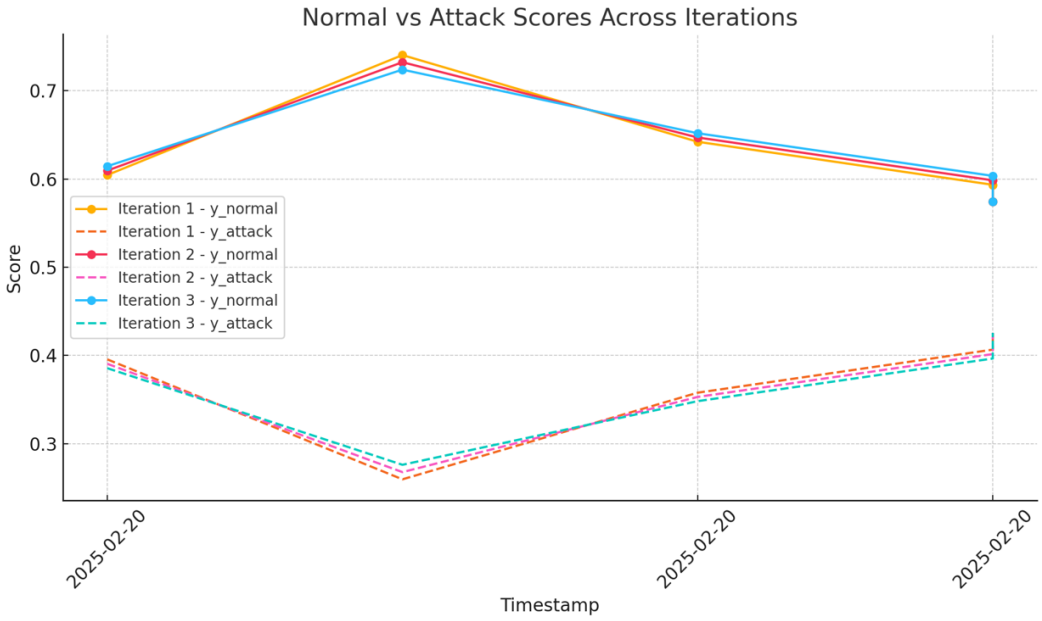

Figure 6: Processing results of IoT devices within the graph neural network across multiple training epochs.

During the experimental modeling, the model's classification performance was also evaluated based on the learned node representations h_i. The model's ability to correctly determine whether a node belongs to the "normal" or "anomalous" class within the graph structure was assessed. Standard classification metrics were used for this purpose: accuracy, recall, precision, F1-score, and ROC-AUC — the latter capturing model stability under varying decision thresholds.

A comparison of four graph-based models was performed, namely GCN, GraphSAGE, GAE, and the proposed GAT. The results of the performance comparison are presented in Table 1 and Figure 7.

Table 1

Comparative evaluation of GNN models on IoT anomaly detection task

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
GCN	0.93	0.9	0.94	0.92	0.95
GraphSAGE	0.927	0.89	0.89	0.91	0.94
GAE	0.918	0.88	0.91	0.89	0.92
GAT	0.925	0.89	0.93	0.91	0.955

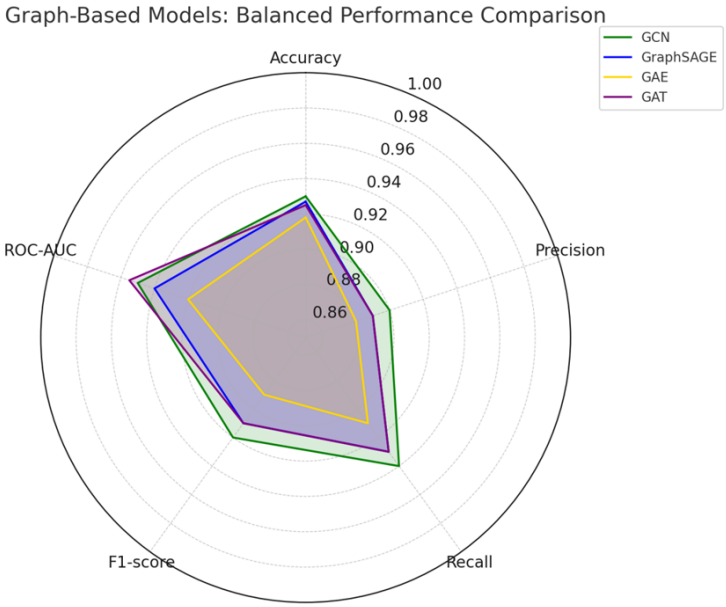

Figure 7: Performance comparison of four graph-based models: GCN, GraphSAGE, GAE, and GAT

The GCN model demonstrates an accuracy of 0.93 and an F1-score of 0.92, indicating its effectiveness in tasks with clearly defined classes. GraphSAGE shows only a slight decrease in accuracy while maintaining stable recall and precision values. GAE, being an autoencoder-based model, underperforms across all metrics but remains suitable for unsupervised learning.

