electronics

MDPI

Systematic Review

The Role of Graph Neural Networks, Transformers, and Reinforcement Learning in Network Threat Detection: A Systematic Literature Review

Thilina Prasanga Doremure Gamage [1,*], Jairo A. Gutierrez [1] and Sayan K. Ray [2]

[1] Department of Computer and Information Sciences, School of Engineering, Computer, and Mathematical Sciences, Auckland University of Technology, Auckland 1010, New Zealand; jairo.gutierrez@aut.ac.nz
[2] School of Computer Science, Faculty of Innovation & Technology, Taylor's University, Subang Jaya 47500, Malaysia; sayan.ray@taylors.edu.my
* Correspondence: thilina.gamage@autuni.ac.nz

Abstract

Traditional network threat detection based on signatures is becoming increasingly inadequate as network threats and attacks continue to grow in their novelty and sophistication. Such advanced network threats are better handled by anomaly detection based on Machine Learning (ML) models. However, conventional anomaly-based network threat detection with traditional ML and Deep Learning (DL) faces fundamental limitations. Graph Neural Networks (GNNs) and Transformers are recent deep learning models with innovative architectures, capable of addressing these challenges. Reinforcement learning (RL) can facilitate adaptive learning strategies for GNN- and Transformer-based Intrusion Detection Systems (IDS). However, no systematic literature review (SLR) has jointly analyzed and synthesized these three powerful modeling algorithms in network threat detection. To address this gap, this SLR analyzed 36 peer-reviewed studies published between 2017 and 2025, collectively identifying 56 distinct network threats via the proposed threat classification framework by systematically mapping them to Enterprise MITRE ATT&CK tactics and their corresponding Cyber Kill Chain stages. The reviewed literature consists of 23 GNN-based studies implementing 19 GNN model types, 9 Transformer-based studies implementing 13 Transformer architectures, and 4 RL-based studies with 5 different RL algorithms, evaluated across 50 distinct datasets, demonstrating their overall effectiveness in network threat detection.

Keywords: anomaly detection; cybersecurity; graph neural networks; intrusion detection systems; network threat detection; reinforcement learning; transformers

Academic Editors: Ning Yu and Wei Zhong

Received: 15 September 2025
Revised: 18 October 2025
Accepted: 23 October 2025
Published: 24 October 2025

Citation: Doremure Gamage, T.P.; Gutierrez, J.A.; Ray, S.K. The Role of Graph Neural Networks, Transformers, and Reinforcement Learning in Network Threat Detection: A Systematic Literature Review. *Electronics* **2025**, *14*, 4163. https://doi.org/10.3390/electronics14214163

1. Introduction

Despite the recent rise in popularity of cloud services, critical government, financial, and healthcare infrastructures still depend on Local Area Networks (LANs) for storing, processing, and sharing sensitive information. As the number of these networks employed by small- to large-scale organizations continues to grow, they become increasingly attractive targets for a wide array of network attacks. Meanwhile, as networking technologies advance, so does the volume and sophistication of the attacks. From basic reconnaissance and port scanning to zero-day exploits, insider threats and advanced persistent threats (APTs), the variety of malicious threats and attack strategies continues to expand today. Therefore, network threat detection and attack prevention have become important components of

modern cybersecurity strategies, protecting sensitive data, digital assets, and ensuring service continuity. Unlike conventional network attack detection, which heavily relies on recognizing known attack signatures during or after an attack, network threat detection's primary goal is to identify early threat indicators through continuous monitoring and analysis of network-related activities before they escalate into full-scale attacks. These threat indicators include, but are not limited to, suspicious user and device behavior, such as unauthorized scanning, privilege escalations, unusual network patterns, and anomalies.

1.1. Intrusion Detection Systems

Today, Intrusion Detection Systems are considered one of the most effective defense mechanisms in protecting critical network infrastructures. According to their underlying detection mechanism, IDS can be categorized into two main categories: Signature-Based IDS (SIDS) and Anomaly-Based IDS (AIDS). SIDS rely on known attack signatures or rules, whereas AIDS flag deviations from normal network behavior by employing a wide range of machine learning and deep learning algorithms. A third IDS category, called hybrid IDS, combines both signature-based and anomaly-based techniques in its intrusion detection strategies. SIDS' focus on attack detection is reactive as they require signatures of malicious activities that represent actual network attacks, whereas AIDS are fundamentally proactive in their strategy. AIDS' main focus is on network threats and the threat stages of developing attacks, flagging threat-related deviations from normal network behavior, and actively detecting precursors to attacks.

1.1.1. Signature-Based IDS

Traditional Signature-based Intrusion Detection Systems are static systems primarily designed to detect known attack patterns by matching suspicious network activities against a database of known attack signatures, also known as fingerprints. SIDS employ straightforward logic, implemented via mechanisms based on rules such as simple if–else conditions or pattern-matching mechanisms like regular expressions [1]. Snort and Suricata are well known for their performance in detecting known attacks and are known to produce low false-positive rates. However, by the time the attack signatures are identified, the network is already under attack and has passed the threat development stage. Therefore, while effective and often operating in near real-time, rule/signature-based IDS are not proactive threat detectors. They inherently fail to detect zero-day and novel network exploits, such as polymorphic attacks [2,3]. SIDS' inability to address developing attacks at their various threat stages makes them inadequate for more recent, sophisticated multistage network attack strategies, such as APT campaigns. Such sophisticated attack campaigns demand careful analysis of network behavior over a period to identify potential pre-attack anomalies or threats before they manifest into real attacks. Moreover, traditional SIDS require continuous manual updates to their attack signature library to provide coverage for new attack types as they emerge and to refine their rules, as the deletion of outdated or inefficient signatures is mandatory, thus making these databases difficult to maintain [2]. Moreover, the organizations that depend on SIDS are vulnerable to new threats until they receive the signature updates to detect those threats. Furthermore, some studies revealed that the threat actors can easily bypass SIDS by making slight changes to the payload, such as re-encoding, no-op padding, and creating polymorphic variants [4].

1.1.2. Anomaly-Based IDS

As a result of the critical issues associated with SIDS, modern IDS are moving toward more proactive threat detection. This approach involves identifying and flagging early warning signs, suspicious network behaviors, or anomalies that may indicate upcoming network attacks. Therefore, modern IDS now focus more on AIDS qualities, where ML and

DL models can be effectively used to detect network threats. These approaches are categorized into three main types based on their training methods: supervised, unsupervised, and semi-supervised. In supervised training, ML and DL models learn from both normal and malicious data. Support Vector Machines (SVM), Decision Trees, and Random Forests are common supervised ML models, while DL models such as Feedforward Deep Neural Networks, Convolutional Neural Networks (CNNs), and sequential models like Recurrent Neural Networks (RNNs) or Long Short-Term Memory (LSTM) models are trained using supervised and unsupervised techniques. Unsupervised ML and DL models are especially important in AIDS because they excel at identifying data points that significantly deviate from normal network behavior as anomalies or threats. However, these methods often produce more false positives, which is a critical issue that needs to be addressed. Common unsupervised ML models include K-Means clustering, one-class SVM, and Isolation Forests, while autoencoders, variational autoencoders (VAEs), and deep Belief Networks are used as unsupervised DL models. The third type, self-supervised learning, which is a subcategory of unsupervised learning, is increasingly employed in recent research, where models predict masked parts of their input data [5].

1.1.3. Limitations of Existing AIDS

While signature-based IDS inherently fail to identify attacks during their early threat stages and are therefore not suitable for detecting sophisticated, subtle, evolving, or zero-day attacks, the traditional ML- and DL-based anomaly detection models proposed as solutions also encounter significant limitations and challenges. These limitations and challenges include, but are not limited to:

- Generating high false-positive rates: AIDS tends to produce more false positives compared to SIDS. This presents a major challenge that leads to model inefficiency, alert fatigue, and unnecessary human interventions, thus requiring ongoing research contributions.
- Inability to capture complex spatial relationships and inherent graph structures related to network threats: Traditional machine learning models treat network traffic as flat, sequential data, making it difficult to accurately capture complex relationships and interdependencies between network nodes and their communications. Such spatial, graph-structured information is crucial for identifying unusual relational patterns associated with threats that can potentially lead to network attacks, including insider threats [6].
- Limitations in handling long-term dependencies in network traffic: Subtle threats that develop over a long period present a significant challenge to current anomaly detection models, such as those based on RNNs, due to problems like vanishing gradients [7]. However, identifying such threats is increasingly important in the current network security landscape.
- Adversarial robustness: Current threat detection methods often lack resilience against adversarial attacks. This makes them vulnerable to evasion techniques used by sophisticated network attacks to manipulate machine learning models with malicious data [6].
- Scalability and high dimensionality of network traffic data: Ongoing research is necessary to process network data at large scales effectively and promptly. Even the typical computer network generates network traffic with high throughput, resulting in large volumes of data that current models struggle to analyze efficiently, leading to performance bottlenecks. Therefore, achieving low latency with high network throughput remains a challenge for traditional ML and DL models. Additionally, the high dimensionality of network data leads to the curse of dimensionality, where the

effectiveness and generalizability of distance-based algorithms decline as the input data dimensions increase, resulting in increased computational complexity, overfitting, spurious correlations, and sparse clusters [8].

- Covert channel detection: Traditional ML models often miss covert channel attacks that use hidden communication paths, resulting in data theft [9].
- Challenges of dynamic and evolving network environments: Real-world networks must often be restructured with different topologies to meet organizational requirements, resulting in variations in their traffic patterns over time. However, traditional anomaly detection models usually assume the conceptual structure of their network environment to be static or slowly changing, making them less effective in the real world.
- Limited interpretability and transparency: Transparency is important when making cybersecurity-based decisions. However, many ML models, particularly deep neural networks, cannot support transparency because they act as black boxes. This lack of transparency and interpretability can make it challenging for security analysts to understand the reasoning behind the detected anomalies, especially those that are previously unknown.

1.2. GNNs

GNNs are a class of deep neural networks designed to handle graph-structured data, where entities and their interrelations are represented as nodes and edges in the graph. Unlike conventional neural networks, which operate on fixed-size input arrays where data is treated as independent, identically distributed samples, GNNs have the advantage of capturing complex interdependencies and interactions between their nodes through the process called message-passing. This allows GNNs to learn rich, hierarchical representations of graph-structured data, hence making them effective in tasks such as node classification, graph classification, and link prediction. Thus, GNNs are increasingly being employed in domains such as social network analysis and drug discovery, where problems can be accurately modeled as graphs [10].

The Potential of GNNs for Network Threat Detection

- Modeling complex relationships and patterns: GNNs have the potential to successfully capture the interdependencies and relationships in computer networks, including LANs, where nodes represent network entities such as hosts or IP addresses, and edges represent the relationships among the nodes, which can be communication flows, network sessions, etc. This unique characteristic of GNNs can be particularly useful for identifying network abnormalities that can span across multiple coordinated entities and their complex interactions, which traditional ML models, such as SVMs or clustering algorithms, struggle to achieve [11]. Furthermore, GNNs can be employed to identify hidden network patterns, such as the ones presented by covert channels.
- Scalability: Computer networks are known to produce a large amount of data. However, GNNs can handle graph data at large scales, making them suitable for typical network anomaly detection tasks. With appropriate architectural or sampling strategies, GNNs can be scaled to accommodate additional data without a significant loss of performance, demonstrating their inherent robustness qualities [12].
- Dynamic graph construction: Automatic dynamic graph construction techniques can be employed to address the challenge of dynamic and evolving network environments, allowing real-time updates to the graph as new data is received [13]. This improves the robustness of GNNs compared to traditional static ML models.