GAT, although ranking second or third across most metrics, achieves the highest ROC-AUC value of 0.955. This highlights its strong ability to distinguish between normal and anomalous nodes under varying threshold conditions. These results suggest that GAT has substantial potential for analyzing long-term behavioral trends in network structures, despite some decline in local classification accuracy.

6. Conclusions

The proposed approach to detecting multi-vector attacks in IoT networks is based on the use of graph attention networks. This allows for the structural properties of the network and the behavioral characteristics of devices to be taken into account. Building a graph model using telemetry and structural connections, along with applying attention mechanisms for weighted aggregation of information, ensures high sensitivity to complex multi-vector attacks. Experimental results have shown that the GAT model is capable of effectively detecting attacks. A comparative analysis with other GNN-based models confirmed the competitiveness of GAT in anomaly detection tasks. In future work, the main focus will be on improving the accuracy of the model and the value of the F1-score.

Declaration on Generative AI

AI tools were used solely as translation and proofreading aids. All content was originally authored by the submitting party.

References

[1] C. Paul, A. Ganesh, C. Sunitha, An overview of IoT based smart homes, in: 2018 2nd International Conference on Inventive Systems and Control (ICISC), IEEE, 2018. doi:10.1109/icisc.2018.8398858.

[2] S. S. I. Samuel, A review of connectivity challenges in IoT-smart home, in: 2016 3rd MEC International Conference on Big Data and Smart City (ICBDSC), IEEE, 2016. doi:10.1109/icbdsc.2016.7460395.
[3] M. S. Farooq, S. Riaz, A. Abid, T. Umer, Y. B. Zikria, Role of IoT Technology in Agriculture: A Systematic Literature Review, Electronics 9.2 (2020) 319. doi:10.3390/electronics9020319.
[4] J. Xu, B. Gu, G. Tian, Review of agricultural IoT technology, Artif. Intell. Agric. 6 (2022) 10–22. doi:10.1016/j.aiia.2022.01.001.
[5] Y. Song, F. R. Yu, L. Zhou, X. Yang, Z. He, Applications of the Internet of Things (IoT) in Smart Logistics: A Comprehensive Survey, IEEE Things J. (2020) 1. doi:10.1109/jiot.2020.3034385.
[6] C. Caballero-Gil, J. Molina-Gil, P. Caballero-Gil, A. Quesada-Arencibia, IoT Application in the Supply Chain Logistics, in: Computer Aided Systems Theory - EUROCAST 2013, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 55–62. doi:10.1007/978-3-642-53862-9_8.
[7] S. Selvaraj, S. Sundaravaradhan, Challenges and opportunities in IoT healthcare systems: a systematic review, SN Appl. Sci. 2.1 (2019). doi:10.1007/s42452-019-1925-y.
[8] B. Farahani, F. Firouzi, K. Chakrabarty, Healthcare IoT, in: Intelligent Internet of Things, Springer International Publishing, Cham, 2020, pp. 515–545. doi:10.1007/978-3-030-30367-9_11.
[9] Z. Lv, B. Hu, H. Lv, Infrastructure Monitoring and Operation for Smart Cities Based on IoT System, IEEE Trans. Ind. Inform. 16.3 (2020) 1957–1962. doi:10.1109/tii.2019.2913535.
[10] R. Sharma, Evolution in Smart City Infrastructure with IOT Potential Applications, in: Intelligent Systems Reference Library, Springer International Publishing, Cham, 2018, pp. 153–183. doi:10.1007/978-3-030-04203-5_8.
[11] Koroliuk, R., Nykytyuk, V., Tymoshchuk, V., Soyka, V., Tymoshchuk, D. Automated monitoring of bee colony movement in the hive during winter season. CEUR Workshop Proceedings, 2024, 3842, pp. 147-156
[12] T. L. Narayana, C. Venkatesh, A. Kiran, C. B. J, A. Kumar, S. B. Khan, A. Almusharraf, T. Quasim, Advances in real time smart monitoring of environmental parameters using IoT and sensors, Heliyon (2024) e28195. doi:10.1016/j.heliyon.2024.e28195.
[13] Didych, I., Mykytyshyn, A., Stanko, A., Mytnyk, M. Application of machine learning methods to the prediction of NO2 concentration in the air environment. CEUR Workshop Proceedings, 2024, 3896, pp. 569–577
[14] D. Tymoshchuk, O. Yasniy, P. Maruschak, V. Iasnii, I. Didych, Loading frequency classification in shape memory alloys: A machine learning approach, Computers 13.12 (2024) 339. doi:10.3390/computers13120339.
[15] Stukhliak, P., Totosko, O., Stukhlyak, D., Vynokurova, O., & Lytvynenko, I. (2024). Use of neural networks for modelling the mechanical characteristics of epoxy composites treated with electric spark water hammer. CEUR Workshop Proceedings, 3896, 405–418.
[16] O. Yasniy, D. Tymoshchuk, I. Didych, V. Iasnii, I. Pasternak, Modelling the properties of shape memory alloys using machine learning methods, Procedia Struct. Integr. 68 (2025) 132–138. doi:10.1016/j.prostr.2025.06.033.
[17] Stukhliak, P., Martsenyuk, V., Totosko, O., Stukhlyak, D., & Didych, I. (2024). The use of neural networks for modeling the thermophysical characteristics of epoxy composites treated with electric spark water hammer. CEUR Workshop Proceedings, 3742, 13–24.
[18] O. Yasniy, P. Maruschak, A. Mykytyshyn, I. Didych, D. Tymoshchuk, Artificial intelligence as applied to classifying epoxy composites for aircraft, Aviation 29.1 (2025) 22–29. doi:10.3846/aviation.2025.23149.
[19] O. Chukur, N. Pasyechko, A. Bob, A. Sverstiuk, Prediction of climacteric syndrome development in perimenopausal women with hypothyroidism, Menopausal Rev. (2022). doi:10.5114/pm.2022.123522.
[20] S. O. Nykytyuk, A. S. Sverstiuk, D. S. Pyvovarchuk, S. I. Klymnyuk, A multifactorial model for predicting severe course and organ and systems damage in Lyme borreliosis in children, Mod. Pediatr. Ukr. No. 2(130) (2023) 6–16. doi:10.15574/sp.2023.130.6.

[21] S. Nykytyuk, A. Sverstiuk, S. Klymnyuk, D. Pyvovarchuk, Y. Palaniza, Approach to prediction and receiver operating characteristic analysis of a regression model for assessing the severity of the course Lyme borreliosis in children, Rheumatology 61.5 (2023) 345–352. doi:10.5114/reum/173115.
[22] N. Nazareth, Y. Y. Ramana Reddy, Financial applications of machine learning: a literature review, Expert Syst. With Appl. (2023) 119640. doi:10.1016/j.eswa.2023.119640.
[23] Wei-Yang Lin, Ya-Han Hu, Chih-Fong Tsai, Machine Learning in Financial Crisis Prediction: A Survey, IEEE Trans. Syst., Man, Cybern.,C (Appl. Rev.) 42.4 (2012) 421–436. doi:10.1109/tsmcc.2011.2170420.
[24] A. Mashrur, W. Luo, N. A. Zaidi, A. Robles-Kelly, Machine Learning for Financial Risk Management: A Survey, IEEE Access 8 (2020) 203203–203223. doi:10.1109/access.2020.3036322.
[25] D. Tymoshchuk, I. Didych, P. Maruschak, O. Yasniy, A. Mykytyshyn, M. Mytnyk, Machine Learning Approaches for Classification of Composite Materials, Modelling 6.4 (2025) 118. doi:10.3390/modelling6040118.
[26] Yasniy, O., Tymoshchuk, D., Didych, I., Zagorodna, N., & Malyshevska, O. Modelling of automotive steel fatigue lifetime by machine learning method. CEUR Workshop Proceedings, 2024, 3896, pp. 165–172
[27] C. Park, C. C. Took, J.-K. Seong, Machine learning in biomedical engineering, Biomed. Eng. Lett. 8.1 (2018) 1–3. doi:10.1007/s13534-018-0058-3.
[28] Tymoshchuk, D., Yasniy, O., Mytnyk, M., Zagorodna, N., Tymoshchuk, V. Detection and classification of DDoS flooding attacks by machine learning method. CEUR Workshop Proceedings, 2024, 3842, pp. 184 – 195
[29] A. Handa, A. Sharma, S. K. Shukla, Machine learning in cybersecurity: A review, WIREs Data Min. Knowl. Discov. 9.4 (2019). doi:10.1002/widm.1306.
[30] Lypa, B., Horyn, I., Zagorodna, N., Tymoshchuk, D., Lechachenko T. Comparison of feature extraction tools for network traffic data. CEUR Workshop Proceedings, 2024, 3896, pp. 1-11.
[31] T. Ngo, J. Yin, Y.-F. Ge, H. Wang, Optimizing IoT Intrusion Detection—A Graph Neural Network Approach with Attribute-Based Graph Construction, Information 16.6 (2025) 499. doi:10.3390/info16060499.
[32] S. Ben Atitallah, M. Driss, W. Boulila, A. Koubaa. Enhancing Internet of Things Security through Self-Supervised Graph Neural Networks. In: Computer Systems and Information Technologies, 2 (2024) 14–20. doi:10.48550/arXiv.2412.13240.
[33] J. L. Guerra, C. Catania, E. Veas, Datasets are not Enough: Challenges in Labeling Network Traffic, Comput. & Secur. (2022) 102810. doi:10.1016/j.cose.2022.102810.
[34] A. S. Ahanger, S. M. Khan, F. Masoodi, A. O. Salau, Advanced intrusion detection in internet of things using graph attention networks, Sci. Rep. 15.1 (2025). doi:10.1038/s41598-025-94624-8.
[35] N. Dash, S. Chakravarty, A. K. Rath, N. C. Giri, K. M. AboRas, N. Gowtham, An optimized LSTM-based deep learning model for anomaly network intrusion detection, Sci. Rep. 15.1 (2025). doi:10.1038/s41598-025-85248-z.
[36] V. Kantharaju, H. Suresh, M. Niranjanamurthy, S. I. Ansarullah, F. Amin, A. Alabrah, Machine learning based intrusion detection framework for detecting security attacks in internet of things, Sci. Rep. 14.1 (2024). doi:10.1038/s41598-024-81535-3.
[37] A. M. Alashjaee, Deep learning for network security: an Attention-CNN-LSTM model for accurate intrusion detection, Sci. Rep. 15.1 (2025). doi:10.1038/s41598-025-07706-y.
[38] K. Naveeda, S. M. H. S. S. Fathima, Real-time implementation of IoT-enabled cyberattack detection system in advanced metering infrastructure using machine learning technique, Electr. Eng. (2024). doi:10.1007/s00202-024-02552-z.
[39] W.W. Lo, S. Layeghy, M. Sarhan, M. Gallagher, M. Portmann. E-GraphSAGE: A Graph Neural Network based Intrusion Detection System for IoT. In: Proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS 2022), Budapest, Hungary, 2022, pp. 1–9. doi:10.1109/NOMS54207.2022.9789878