1.3. Transformers

The Transformer architecture was introduced in the paper titled "Attention is All You Need" [14], and belongs to the family of sequential deep learning models. Transformers use a mechanism called "Self-attention" which enables them to measure the importance of different elements with respect to each other within long input data sequences. This allows the Transformer to capture long-range dependencies and relationships in the input data, such as long-form text prompts, efficiently. Therefore, Transformers are particularly effective in large datasets with complex patterns, due to their ability to handle long sequences in parallel and focus on relevant parts of the data through the self-attention mechanism [14].

In recent years, Transformers have given rise to Large Language Models (LLMs) and generative artificial intelligence (GenAI). However, despite the huge potential of the Transformers, they have not been largely explored in the cybersecurity literature. The aforementioned traits of Transformers are highly suitable for analyzing large network traffic data presented as network flows (e.g., source and destination IP addresses, ports) or sequences of network events (e.g., login attempts, file transfers) to detect sophisticated network threats.

Potential of Transformers for Network Threat Detection

- Handling long sequential data: Transformers are designed to process sequences of data and capture dependencies over long ranges. Hence, they can be potentially used in analyzing network traffic to identify sophisticated threats that evolve slowly by detecting abnormal patterns growing over extended periods [14].
- Attention mechanisms: By utilizing its attention mechanisms, the Transformer model can focus on relevant parts of its input sequence. This helps highlight important features that correspond to malicious behaviors and thus enhances the precision by reducing false positives.
- Parallel processing: Balancing real-time detection vs. accuracy is critical in any anomaly detection mechanism. Achieving low latency in high-throughput environments is crucial for supporting tasks such as network traffic analysis, which is challenging for traditional sequential models, including RNNs. However, Transformers can process entire sequences simultaneously, enabling low-latency detection in network environments.
- Versatility in data handling: Furthermore, Transformers are highly versatile and can be applied to handle different input modalities, enabling multi-faceted network threat analysis, which can improve model robustness [15].
- Interpretability and transparency: Attention mechanisms in Transformers can be inferred to highlight the most relevant parts of the input data that they paid attention to during the threat detection process. This can be exploited to interpret the Transformer's decision up to some degree, and thus, the overall transparency can be improved.

1.4. Reinforcement Learning

Reinforcement Learning (RL) is a machine learning algorithm where an RL agent learns to make decisions about its environment through a reward and penalty system. The RL agent receives a reward from the environment when its action is expected or desirable, or else, it gets a penalty. It can be a powerful learning tool for solving complex network-related issues where traditional, static systems may fail, as it improves system performance over time by continuously observing how the network behaves, exploring new actions, and updating its policies for long-term outcomes. Thus, RL has the potential to adaptively learn optimal policies for network threat detection, especially when networks are dynamic or when labeled data is limited. Moreover, RL algorithms can be jointly employed with

other ML and DL models as core IDS components to support adaptive and proactive decision-making [16].

Potential of Reinforcement Learning in Network Threat Detection

- Adaptive learning: RL enables IDS to adapt to new and evolving threats by continuous learning via interactions with the network environment. This ensures the responding strength of RL-enabled IDS against changing attack patterns.
- Optimization of detection strategies: By learning the optimal policies, RL can optimize the trade-offs between detection accuracy, false positives, and computational efficiency. This can significantly improve the overall performance and resilience of network threat detection systems [17].
- Decision making in real time: RL can improve real-time IDS decisions and responses based on the current state of the network [18].
- Adapting optimal hyperparameters of the base models: RL can effectively choose optimal hyperparameters of ML and DL models when set up in hybrid environments.

1.5. Aims and Contributions of the SLR

1.5.1. Aims

This SLR aims to conduct a thorough examination of the role of GNNs, Transformers, and RL algorithms in network threat detection by systematically analyzing and synthesizing existing peer-reviewed literature. Thus, it seeks to:

1. Explore how GNNs, Transformers and RL algorithms have been employed in current network threat detection literature, standalone or in hybrid manners.
2. Explore the effectiveness of these models and algorithms in network threat detection, especially in detecting sophisticated modern network threats.
3. Identify key trends, strengths, limitations, and gaps in existing approaches.

1.5.2. Contributions

1. Providing a potentially novel conceptual framework for classifying network threats inspired by the MITRE ATT&CK model and cyber kill chain that clearly separates network threats from network attacks and maps such network threats to their respective MITRE ATT&CK tactics.
2. A domain-specific contextualization of network threat detection focused on conventional Ethernet and TCP/IP-based LANs.
3. A structured and detailed synthesis of how GNNs, Transformers and RL algorithms are currently employed in conventional LAN-based network threat detection.

2. Literature Review

2.1. Traditional Machine Learning Methods for Network Threat Detection

Traditional ML classifiers have long been used in intrusion detection systems. ML algorithms like Random Forests, Decision Trees, K-Nearest Neighbors, Support Vector Machines, and XGBoost are commonly employed as network traffic classifiers. Although these ML models perform well on classic benchmarks, such as KDDCup'99 or NSL-KDD, they produce lower performance on more modern datasets, such as CICIDS2017/2018 and UNSW-15, often achieving accuracies between 70% and 90% [19]. Specifically, Ref. [19] reported that Random Forest (RF) obtained 74.87%, XGBoost 71.43%, Decision Tree (DT) 74.22%, and K Nearest Neighbor (KNN) 71.10% accuracy, respectively, on the UNSW-NB15 dataset at a reduced feature set of 11 chosen using a Random Forest Feature Importance. The study also stated that traditional ML models frequently struggled to generalize to unseen data. According to [20], traditional ML classifiers often struggled with complex network

traffic and required dimensionality reduction, class balancing, and ensemble methods to remain competitive with DL models. Among these popular ML threat classifiers, tree-based ML models are popular for their interpretability and inference speed. Furthermore, RF is less prone to overfitting than single tree-based methods. As explained above, they all struggle to generalize to complex modern network data. However, when combined with deep learning models, such DL hybrids were shown to increase RF-only performance. In the study in [21], an ensemble RF + Deep Sparse Autoencoder (DSAE-RF) achieved 99.83% accuracy on CICIDS2017. Overall, tree-based methods struggled to perform better without richer features or by creating ensembles with deep learners, although they remained computationally cheap at inference when compared to more computationally intensive models like SVMs [21].

According to [20], SVMs have been widely used in network threat detection. However, they perform poorly when presented with very large or imbalanced feature sets and often require data to be preprocessed using feature scaling techniques or PCA to achieve better performance. However, such data preprocessing frequently results in computationally intensive pipelines, and thus is insufficient for dealing with large-scale network traffic data. The study in [22] employed an SVM-based threat anomaly detection approach for both binary and multiclass classification tasks, evaluating it on the UNSW-NB15 dataset. Despite using a Radial Basis Function (RBF) to map a low-dimensional space to a high-dimensional space, the study's performance metrics demonstrated moderate performance. A maximum accuracy of 85.99% was obtained for the binary classification, where the threat classes of UNSW-NB15 were merged into a single class. However, the multiclass accuracy was relatively low, at only 75.77%. According to [23], despite their naïve feature independence, which often results in moderate performance on rich and large network data, Naive Bayes (NB)-based network threat classifiers generally train faster than other ML models. However, Ref. [23] also noticed that NB usually generalizes better to previously unknown threats and attacks due to its nature of making fewer assumptions.

Overall, traditional ML network threat classifiers are computationally cheaper and more interpretable than their DL counterparts. However, their performance on modern network threats and modern network data often falls under moderate to underperformance categories, thus limiting their practical application in IDS. According to the literature, careful feature selection or combining with DL hybrids can elevate the performance; however, this results in increased computational power, detection latency, and added complexity [3]. Despite these model design advancements, sophisticated modern network threats, such as APTs, zero-day attacks, and subtle insider threats, can still bypass traditional ML models, thus justifying ongoing research into more advanced and robust deep learning architectures that can generalize to modern network threats.

2.2. Traditional Deep Learning Methods vs. GNNs and Transformers for Network Threat Detection

Common spatial DL models such as CNNs, sequential DL models like RNNs, including their gated versions such as LSTM and GRU, and feedforward Deep Neural Networks have been frequently applied to network threat and network attack detection. Furthermore, hybrids of spatial and sequential models have been used in IDSs [24]. According to [25], many DL models achieved high performance in old network datasets such as the KDD Cup 99. Hybrid CNN-BiLSTM models achieved 99.78% accuracy, 99.58% precision, 99.72% recall, and 99.73% F1-Score on NSL-KDD. In [3], CNN and LSTM models achieved an accuracy of 98% when trained and evaluated using the SMOTE-balanced CICIDS2017 dataset.

A CNN-GRU and a CNN-Bi-LSTM achieved F1-scores of 92.05% and 90.08%, respectively, for the CICDS2017 dataset in a binary classification task, whereas their proposed E-GraphSAGE model achieved an F1 score of 99.76%. The dataset was not class-balanced

using any sampling method or synthetic records generation. Instead, to preserve the original class imbalance in the dataset, a sub-sample was selected. According to the study, this reflected the ratio of benign to malicious network flows in the real world, which was then used to evaluate various DL architectures and, thus, various DL models; XGBoost_LSTM, AT_LSTM, and CNN_GRU reported F1-scores of 89.96%, 90.36%, 92.20%, and 93.61%, respectively. The proposed E-GraphSAGE reported an F1-score of 98.65%, outperforming all DL-based approaches [26].

A Transformer architecture was evaluated using several network intrusion datasets against various network threat detection scenarios in the study in [26]. First, the model was tested on the NSL-KDD dataset in a class-wise classification problem, where it achieved high performance across Analysis, Backdoor, Fuzzers, Reconnaissance, Shellcodes, and benign classes, reporting accuracy ranging from 97.0% to 99.9% and recall ranging from 98.1% to 99.7%. When evaluated on the UNSW-NB15 dataset, the Transformer achieved exceptionally high performance in a class-wise classification problem, with accuracies ranging from 99.22% to 99.82% and recall values ranging from 98.39% to 99.38% across the Normal, Probe, U2R, and R2L classes. Furthermore, the model maintained high precision, ranging from 98.24% to 99.38%, and F1-scores from 98.56% to 99.30%, across all classes of the UNSW-NB15 dataset, confirming its strong generalization capabilities in various network threat detection scenarios. Additionally, when compared with multiple state-of-the-art (SOTA) baselines using an NSL-KDD binary classification task, the proposed Transformer outperformed all of them. When evaluated on the NSL-KDD dataset, SOTA models reported accuracy of 91.86% (ResNet34), 97.64% (RNN), 98.71% (LSTM), and 98.79% (Vision Transformer), whereas the proposed Transformer again surpassed all baseline SOTA models with a clear margin, achieving 99.25% accuracy, 99.07% precision, 99.02% recall, and 99.05% F1-score. Furthermore, when evaluated on a UNSW-NB15 binary class task, SOTA models reported 91.00% accuracy for ResNet34, 94.2% for RNN, 95.0% for LSTM, and 95.9% for Vision Transformer, with recall values ranging from 90.1% to 96.2% across all the models. However, the proposed Transformer was not evaluated on the UNSW-NB15 dataset in a binary classification setting. As the paper stated, none of these comprehensive performance evaluations employed any class-balancing technique in any of the datasets. However, focal loss was used as the loss function.