[40] Y. Wang, Z. Han, Y. Du, J. Li, X. He, BS-GAT: a network intrusion detection system based on graph neural network for edge computing, Cybersecurity 8.1 (2025). doi:10.1186/s42400-024-00296-8.

[41] T. Ngo, J. Yin, Y.-F. Ge, H. Wang, Optimizing IoT Intrusion Detection—A Graph Neural Network Approach with Attribute-Based Graph Construction, Information 16.6 (2025) 499. doi:10.3390/info16060499.

[42] Klots Y., Petliak N., Martsenko S., Tymoshchuk V., Bondarenko I. Machine Learning system for detecting malicious traffic generated by IoT devices. CEUR Workshop Proceedings, 2024, 3742, pp. 97 – 110

[43] Petliak N., Klots Y., Titova V., Salem A.-B.M. Attack detection system based on network traffic analysis by means of fuzzy inference. CEUR Workshop Proceedings, 2024, 3899, pp. 201 – 213

[44] Titova, V., Klots, Y., Cheshun, V., Petliak, N., Salem, A.-B.M. Detection of network attacks in cyber-physical systems using a rule-based logical neural network. CEUR Workshop Proceedings, 2024, 3736, pp. 255–268

[45] Stetsiuk, M., Anikin, V., Pyrch, O., Kozelskiy, O., Salem, A.-B.M. Method of detecting anomalies in IOT device traffic based on statistical analysis using the modified z score. CEUR Workshop Proceedings, 2025, 3963, pp. 284–298