According to [27], during an APT evaluation performed on the CSE-CIC-IDS2018 dataset, a CNN achieved 93.01% accuracy, whereas the GCN achieved 95.96% accuracy. In the study in [28], the Transformer architecture was benchmarked against RNN/CNN models using the CICIDS2017 dataset, which was oversampled to address minority class issues with SMOTE. The class-wise performance of the proposed Transformer, evaluating different network threats, demonstrated near-perfect performance for the benign, PortScan, Patator-FTP, Patator-SSH, and Brute-Force classes, consistently reporting accuracy, precision, recall, and F1-scores in the very high 97–99% range. Infiltration class was also detected at high levels, ranging from 95% to 99% across all reported performance metric types, and the Bot attack class was detected at around 92%. According to the paper, the most challenging class to detect was SQL Injection, with all performance metrics values ranging from 47% to 60%. The authors continued the performance evaluation by conducting a comparison between SOTA ML baselines and their proposed model, demonstrating that the Transformer architecture could outperform all the SOTA RNN and LSTM models in global accuracy across all classes. The Transformer further outperformed all the SOTA models in macro-averaged metrics, such as precision, recall, and F1-scores, with reported values of 99.35%, 98.98%, 98.83%, and 99.17%, respectively.

The CERT Insider Dataset (v4.2) was evaluated using several baseline DL and GCN models in the study in [29]. The proposed session-graph GNN was able to achieve near-

perfect performance in all reported metrics, including 99.56% TPR and 0% FPR, detecting insider threats in the AB-II (Abnormal Behavior-II) subset. The authors employed 7 heuristic rules to build an Associated Session Graph (ASG) with session nodes, core/boundary nodes, and inter-session edges. When a comparison evaluation was conducted with traditional DL baselines such as CNNs and LSTM models, which reported moderate performance with TPRs ranging from 76% to 96% and FPRs between 3% and 14% and with a standard GCN, which reported 97.12% TPR and 3% FPR, the proposed ASG-based GNN significantly outperformed them all. The study reports high performance, including 99.56% accuracy, 99.54% precision, 99.53% F1 score, 99.56% TPR, 0% FPR, and 99.56% AUC for the ASG-based GNN. This study demonstrates that when vanilla GNN architectures are enhanced with encoded relationships across sessions, they are highly capable of detecting insider threats compared to traditional sequence-based DL models.

2.3. GNNs in Network Threat Detection

The authors of [30] mentioned that IDS-based traditional machine learning models often fail to learn network interdependencies, and resulting incorrect predictions usually contain a high number of false positives and negatives. The authors further stated that the traditional IDS often only used individual samples of network traffic and host logs to extract abnormal patterns and ignored the relationships between the associated network nodes. To address this issue, the authors propose GNN-IDS, which can represent both static and dynamic network attributes by employing attack graphs and real-time measurements. According to the study, the proposed GNN-based IDS can learn network connectivity and measure the importance of neighboring network nodes and their aggregated features, producing more accurate and reliable predictions with fewer false detections. The solution was evaluated with two synthetic datasets, including one dataset generated from CICIDS 2017 [30]. The proposed Graph Convolution Network with learnable Edge Weights (GCN-EW) version of the model achieved 86.84% Precision, 95.47% Recall, 90.53% F1-score, 98.94% AUC, and 3.63% FPR in the first synthetic dataset, and it achieved 94.14% Precision, 98.02% Recall, 95.97% F1-score, 99.65% AUC, and 1.40% FPR in the CICIDS2017-based synthetic dataset. The proposed GAT version achieved 86.61% Precision, 95.49% Recall, 90.39% F1-score, 98.87% AUC, and 3.73% FPR in the first synthetic dataset, and 94.74% Precision, 98.72% Recall, 96.62% F1-score, 99.72% AUC, and 1.27% FPR in the CICIDS2017-based synthetic dataset.

In [26], the authors suggest a novel preprocessing technique to standardize network flow data, thus improving GNN-based malicious network flow detection. In their approach, they construct graph nodes from network flow features and graph edges based on node IP relationships. Furthermore, the authors propose a hybrid GNN model combining Graph Convolution Networks (GCNs) and SAGEConv to enhance the detection process. The proposed hybrid model achieved high accuracies of 99.76% and 98.65% for the CIC-IDS2017 and UNSW-NB15 datasets, respectively [26].

The following study focuses on a GAT within a federated learning framework to improve network attack detection while preserving data privacy. The authors use log density information to construct a graph representation of the network, allowing them to capture complex node relationships and network behavior over time, thus improving detection accuracy. The attention mechanism of the GAT employs fingerprint-weighted Jaccard similarity to calculate the attention scores. The proposed Federated Graph Attention Network (FedGAT) enables collaborative model training across distributed devices. Experimental validation conducted using the NSL-KDD dataset shows that FedGAT achieves exceptionally high accuracy levels, ranging from 99.983% to 99.998%, by effectively detecting cross-level and cross-department attacks while maintaining privacy [31].

Various APT datasets (e.g., Cadets-Unicorn, Streamspot, Theia-E3, Streamspot) were evaluated in [32]. The proposed GNN achieved perfect detection across several of the datasets and near-perfect detection across most of them. When benchmarked against prior methods (e.g., Kairos-GNN, Threatrace), they reported F1 scores in the range of 83–96%. The proposed GNN was able to surpass all previous techniques by modeling both time and graph structure. Multi-stage APTs inherently possess a temporal dynamic, which is why the proposed GNN, called "CONTINUUM," is successful in this study.

In the following paper, the authors explored the suitability of nested graph structures for enterprise-level networks, where the outer graph was constructed based on host communications and local activities within each host (e.g., event graphs form the inner graphs). According to the paper, the proposed NestedGNN is the first GNN designed for nested graphs consisting of three types of layers called inner GNN layers, nested graph layers, and outer GNN layers [33].

The applicability of Spatio-Temporal Graph Convolutional Networks (STGCN) for insider threat detection using the psychometric features and user behavior logs of the CMU CERT 4.2 dataset is evaluated in [34]. First, the authors constructed a knowledge graph to model employee interactions within the network, where users formed the nodes and their communications or access to events formed the edges in the graph. The authors compared three graph neural networks: a standard GCN, an STGCN, and a Capsule GNN. In STGCN, graph convolution layers were incorporated to capture spatial information, while the LSTM layers were incorporated to capture sequential patterns in user behavior over time. STGCN achieved an accuracy of 94% and demonstrated robustness in detecting insider threats with minimal false positives by focusing on complex temporal anomalies and subtle behavioral patterns associated with them. Moreover, authors highlighted the importance and the effectiveness of combining GNNs with the layers or model architectures capable of modeling temporal patterns, particularly for insider threat detection [34].

Ref. [35] proposes a taxonomy for organizing existing literature on GNNs in relevance to IDS. The study categorized the analyzed literature into three main categories; Graph Construction, Network Design, and Model Deployment whereas, Graph Construction focused on converting raw network data into static or dynamic graph structures, Network Design concentrated on the architecture of the GNN model such as GraphSAGE or Graph Attention Networks (GAT), and finally Model Deployment dived into the real world challenges in deploying GNNs in IDS such as scalability issues, optimizing the required computations and integration with other models for improved performance.

2.4. Transformers in Network Threat Detection

IDS-MTran proposes a novel multi-scale intrusion detection framework. The framework first utilizes multi-scale convolution operators, deployed as 1×1, 3×3, and 5×5 kernels, to extract diverse network traffic feature representations, followed by a Patching with Pooling (PwP) module to reduce noise and enhance feature interactions. Then, a Transformer with multiheaded attention and positional encoding is utilized in the framework to capture global dependencies across the scales. The Cross-Feature Enrichment (CFE) module in the framework integrates multiscale features via up/down sampling and distillation. During the training process, the focal loss was utilized to address the significant data imbalance of the malicious and benign samples across multiple network threats and attack categories. Thus, performance evaluations of IDS-MTran on the NSL-KDD, CIC-DDoS 2019, and UNSW-NB15 datasets show that the model is able to achieve over 99% accuracy in multiple network threat categories [36].

The "Flow Transformer" proposed by [37] is built on TensorFlow and provides a modular pipeline consisting of three components: dataset ingestion, preprocessing, and

model construction. The first component supports various flow-based network traffic data formats (e.g., CICIDS Flowmeter, NetFlow V5), handling both categorical and numerical flow features such as protocols, port numbers, flow duration, and packet counts. The model's binary classification evaluation on the CICIDS-2018 dataset demonstrated an accuracy level of 98% with a basic Transformer architecture consisting of 2 layers, 2 attention heads, and 128 features [37].

A neural Transformer-based model for detecting real-time zero-day threats in network traffic data is proposed in [38]. First, network flow features such as protocol type, packet size, source and destination IPs are embedded into the vector space to be processed by the Transformer module. Then the Transformer utilizes its self-attention mechanism to calculate the importance of different parts of network packet sequences, identifying complex dependencies and patterns resulting from threat-related anomalies. The authors reported that their model requires low computing resources, specifically 55% CPU and 40% GPU, to achieve a low latency of 25 ms, concluding its suitability for real-time applications. The model was evaluated using CICIDS2017 and a simulated network traffic dataset that included benign activities such as file transfers, normal browsing, video streaming, and various network threat types, including zero-day intrusions. Zero-day traffic was used for evaluating the model's performance on previously unseen threats. The proposed model achieved 96% accuracy, 94% precision, 92% recall, and a 93% F1 score. Furthermore, the ROC-AUC score was 0.97, with precision and recall trade-offs indicating the model's robustness in imbalanced datasets such as CICIDS2017 and the simulated dataset.

The Markov property assumes that the next state depends only on the current state. However, in the context of network security, the threats that develop over time with a historical context cannot be accurately modeled according to the Markov property, and thus, such threat detection techniques based on RL algorithms cannot depend solely on the previous network state. Ref. [39] tackles this issue by proposing a model called "Decision Transformer". Unlike traditional RL methods, such as Q-learning or policy gradients, the proposed methodology treats the RL task as a supervised learning problem. Thus, the authors constructed an RL-Transformer architecture with past trajectories of rewards, detection decisions, and network packets to produce future detection actions. The proposed method also introduces a trade-off between timeliness and detection accuracy by using a reward function that rewards accurate and timely decisions and penalizes inaccurate and delayed ones. The authors also employed an autoencoder to compress packet payload features into embeddings to process packet sequences of arbitrary lengths more efficiently. Once trained on offline datasets such as UNSW-NB15, experimental results showed that decision Transformers achieved higher accuracy and lower latencies during the detection process across different sampling policies, including Expert, Medium, and Random, demonstrating their real-time applicability and robustness.

Ref. [40] proposes a novel cyber threat detection approach through Vision Transformers (ViTs) and Knowledge Distillation (KD). The proposed method encodes input data features into color imagery representations, where each feature is mapped into a three-channel RGB pixel, followed by the extraction of explainable imagery signatures of cyber threats using a ViT teacher model. Here, ViTis self-attention mechanisms are used to record relationships between imagery patches and generate explainable attention maps, followed by creating imaginary representations using these attention maps and feeding them to a CNN student model during the KD process. The CNN student model is then trained to mimic the teacher's output using a combined loss function made with cross-entropy and Kullback–Leibler (KL) divergence. The authors evaluated their model on four benchmark cybersecurity datasets named CICMalDroid20, CICMalMem22, NSL-KDD, and UNSW-NB15, achieving overall accuracies of 87.2%, 82.6%, 82.9%, and 78.4%, respectively.

Although the performance of the ViTs is not top-notch, the study demonstrates the various ways that Transformer architectures can be employed in network threat detection.

A comprehensive survey of 118 papers, including 40 preprints published between 2017 and 2024, has been conducted by [41] to focus solely on Transformers and Large Language Models (LLMs) used in network intrusion detection tasks. The survey has identified key technologies used in these studies, such as attention based architectures for feature extraction, CNN/LSTM hybrid Transformers for combining spatial and sequential data analysis, GAN-Transformers for generating synthetic datasets to address the issue of imbalanced datasets, Vision Transformers (ViTs) for processing network traffic as 2D images, and LLMs like BERT and GPT to enhance contextual understanding and the predictions. The survey further provides several recommendations for future research, including the development of LLM agents with RL components, the use of multimodal inputs, and knowledge distillation.

2.5. RL Algorithms in Network Threat Detection

Ref. [42] proposes a Deep Reinforcement Learning (DRL) framework using three advanced RL techniques, which are Deep Q-Network (DQN), Actor–Critic, and Proximal Policy Optimization (PPO). The study further conducted a performance comparison against the traditional Q-Learning algorithm. Open port attacks, spoofing, keylogging, and finally credential theft were replicated and simulated in a controlled environment using the MITRE Attack framework. Their key findings showed that the Actor–Critic outperformed other RL algorithms by achieving the highest success rate of 0.78, the fewest iterations of 171, and the highest average reward of 4.8. Thus, the paper further demonstrates that DRL algorithms, such as Actor–Critic, are highly effective in learning and detecting dynamic network attacks and potential threats [42].

Ref. [43] has conducted a comprehensive review of the applicability of DRL in cybersecurity, focusing on threat detection and protection. The paper focuses on DRL in both Network Intrusion Detection Systems (NIDS) and Endpoint Detection and Response Systems (EDR). The key technologies used in the reviewed work can be categorized into DQN, Double DQN, Dueling DQN, and policy-based methods, such as Deep Deterministic Policy Gradient (DDPG) and Proximal Policy Optimization (PPO). The paper covers different applications of DRL in cybersecurity, such as anomalous and benign network traffic classification, botnet detection evasion, and adversarial attacks on malware classifiers (Yang, Acuto, Zhou, & Wojtcza, 2024 [43]).

ID-RDRL integrates a DRL model with a Multilayer Perceptron (MLP) to classify network threats out of benign network traffic flow data. The first phase of the approach is a feature selection technique called Recursive Feature Elimination (RFE) combined with a Decision Tree, which filters out 80% of the redundant features in the CSE-CIS-IDS2018 dataset, thus significantly reducing the computational effort. The next phase is the Deep Q-Network classifier, which operates within the Markov Decision Process (MDP) framework. It begins with the Mini-Batch module, which processes the input dataset by treating features as states and labels as actions, creating tuples labeled as “state”, “action” and “next state”. These tuples are then fed into a Convolution Neural Network (CNN), followed by the MLP to extract feature information. Then the DQN utilizes an ε-greedy algorithm to select actions (i.e., classifying traffic as normal or attack) based on the current state to maximize the expected reward. The reward function is set to provide a reward of 1 if the predicted action matches the actual label, thus a 0 otherwise, with a low discount factor of 0.01 to focus on immediate rewards. The authors’ reason behind their reward function strategy is the low correlation between the states in the dataset. The backpropagation through the DQN network updates the Q-function, enabling the model to learn the optimal policy for

the classification task. The experimental evaluation results of the study demonstrate that the proposed ID-RDRL can achieve a 96.2% accuracy and an F1 score of 94.9% [44].

Ref. [45] explores the vulnerability of DRL to adversarial attacks by employing a state-of-the-art DRL IDS agent in an adversarial environment using a Deep Q Network for classification tasks. The model was trained using the NSL-KDD dataset and evaluated against two adversarial attack methods, namely Fast Gradient Sign Method (FGSM) and Basic Iterative Method (BIM), both in their targeted and untargeted forms. The results show that adversarial examples degraded the detection performance of the Deep RL agent. The study further reported that targeted adversarial attacks were more effective in influencing malicious traffic to be detected as benign.

The Multi-Agent Feature Selection Intrusion Detection System (MAFSIDS) is a network attack classification framework proposed by [46], which consists of two main modules: the feature self-selection algorithm and the Deep Q-Learning module. The feature self-selection module employs a multi-agent approach to reduce the 2N feature selection space to N agent representations and thus reduces redundant features by 80% compared to the original dataset. Furthermore, a GCN is employed in the study to extract deeper features from the data to improve feature representation accuracy. The function of the Deep Q-Learning module is to utilize Mini-Batches to encode data, allowing the RL to be applied in a supervised learning context. According to the evaluation results, the framework achieved accuracies of 96.8% and 99.1%, and F1-scores of 96.3% and 99.1% on the CSE-CIC-IDS2018 and NSL-KDD datasets, respectively.

2.6. Taxonomy of the Research Domain

A high-level summary of the research domain covered in this paper is presented in Table 1.

Table 1. Taxonomy of the research domain.

Approach Type	Subcategory/Technique Group	Representative Models/Methods
Signature-Based Detection (SIDS)	Rule and Signature-Based Systems, Pattern Matching Databases	Signature rules (Snort, Suricata)
Anomaly-Based Detection (AIDS)	Traditional Machine Learning	Supervised: SVM, RF, DT, KNN; Unsupervised: K-Means, One-Class SVM, Isolation Forest; Semi-/Self-Supervised: Autoencoders, VAEs
	Deep Learning	Spatial: CNNs; Sequential: RNN, LSTM, GRU; Hybrid Spatial–Sequential: CNN–LSTM, CNN–GRU
	GNNs	Static: GCN, GAT, GraphSAGE; Spatio-Temporal: STGCN, E-GraphSAGE; Hierarchical: NestedGNN; Federated: FedGAT
	Transformers	Sequential: Vanilla, Temporal; Hybrid/Multiscale: IDS-MTran, FlowTransformer; Vision: ViT; Decision: RL-integrated Transformers
	Reinforcement Learning	Value-Based: Q-Learning, DQN; Policy-Based: Actor–Critic, PPO; Hybrid: DQN + MLP, Feature-Selection RL; Multi-Agent: MAFSIDS

2.7. Cost-Effectiveness Comparison Between Traditional ML and Deep Learning Models vs. GNNs, Transformers, and Reinforcement Learning Models

Across the literature, there is no consistent reporting format for cost or resource usage. Several recent Transformer, GNN, and RL papers already present partial resource measurements or engineering fixes. However, the field lacks standardized, quantitative resource reporting across studies. Therefore, a definitive, numeric cost-effectiveness ranking is not yet possible from the available literature.

However, the literature cited in this manuscript provides qualitative evidence and some concrete evidence that:

1. Advanced models (GNNs, Transformers, RL) impose higher preprocessing, training, and operational costs than classic ML baselines, and
2. Engineering mitigations (lite models, input encodings, model distillation, or hybrid supervision + RL) are being proposed to reduce these costs.

2.7.1. Data Collection and Preprocessing

- Traditional ML and DL baselines: Surveys of classical and ML-based IDS research note that many traditional ML and DL pipelines with engineered features operate on compact flow or summarized tabular features (e.g., NetFlow/summary statistics), which reduces preprocessing and storage overhead relative to heavy representation pipelines [19,20].
- Heterogeneous or nested graphs require additional preprocessing: Reviews of GNN methods emphasize that applying GNNs to network security requires an explicit graph construction step (hosts/sessions/processes $\rightarrow$ nodes; flows/relations $\rightarrow$ edges) and design choices for heterogeneous or nested graphs; those preprocessing steps incur extra engineering cost (feature to graph mapping, edge definition logic) compared to flow/tabular pipelines and homogeneous graphs [26,30,32]. The literature also discusses the need to maintain or update graph structures for streaming data, which adds ongoing processing overhead [30,34].
- Transformer tokenization/sequence handling increases preprocessing: Transformer-based NIDS research explicitly discusses the need to prepare sequential inputs (tokenization/embedding, padding/striding, or patching) for long flows or logs. This processing can increase memory usage and I/O when sequences are long, and authors have proposed input encodings and striding/patching to reduce this cost [36,38]. The Flow Transformer paper explicitly reports that careful input encoding and head selection can affect model size and speed. It further demonstrates design choices that reduce model size and improve training/inference time [37].
- RL environment and trajectory generation cost: RL approaches to IDS typically require the definition of an environment (state, action, reward) and the generation of agent trajectories for training. Building faithful training environments or realistic simulators adds engineering and annotation costs not normally required for supervised classifiers [42,44].

2.7.2. Training Compute and Sample Efficiency

- Traditional ML: These models are generally inexpensive to train and to run in inference, which is reiterated in literature comparing ML baselines with deeper models [19,20].
- Conventional DL: Typical deep baselines require GPU acceleration but are often tractable for common IDS datasets. Comparative studies report DL baselines as a middle ground in terms of compute cost versus performance [3,25].
- GNNs: These models require moderate to high compute demands, depending on design. Methodological reviews of GNNs report that message passing and heteroge-

neous/hierarchical graph designs increase memory and compute demands. Temporal GNN variants and nested/hierarchical graphs further increase training complexity and memory usage [30,34]. These studies demonstrate that computational cost as a key practical challenge when scaling GNNs to high-throughput network data.

- Transformers: Reviewed Transformer literature notes that Transformer models, especially large pre-trained or long sequence variants, cause higher GPU memory consumption and longer training times. NIDS Transformer frameworks (e.g., Flow Transformer) report measurable model size and runtime tradeoffs, and show that careful design, such as input encoding and classification head choices, can reduce model size by >50% and improve training/inference time without loss of detection performance in experiments [36–38]. Thus, Transformers can be computationally heavy, yet they are open to engineering optimizations.
- RL: Deep RL studies for IDS warn that RL training can be sample inefficient and require many episodes to converge. This leads to large cumulative compute (many interactions/episodes) compared to supervised training, and authors recommend sample-efficient RL approaches or hybrid supervised + RL designs for practicality [42,44,46].

2.7.3. Online Updating and Adaptivity

- Traditional ML: Traditional ML retraining is typically fast. However, naïve retraining does not handle concept drift well. Literature recommends periodic retraining or incremental approaches while noting limitations for evolving threats [19,20].
- GNNs: Some temporal GNN designs support incremental graph updates. However, GNNs generally require recomputing node/edge representations when topology or relations change. Thus, the literature highlights the engineering cost of maintaining up-to-date graph pipelines in streaming settings [30,34].
- Transformers: Fine-tuning Transformers can support adaptation to new data. However, large models remain costly to update and the literature reports distilled, shallow, or strided variants as practical choices to reduce update overhead [36,38].
- RL: RL is naturally framed for continual adaptation (policy updates). However, the literature warns about stability and the risk of degraded behavior during continued online learning [45,46].

2.7.4. Operational Complexity and Maintainability

- Pipeline complexity: As mentioned above, existing literature highlights that GNNs and Transformer systems typically demand more complex data pipelines, such as graph processing frameworks for GNNs and long sequence tokenization and large model serving for Transformers. And thus, higher operational and maintenance costs compared to classic ML pipelines [26,30,36].
- Safety & robustness costs: RL studies document adversarial concerns such as reward poisoning and exploitation during learning, which require defenses and monitoring when RL is deployed online. These concerns highlight the need for additional safeguards [45].

Table 2 provides a high-level estimation and comparison of the costs associated with Traditional ML/DL models, GNNs, Transformers, and RL algorithms.

Table 2. High-level cost comparison of traditional ML/DL, GNNs, Transformers, and RL models.

Cost Dimension	Traditional ML/DL	GNNs	Transformers	Reinforcement Learning
Data collection/preprocessing	Low: Flow/tabular extraction, well established pipelines [19,20].	High: Graph construction (node/edge design), heterogeneous/nested graphs, and dynamic graph maintenance increase overhead [26,30,32,34].	Moderate–High: Tokenization/embedding and long sequence handling, input encoding choices affect size/speed [36–38].	High: Environment and trajectory design, reward engineering, and realistic simulator development contribute to cost [42,44].
Training compute/sample efficiency	Low	Moderate–High: (message passing, heterogeneity, and temporal variants add cost) [26,30,34].	High: High cost for large/long sequence variants. Engineering (lite/strided) can reduce cost [36–38].	High: Sample-inefficient iterative training may require many episodes [42,44,46].
Online updating & adaptivity	Low cost to retrain but limited adaptivity [19,20].	Moderate-High: Supports temporal updates but operationally heavier to maintain graphs online [30,34].	High: Large models are expensive to update. Distillation/lite designs help [36–38].	High: Naturally adaptive, but online updates require full environment execution, monitoring, and safety controls; operational and computational costs are significant [45,46].
Operational complexity	Low	High: Graph pipelines, indexing, and dynamic updates contribute to cost [26,30,34].	Moderate-High: Model size and serving complexity contribute to cost [36–38].	High: Environment management, monitoring, and safety precautions increase the cost [45].

2.8. Research Gap and the Significance of the SLR

The preliminary literature review has clearly demonstrated the potential of GNNs, Transformers, and RL in modern network threat detection tasks. Modern, sophisticated network threats can consist of multiple stages or slowly evolve and be subtle, presenting a huge challenge to conventional intrusion detection systems equipped with traditional ML and DL models. Hence, there is a need for ongoing research into GNNs, Transformer architectures, and RL algorithms, as well as their applications in the highly critical cybersecurity domain.

However, as the above literature review clearly indicates, and despite recent research interest in the applicability of GNNs, Transformers, and RL for network threat detection, there is a significantly limited number of Systematic Literature Reviews (SLRs) covering the topic in the literature. The existing systematic literature reviews' main focus is on traditional ML and DL approaches. For example, a 2024 SLR on intrusion detection, which covers studies from 2018 to 2023, highlights that publications focusing solely on traditional methods are considered the top approaches to network threat detection [47]. Thus, the SLR covers models such as CNNs, SVMs, and decision trees, yet the emerging GNNs and Transformer models have been completely overlooked.

When an SLR about GNNs, Transformers, and RL algorithms appears in network threat detection literature, the focus would normally be only on reviewing a single model type. Thus, the preliminary literature research found that there are limited recent surveys. For example, one study conducted a comprehensive literature review on Transformers and Large Language Models (LLMs) in intrusion detection systems [41]. Another study surveyed the use of GNNs in IDS [35]. Thus, there is almost no systematic literature review, any other type of literature review, nor a survey done for all three types of models together or as a combination of two.

However, a comprehensive and comparative analysis of these models is simply not feasible with standalone single-model reviews. Therefore, to conduct a comparative investigation into the strengths and limitations of GNNs, Transformers and RL in network threat detection tasks, to uncover their key trends and technical gaps in current research, a focused SLR is timely and well justified. An SLR is further justified by the need to consolidate knowledge, as the current body of research on network threats, conducted using the models in question, remains highly fragmented. Such an SLR will also provide insights into the best practices for effectively using these models in the cybersecurity domain.

3. Methodology

3.1. Research Questions

3.1.1. Primary Research Questions

1. How are GNNs, Transformers, and RL used individually or in combination for network threat detection?
2. How do different model architectures, datasets, evaluation metrics, and types of network threats addressed compare across studies involving GNNs, Transformers, and RL?

3.1.2. Secondary Research Questions

1. What are the strengths, limitations, and gaps observed in current research utilizing GNNs, Transformers, and RL for network threat detection?
2. What are the observable trends that integrate GNNs, Transformers, and/or RL for enhanced network threat detection?

3.2. Scope of the Study

3.2.1. Network Context

Modern network infrastructures are increasingly hybridized, spanning LAN, cloud, Internet of Things (IoT), and other networks. However, the network scope in this review is restricted to conventional Local Area Networks (LANs), including both wired and wireless nodes and their interconnections operating under conventional Ethernet and TCP/IP architecture. Therefore, studies related to software-defined networks (SDNs), cloud-native networks, IoT networks, mobile networks (4G/5G), mobile ad hoc networks (MANETs), etc., were excluded. Moreover, unrelated noise from papers related to non-computer networks, such as networks on chips, UAV-based networks, social networks and specialized forms of interconnected networks, was filtered out from the scholarly database search results.

This deliberate restriction to LAN-based environments is grounded in both methodological rigor and conceptual focus. From a methodological standpoint, LANs provide a controlled and reproducible environment that has been most widely adopted in empirical network threat detection studies. Most frequently used benchmark datasets in the literature, such as UNSW-NB15, CICIDS2017, CSE-CIC-IDS2018, and LANL are LAN-structured, capturing internal and boundary traffic typical of enterprise, academic, and government networks. On the other hand, cloud-based telemetry (e.g., network flows, access logs, security events from AWS, Azure, Google Cloud) is often proprietary, access-restricted, and often lacks standardized public benchmarks, limiting dataset comparability. LAN environments also provide a well-defined and stable graph topology that is consistent for analyzing spatial and temporal relationships in network traffic, which is essential when evaluating GNNs and Transformer-based architectures. Restricting the scope to LAN-oriented systems, therefore, facilitates comparability of performance metrics and threat taxonomies across analyzed studies, enabling a methodologically consistent synthesis while reducing heterogeneity.

From a conceptual standpoint, LAN-based detection represents the core operational layer of organizational cybersecurity, where spatial (topological) and temporal (traffic-evolution) anomalies often first manifest or, conversely, are first observed as propagated effects from compromised cloud or IoT environments. While cloud infrastructures also exhibit internal multi-stage threat propagations progressing through reconnaissance, privilege escalation, lateral movement, and exfiltration stages similar to those mapped in ATT&CK or kill chain frameworks, the LAN remains the most empirically accessible and structurally consistent domain for evaluating detection methodologies. This highlights the LAN's role as both an origin and convergence point of threats within hybrid networking environments (e.g., LAN-Cloud hybrid infrastructures). Because this review investigates how GNNs, Transformers, and Reinforcement Learning capture spatial, temporal, and adaptive properties in network detectable threats, therefore, a LAN-based context offers a consistent substrate for evaluating these models under comparable and interpretable conditions.

The exclusion of cloud, IoT, and other network environments therefore represents a deliberate methodological boundary rather than a limitation of perspective. These architectures operate differently from traditional LANs: cloud systems use multi-tenant and dynamically managed resources, IoT networks often have limited and intermittent connectivity, and SDN separates control and data planes. These differences, including the distinct attack vectors each environment faces, make them not directly comparable with LAN-based threat detection. Thus, including all different networking contexts in the same framework would compromise methodological consistency and interpretability of the results. Moreover, there are still a number of organizations, particularly small to medium-sized enterprises, which continue to rely on LAN-based infrastructures without cloud

integration for reasons such as privacy concerns, making LAN-focused threat detection studies directly relevant to such organizations.

Nevertheless, the insights derived from this LAN-focused synthesis remain highly transferable to broader contexts. Many of the mechanisms identified, such as graph-based spatial reasoning, temporal attention, and adaptive reinforcement strategies, can inform future applications in cloud, IoT, mobile, and other network environments.

3.2.2. Threat Context

This review focuses on synthesizing studies that evaluate network-detectable threats, either through network traffic or host behavior. Therefore, non-network detectable threats operating solely at the Open System Interconnection (OSI) model's physical layer, such as jamming and electromagnetic interference, were excluded from the review.

Furthermore, this review exclusively focuses on the detection of network threats and their associated behavior, not the direct detection of network attacks. Threats are generally recognized as the potential to harm. Therefore, network threats can be defined as malicious or unauthorized activities, entities, or behaviors that indicate the possibility of a future attack directed at a network and its resources, where actual harm or disruption is realized when exploited by the threat actors. Therefore, the motivation behind exclusively focusing on network threats is the importance of threat mitigation before actual damage can be inflicted, facilitated by the proactive detection of such threat entities, their activities, and the early stages of multi-stage attacks, such as APT attack campaigns.

However, the terms threat and attack are used interchangeably in cybersecurity literature, which makes it extremely difficult to conduct an in-depth SLR in the strict sense of network threat detection. However, the SLR overcomes this grey area of challenge by drawing a clear boundary between these two conceptually different yet often intermingled terms.

3.3. Threat Frameworks and Taxonomies

The MITRE ATT&CK [48] and the cyber kill chain developed by Lockheed Martin [49] provide a technical framework for threat modeling and detection at various stages, backed up by a classification of adversarial tactics and techniques. While these two frameworks provide well-established taxonomies of tactics and attack phases, they do not formally distinguish threats from attacks, nor do they classify threats into structured groups. To address this gap, this review proposes a novel threat classification framework that systematically categorizes network threats, maps them to MITRE ATT&CK tactics and corresponding Cyber Kill Chain stages, and distinguishes them from subsequent attacks. The framework is unique in its methodological, rule-based approach and its focus on early stage, network detectable adversarial behaviors. The classification and categorization of threats evaluated in the reviewed studies, according to this framework, are presented in Section 4.

3.4. Methodological Framework for Threat Classification

To operationalize the proposed classification framework, each threat type reported in the reviewed studies was systematically mapped to the Enterprise MITRE ATT&CK tactic categories and their corresponding Cyber Kill Chain stages. This dual mapping provides both a technical dimension (what the threat or attack is) and a chronological dimension (where it sits within the lifecycle of an adversarial campaign). The classification frameworks is explained below.

3.4.1. Conceptual Framework

By combining MITRE ATT&CK and the Cyber Kill Chain, the framework establishes a hybrid taxonomy that captures both technical detail and chronological sequence:

- MITRE ATT&CK offers fine-grained categorization of adversarial tactics.
- Cyber Kill Chain provides temporal positioning within the attack lifecycle.

Neither framework alone achieves this duality. MITRE lacks chronological context, whereas the Kill Chain lacks technical granularity. Their integration therefore delivers a unified, interpretable structure capable of explaining what a behavior represents when it occurs, and crucially, whether it has transitioned from potential to realized impact. This interpretive dimension is essential for early-stage network threat detection.

Then each reported threat or attack label (e.g., PortScan, Infiltration, SQL Injection, Botnets) was aligned with the most appropriate ATT&CK tactic and Kill Chain stage according to its primary function and observable behavior. When a threat overlapped multiple tactics, the most representative or earliest stage tactic was chosen to maintain consistency with the proactive detection focus of this review. For example, SQL Injection was classified under Initial Access since it represents an entry vector, even though it may later contribute to lateral movement. Ambiguous or broad terms such as "malicious traffic" were refined by consulting the dataset documentation and experimental context.

To provide a clear operational reference within the proposed classification framework, Table 3 summarizes the 14 Enterprise MITRE ATT&CK tactics alongside their network detectable behavior and corresponding Cyber Kill Chain stages. The "Network Detectable Behavior" column contextualizes each ATT&CK tactic in network observable behaviors and indicators, such as unauthorized access, malware implantation, or port scanning, so that the technical definitions are grounded in observable events within network environments.

Table 3. Mapping Enterprise MITRE ATT&CK tactics to Cyber Kill Chain stages with their network detectable behaviors.

MITRE ATT&CK Tactic ID	MITRE ATT&CK Tactic Name	Network Detectable Behavior	Cyber Kill Chain Stage (Depending on Adversarial Technique)
TA0043	Reconnaissance	Gathering information for network attack planning.	Reconnaissance
TA0042	Resource Development	Setting up network attack infrastructure.	Weaponization/Reconnaissance
TA0001	Initial Access	Initial access to the network.	Delivery/Exploitation
TA0002	Execution	A phase where adversaries execute malicious code on a local or remote network to enable future attack actions.	Exploitation/Installation
TA0003	Persistence	Threat actors' ability to maintain hidden access within the network.	Persistence/Installation
TA0004	Privilege Escalation	Attempts to gain unauthorized higher access.	Exploitation/Installation
TA0005	Defense Evasion	Obfuscating threat presence to avoid detection by network scanners.	Actions on Objectives/Installation
TA0006	Credential Access	Attempting to obtain network credentials to internal network systems/resources.	Exploitation/Installation
TA0007	Discovery	Probing internal network resources and services.	Reconnaissance/Exploitation

Table 3. *Cont.*

MITRE ATT&CK Tactic ID	MITRE ATT&CK Tactic Name	Network Detectable Behavior	Cyber Kill Chain Stage (Depending on Adversarial Technique)
TA0008	Lateral Movement	Unauthorized moving between systems within the network using stolen credentials, exploiting vulnerabilities, etc.	Lateral Movement
TA0009	Collection	Data aggregation and staging on compromised hosts before exfiltration, limited direct network visibility, but may exhibit detectable staging patterns.	Exfiltration/Actions on Objectives
TA0011	Command and Control	Communicating with compromised hosts using beaconing or covert communications.	Command & Control
TA0010	Exfiltration	Unauthorized network data access or attempts at data movement.	Exfiltration/Actions on Objectives
TA0040	Impact	Post-exfiltration or disruption activities such as service shutdown, data destruction, or ransomware deployment. Beyond the proactive detection scope, but included for completeness.	Impact/Actions on Objectives

3.4.2. Context-Aware Definition of Threats and Attacks

This review distinguishes between threats and attacks not as fixed categories but as different operational states within the adversarial lifecycle:

- Threats denote capabilities, behaviors, or preparatory actions that have the potential to cause harm but have not yet materialized into a successful exploitation or system compromise, e.g., malware download, infiltration attempts, reconnaissance, or credential-testing activities.
- Attacks represent realized exploitation or harm, where the threat has transitioned into active execution against a target, achieving adversarial objectives or producing measurable system impact, e.g., DDoS flooding.

Some actions (e.g., SQL Injection, Privilege Escalation, Malware Download) may act as both independent attacks and precursors within larger campaigns such as APTs. Hence, the framework treats threat and attack as context-dependent classifications determined by realization rather than by label alone.

3.4.3. Threat/Attack Determination Rules

Table 4 presents the rationale for distinguishing between four categories: “Threat”, “Attack”, “Dual-context”, and “Default to Threat”.

This ensures that cases such as fileless malware remain threats (since no harm yet occurs in the relevant context), whereas post-exfiltrations are out of scope (as damage has already been done). In multi-stage APTs, early stages (Reconnaissance → Initial Access → Persistence) are classified as threat phases, while later stages (Exfiltration to Impact) are generally considered as attack phases. Moreover, as a single APT campaign encompasses both threats and attacks, its default classification as a whole is therefore a “Threat”, validating the consistency of the threat classification framework.

Table 4. Threat/Attack determination rationale.

Condition	Classification	Rationale
Behavior indicates intent, capability, or preparatory action without direct harm.	Threat	Potential risk: preparatory or reconnaissance stage.
Behavior achieves compromise, disruption, or data loss.	Attack	Realized exploitation or system impact.
Behavior may serve as both a standalone attack and a precursor within a multi-stage campaign.	Dual-context (Threat ↔ Attack)	Treated as a threat in proactive detection, attack when harm is evident.
The dataset or study does not specify the outcome (success unclear)	Default to Threat	Aligns with a proactive early warning perspective.

3.4.4. General Rules for Mapping (From Network Activity/Label → Classification)

- Treat attempted actions as Threats, and realized or successful actions as Attacks.
- Choose the earliest relevant stage when multiple stages apply.
- When the dataset or paper does not clarify success, default to Threat to align with the proactive detection perspective.
- Document all ambiguous cases in an ambiguity log for reviewer consensus.

3.4.5. Step by Step Procedure for Manual Classification

1. Extract labels: List every threat/attack label reported in the paper or dataset.
2. Normalize terminology (optional): If necessary, convert labels to lowercase, remove punctuation, and harmonize synonyms (e.g., port scan → portscan) for consistency.
3. Identify observed behavior: Examine dataset documentation or study description to infer practical meaning.
4. Apply mapping rules. Assign:
    - MITRE ATT&CK Tactic and ID
    - Cyber Kill Chain Stage
    - Classification (Threat/Attack/Dual-context)
5. Resolve ambiguity. If the label fits multiple categories, select the earliest stage and record justification.
6. Reviewer verification (Optional). A second reviewer reclassifies a 20–30% random subset. Compute inter-rater reliability (Cohen's κ). Resolve disagreements through discussion.
7. Record results. Maintain a spreadsheet with fields (Example fields are given below): Study ID | Dataset | Original Label | Mapped Tactic | Kill Chain Stage | Classification | Ambiguity Flag | Justification | Reviewer 1 | Reviewer 2 | Final Decision.

3.5. Research Design

3.5.1. Methodological Framework

This research adopted a Systematic Literature Review framework, based on the concept that knowledge can be extracted from the existing literature using unbiased and transparent methods. The SLR was conducted according to the PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses) 2020 guidelines for systematic transparency and reproducibility. Thus, the SLR protocol was preregistered in the Open Science Framework (OSF) before the study screening process. Moreover, this SLR can be classified as secondary research that adopted a structured, comparative narrative synthesis approach, primarily qualitative in nature, supported by quantitative aggregative elements. The qualitative part of the SLR examined elements such as modeling preferences, integration methods, methodology, strengths, limitations, etc. Among the quantitative

elements are frequencies of model evaluation instances, datasets, tested threat types, and the reported model performance metrics such as accuracy, F1-score, AUC, etc. Due to the extreme heterogeneity of main model types, different model combinations and methods, various datasets, and performance metrics of choice, a meta-analysis-based SLR was not feasible. However, research methods employed in this SLR are typically ideal for uncovering methodological trends and gaps in the literature, common model integration patterns, and drawing valuable insights, such as why certain approaches succeed or fail.

3.5.2. Research Design Components

The research design components are elaborated in Table 5.

Table 5. Research design components of the SLR.

Component	Description
Research Paradigm/Philosophy	The philosophy behind this SLR is post-positivist as the study assumes that objective performance and usage knowledge about GNNS, Transformers and RL algorithms can be extracted from systematically evaluating existing empirical studies.
Research Approach	The conclusions were drawn from observed patterns and insights from existing literature rather than from deductive testing. Therefore, the approach was primarily inductive.
Research Methodology	An SLR following PRISMA 2020 guidelines, followed by an OSF protocol pre-registration.
Design Type/Purpose	This study described the usage of GNNs, Transformers and RL, hence descriptive, compared model performance with each other, therefore comparative, and identified trends and patterns, therefore exploratory. And finally, it assessed the strengths and weaknesses of the utilized models and methodologies, therefore evaluative.
Data Collection & Storing Method	Secondary data collection for this SLR was done using structured search queries in Scopus, IEEEXplore and ACM DL, followed by manual searches in Scopus and IEEEXplore and snowballing to collect peer-reviewed journal articles and conference papers published from the year 2017 to 2025. The extracted data was stored in two tables called “metrics” and “study” in a relational database. Study_ID denoted with the notation S1, S2. . .SX was shared between the two tables.
Data Types	This SLR employs Mixed data types as both quantitative elements (accuracy, F1 score, AUC, etc.) and qualitative elements (integration methods, limitations of the methodologies, etc.) were extracted.
Analysis Strategy	Descriptive statistics for recorded performance metrics (accuracy, F1 score, AUC, etc.) and light thematic synthesis for drawing qualitative insights (common model limitations, trends, etc.) were conducted as the primary analysis techniques. Furthermore, to apply quality appraisal, a risk of bias assessment was performed for each included study.
Ethical Considerations	Not applicable as the study used publicly available secondary data in a non-privacy-critical domain. However, ethical research conducts were ensured via OSF registration and proper citations.
Validity & Reliability/Trustworthiness	Ensured by using PRISMA 2020 guidelines, OSF pre-registration protocol, ROB checklist and consistent screening and extraction criteria.

3.5.3. SLR Protocol and Registration

This systematic literature review followed the PRISMA 2020 methodology/checklist [50] (Supplementary Materials). And the SLR protocol was registered in OSF at https://osf.io/zs5a9 (accessed on 22 October 2025) to ensure transparency, reproducibility, and reduce bias.

3.5.4. Eligibility Criteria

The collected studies consisted of peer-reviewed journal articles and conference papers. The inclusion and exclusion criteria for this research were defined to include only relevant, interpretable, recent, high-quality research.

Therefore, the chosen studies were limited to studies written in English and published since 2017. The starting year was selected to align with the studies done after the beginning of the Transformer era, as the Transformer architecture was first published in 2017 by the landmark paper called "Attention is all you need" [14]. Furthermore, only the papers reporting quantitative performance metrics were eligible for this review. None of the purely theoretical papers without any experiments were included. Furthermore, grey literature, such as reviews, editorials, white papers, etc., was excluded. And the papers not focused on computer network contexts or non-transparent studies were also excluded.

3.5.5. Critical Appraisal (Risk of Bias Assessment)

For quality assessment, a custom quality assessment rubric was constructed. This custom RoB checklist was inspired by [51], PRISMA 2020 [50], and the CASP principles [52], and was designed to satisfy the machine learning/AI-related SLR requirements. Each of the 36 sources included was assessed against the above RoB criteria and documented. The list can be found in the Appendix A.

3.5.6. Information Sources and Search Strategy

The comprehensive, structured search query, which was used to search for relevant research studies in the chosen scholarly databases, was inspired by the aforementioned scope and the exclusion/inclusion criteria. Thus, the peer-reviewed journal articles and conference papers were searched, their titles and abstracts were screened, and they were collected from three well-known scholarly databases, namely, Scopus, IEEE Explorer, and ACM DL.

Search query:

("Graph Neural Networks" OR gnn* OR graphsage* OR gcn* OR gat OR gin OR "R-GCN" OR mpnn* OR "Graph Convolutional Network" OR "Message Passing Neural Network" OR "Graph Isomorphism Network" OR "Relational Graph Convolutional Network" OR "Graph Attention Network" OR "Dynamic Graph Convolution Network" OR transformer* OR "attention mechanism" OR "self-attention" OR "Vanilla Transformer" OR "Vanilla Transformers" OR "Vision Transformer" OR "Vision Transformers" OR "Temporal Transformer" OR "Temporal Transformers" OR "Graph Transformer" OR "Deep Reinforcement Learning" OR "Reinforcement Learning" OR rl OR q-learning OR "Deep Q-Learning" OR sarsa* OR drl* OR dqn OR "Policy Gradient" OR ppo OR actor-critic OR a2c OR sac)

AND

("network threats" OR "network anomalies" OR "brute force" OR "network scanning" OR "protocol abuse" OR "malicious tunneling" OR "DNS tunneling" OR "traffic obfuscation" OR "reverse shell" OR botnet OR "port scanning" OR "insider threats" OR "insider threat" OR "advanced persistent threats" OR "zero day" OR "zero-day" OR "zero-day exploits" OR "data exfiltration" OR "privilege escalation" OR "host compromise" OR "lateral movement" OR "persistence" OR "credential access" OR "command and control" OR "c2 traffic" OR "defense evasion" OR "initial access" OR exploit OR reconnaissance OR "post-exploitation" OR beaconing OR "data staging" OR "encrypted C2 traffic" OR "dns spoofing" OR "ip spoofing" OR "tcp syn flood" OR "udp flood" OR ddos OR pivoting OR "exfiltration over DNS" OR "exfiltration over HTTPS" OR "malicious payload")

AND

("computer networks" OR "network threat detection" OR "network traffic" OR "network monitoring" OR "network layer" OR "network anomaly detection" OR "network intrusion" OR "network intrusion detection" OR "network flows" OR netflow OR "network communications" OR "flow-based" OR "network behavior" OR "IP address" OR "IP header" OR "routing anomaly" OR "packet header" OR tcp OR udp OR "connection patterns" OR "flow-based detection" OR ftp OR smtp OR "application layer protocol" OR "payload analysis" OR "MAC address" OR "Ethernet frame" OR VLAN OR "session management" OR "SSL/TLS handshake" OR TLS)

NOT

(5 g OR 4 g OR "network on chip" OR chip OR power OR grid OR "smart grid" OR "mobile networks" OR "internet of things" OR iot* OR cloud* OR "cloud security" OR endpoint* OR "radio access network" OR "physical layer attacks" OR "spectrum sensing" OR "physical layer security" OR rfid* OR jamming OR phishing OR malware OR "spam detection" OR "network attacks detection" OR "network attack detection" OR cryptocurrency OR blockchain OR "software vulnerability" OR patching OR "static code analysis" OR review OR survey OR slr OR "Literature Review" OR "Systematic Literature review" OR nlp OR vision OR "recommender systems" OR social OR human OR bio OR transportation OR health OR drug)

After running the search query and using filters (year range 2017–2025, Language—English, and only peer-reviewed journal articles and conference papers), 234 studies were found in Scopus, including 123 journal articles and 111 conference papers, where IEEExplorer yielded 159 results, including 126 conference papers and 33 Journal papers. ACM DL search resulted in 17 conference papers and only 2 journal articles. Altogether, there were 412 initial studies collected from these three scholarly databases.

3.5.7. Screening and Study Selection

The three-stage screening (titles only, titles + abstracts, and full text) was designed to efficiently narrow down a high volume of search results while maintaining scientific rigor. In-depth analysis of ambiguous cases was required to ensure the transparency of the review. Inclusion and exclusion criteria were clearly defined during the SLR protocol registration and strictly applied to ensure the transparency of the screening process. The final selection of paper citations from each of the databases was downloaded mostly in RIS (Some in EndNote and Bib text formats as required) format and were sent to EndNote for reference management.

Deduplication was performed for 412 chosen studies in EndNote, where 372 studies resulted in unique entries. However, within Rayyan, an additional 11 duplicates were found. Thus, a total of 361 studies were reviewed through title and abstract screening in Rayyan, resulting in 108 studies for full-text screening. Finally, 32 studies were qualified for the systematic review after the full-text screening. However, to ensure maximum coverage and to ensure that no key studies were excluded, supplementary model-specific manual searches were conducted in Scopus. This resulted in four more qualified studies based on the inclusion criteria, hence the final 36 studies. PRISMA flow Diagram related to the study selection process is elaborated in Figure 1.

3.5.8. Data Extraction and Quality Assessment

Data was extracted from the final 36 papers according to predefined data extraction templates. No interpretations were made during the extraction of qualitative and quantitative elements, which was conducted according to PRISMA guidelines and by strictly following the data extraction instructions.

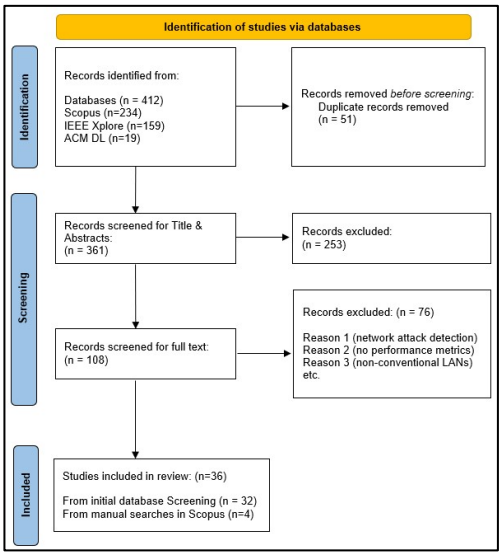

Figure 1. PRISMA flow diagram.

The following elements were chosen for the quantitative synthesis to identify trends, patterns, and to draw conclusions: Title, Study_Year, Model, Dataset, Threat_Type, Hyperparameter_Values, Accuracy_%, Precision_%, Recall_%, F1_Score_%, Micro_F1_Score, Macro_F1_Score, MCC, FPR%, FNR%, TPR%, TNR%, AUC, Specificity, FAR, Jaccard, Detection_Rate.

The following details were extracted from the chosen studies to conduct a qualitative analysis: model_integration_method, Methodology, Strengths, Limitations, Conclusions & Future_Works.

3.5.9. Limitations of the Chosen Research Method

The following summarizes the limitations identified in the chosen research method.

- The current body of research, including this SLR, has predominantly focused on LAN-based environments using datasets such as UNSW-NB15, CICIDS2017, CSE-CIC-IDS2018, and LANL. However, modern infrastructures increasingly rely on cloud computing, IoT ecosystems, and other hybrid architectures, limiting the broader applicability of LAN-based threat detection research.
- Inability to conduct a meta-analysis: The extreme heterogeneity caused by various performance metrics of choice, a wide range of different datasets, and network threat types made a meta-analysis impossible in this systematic literature review.
- Exclusion of non-peer-reviewed preprints: Non-peer-reviewed preprints were excluded as part of the gray literature during the searching and screening process. However, most of the GNN and Transformer literature is recent, especially in the cybersecurity domain. Thus, by excluding grey literature, there is a possibility of missing such high-quality yet non-peer-reviewed research from this SLR.
- Language bias: Due to resource constraints, only the journal articles and conference papers written in English were included in the review. This might have brought a regional/language-based bias to the review. For example, there are high-quality Chinese papers written in the scope of this SLR. However, it was not possible to analyze and synthesize such documents due to the shortage of Chinese-fluent reviewers.

3.5.10. Consideration of Alternative Methods

Alternative research methodologies such as narrative literature reviews, bibliometric analysis (methodology + methods), and meta-analysis (a strictly quantitative approach

to SLR) were initially considered and were later rejected due to the following reasons. Narrative reviews are qualitative, intended to provide a summary of the literature without a formal structure, and thus do not offer methods to analyze the extracted quantitative elements. Bibliometric analyses are more suited for mapping research trends than conducting methodological synthesis, hence they were rejected. A Meta-analysis SLR was not feasible due to the lack of standardization across studies in terms of their datasets, evaluated threats, and preferred performance metrics.

4. Results

4.1. Characteristics of Included Studies

The review included 36 studies published between the years 2017 and 2025. Among them, 23 studies are GNN related (including the GCN-RL hybrid), nine are Transformer related, and four are RL related, as shown in Figure 2. The included 36 studies comprise 23 journal articles and 13 conference papers, as shown in Figure 3. Regarding hybrid models combining GNNs, Transformers, and RL, there are two partial GNN-Transformer hybrids and one full GCN-RL hybrid (S21 [53]). Partial GNN-Transformer hybrids do not utilize full Transformer architectures in their hybrid designs. Among the partial hybrids, the first study (S28 [54]) employs a Transformer type multi-head attention and residual connections, and thus is considered neither a true GAT architecture nor a true GNN-Transformer hybrid. The other study (S29 [55]) employs a Transformer-XL style multi-head attention only, and yet again, it is not qualified as a true GAT nor a true GNN-Transformer. The number of GNN-based, Transformer-based, and RL-based studies per year in the review is shown in Figure 4. The popularity of GNN-based approaches is highlighted in Figure 5.

The studies included in this review demonstrated heterogeneity in terms of their evaluation methodologies, datasets, and threat types, making a systematic literature review a complex process. Most of the evaluated datasets contain multiple types of threat records, evaluated either individually or merged into a general attack/threat class. The review found that some studies have customized popular APT datasets to evaluate different stages of APT threats, whereas another set of datasets was prepared to train and test both known threats and unknown zero-day threats using techniques such as dataset shifting and masking. Most of the studies evaluated multiple datasets and various subsets to demonstrate the generalizability of their proposed models. As a result of the aforementioned strategies, a wide range of datasets and threat types has been identified by this review. Thus, there are 50 distinct datasets and 56 distinct threat types, including more advanced APTs, stealthy insiders, and previously unseen zero-day threats, that have been independently evaluated by the included studies.

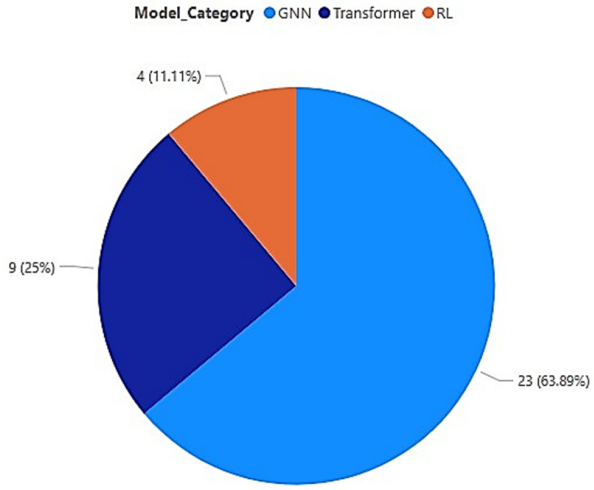

Figure 2. GNN, Transformer and RL model distribution in the review (%).

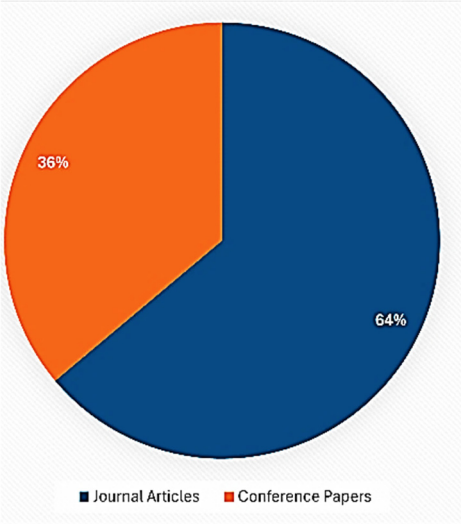

Figure 3. Distribution of journal articles and conference papers in the review (%).

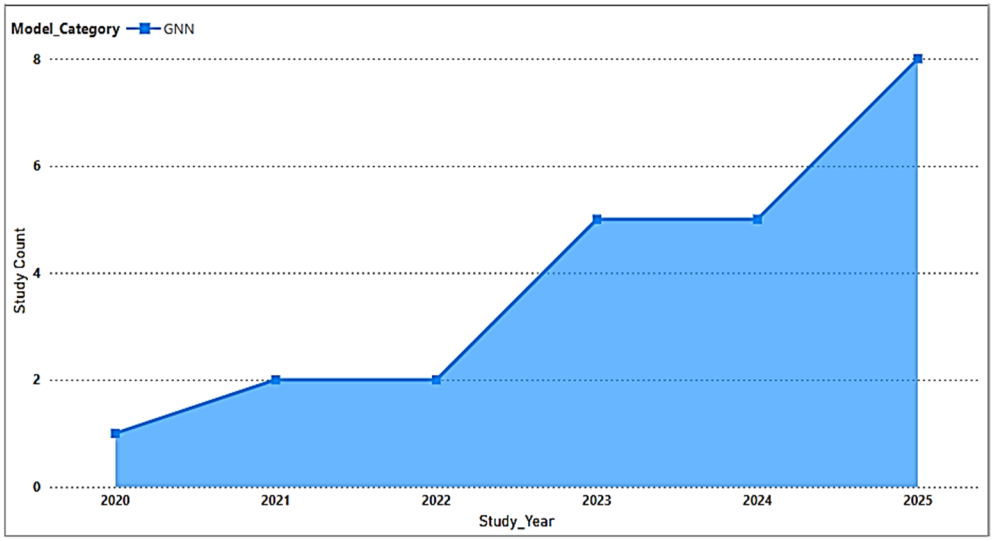

Figure 4. Number of GNN-based, Transformer-based, and RL-based studies per year in the review.

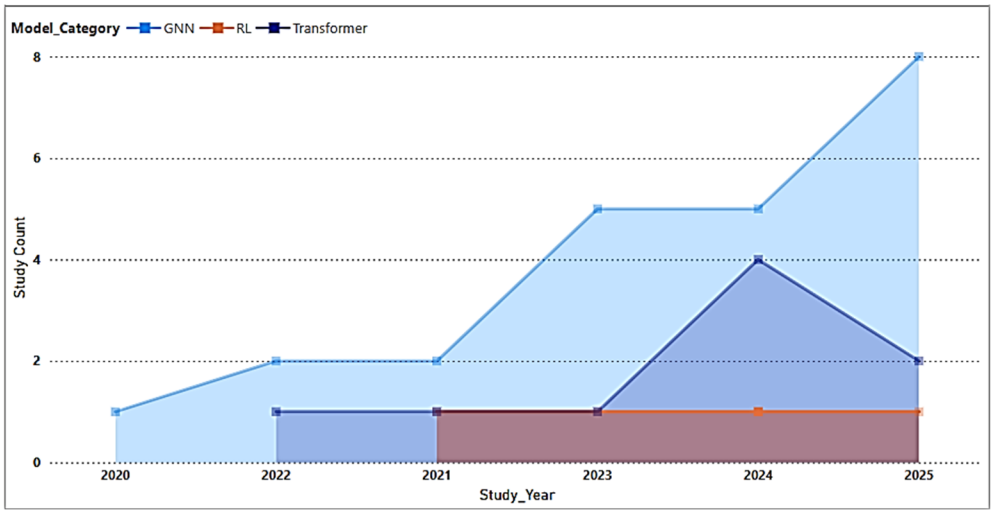

Figure 5. Rise of GNN-based studies in network threat detection.

The list of studies, including their study IDs, citations, Titles, and published years, is listed in Table 6.

Table 6. Studies included in this SLR.

Study_ID	Title	Year
S1 [56]	RAGN: Detecting unknown malicious network traffic using a robust adaptive graph neural network	2025
S2 [57]	A Novel Approach for APT Detection Based on Ensemble Learning Model	2025
S3 [29]	Detect Insider Threat with Associated Session Graph	2024
S4 [58]	AnoGLA: An efficient scheme to improve network anomaly detection	2022
S5 [59]	TBA-GNN: A Traffic Behavior Analysis Model with Graph Neural Networks for Malicious Traffic Detection	2025
S6 [60]	Euler: Detecting Network Lateral Movement via Scalable Temporal Link Prediction	2023
S7 [61]	Network Anomaly Detection Using a Graph Neural Network	2023
S8 [62]	Insider Threat Detection Using Generative Adversarial Graph Attention Networks	2022
S9 [63]	A Hierarchical Approach for Advanced Persistent Threat Detection with Attention-Based Graph Neural Networks	2021
S10 [64]	Deeply fused flow and topology features for botnet detection based on a pretrained GCN	2025
S11 [65]	Graph-Based Approaches to Detect Network Anomalies and to Predict Attack Spread	2023
S12 [26]	FN-GNN: A Novel Graph Embedding Approach for Enhancing Graph Neural Networks in Network Intrusion Detection Systems	2024
S13 [66]	Flash: A Comprehensive Approach to Intrusion Detection via Provenance Graph Representation Learning	2024
S14 [67]	Bot-DM: A Dual-Modal Botnet Detection Method Based on the Combination of Implicit Semantic Expression and Graphical Expression	2024
S15 [68]	An active learning framework using deep Q-network for zero day attack detection	2024
S16 [69]	Hybrid model for network anomaly detection with gradient boosting decision trees and tabtransformer	2021
S17 [70]	LGANet: Local Graph Attention Network for Peer-to-Peer Botnet Detection	2021
S18 [71]	Graph neural network approach with spatial structure to anomaly detection of network data	2025
S19 [72]	GraphMal: A Network Malicious Traffic Identification Method Based on Graph Neural Network	2023
S20 [73]	Multi-attributed heterogeneous graph convolutional network for bot detection	2020
S21 [53]	A Large-Scale P2P Botnet Detection Framework via Topology and Traffic Co-Verification	2024
S22 [74]	Graph Neural Networks with scattering transform for network anomaly detection	2025
S23 [75]	A Network Security Situation Element Extraction Method Based on Conditional Generative Adversarial Network and Transformer	2022
S24 [76]	AJSAGE: A intrusion detection scheme based on Jump-Knowledge Connection To GraphSAGE	2025
S25 [77]	Network intrusion detection system by learning jointly from tabular and text-based features	2023
S26 [78]	The Detection of Abnormal Behavior by Artificial Intelligence Algorithms Under Network Security	2024
S27 [79]	HyperTTC: Hypergraph-Empowered Tactic-Specific Traffic Clustering for Atomized APT Detection	2025
S28 [54]	Detecting abnormal logins by discovering anomalous links via graph transformers	2024
S29 [55]	Deep Temporal Graph Infomax for Imbalanced Insider Threat Detection	2023
S30 [80]	FlowTransformer: A Transformer Framework for Flow-Based Network Intrusion Detection Systems	2024
S31 [81]	AI-Driven Transformer Frameworks for Real-Time Anomaly Detection in Network Systems	2025
S32 [82]	Copula entropy regularization transformer with C^2 variational autoencoder and fine-tuned hybrid DL model for network intrusion detection	2025
S33 [83]	Network Intrusion Detection System Using Reinforcement Learning Techniques	2023
S34 [84]	Asynchronous Advantage Actor-Critic (A3C) Learning for Cognitive Network Security	2021
S35 [85]	Dueling Deep Q-Learning for Intrusion Detection	2025
S36 [36]	A novel multi-scale network intrusion detection model with transformer	2024

4.2. Threats Evaluated

In this review, a threat evaluation instance is defined as the assessment of a single threat by a given model or approach within a dataset and experimental setting. Multiple evaluations of the same model within a study (e.g., across datasets, threat classes, or different experimental configurations) are counted as separate threat evaluation instances.

Moreover, in several studies, the authors evaluated their proposed models alongside a range of baselines (e.g., RNN, CNN, or traditional machine learning classifiers) or under multiple experimental configurations. To maintain consistency and focus, only the performance metrics corresponding to the proposed model and in its best-reported configuration were reported. Baseline results and auxiliary experiments, such as ablation studies or evaluations on unrelated datasets, were not included in the performance tables, as they fall outside the scope of this review. This approach ensured comparability across studies while highlighting the threat evaluations most relevant to the research objectives.

Thus, among the 56 distinct threat types covered, the most frequently evaluated are the Botnets in 30 instances (General Botnets in 23 instances, P2P Botnets in 4, C2 Botnets in 2 and kelihos Botnet in 1), Backdoor & Reconnaissance (22 instances), Analysis & Fuzzers (21 instances), Shellcode (19 instances), Infiltration (17 instances) and Portscan, Probing, R2L and U2R (13 instances). Twenty-six distinct threat types are evaluated a single time, either alone or in a binary or multiclass classification. Some of these distinct threat types fall into broader, multistage threat categories, such as APT, and are clustered accordingly.

4.3. Consistency of the Threat Classification Framework in Practical Applications

The threat classification framework achieves consistency by separating the analytical dimensions of what a behavior represents (technical function) and when it occurs (chronological context). By combining MITRE ATT&CK and Cyber Kill Chain models, it provides both technical precision and temporal alignment, enabling coherent classification even when different studies use overlapping terminology. For example, activities such as port scanning, fuzzing, and credential testing are consistently categorized under Reconnaissance/Threats, whereas later stages, especially within multi-stage APTs such as exfiltration attempts or lateral movement are represented in the framework as Dual-context or Threat, depending on observable network impact.

Evidence from this SLR supports the framework's internal consistency in practical use. Across the 36 reviewed studies, 56 distinct threat types were successfully mapped to their corresponding ATT&CK tactics and Cyber Kill Chain stages without logical conflict, as shown in Table 7. The vast majority of threat labels were unambiguously classifiable under the established rules, and only a small subset required adjudication (e.g., Impersonation or Zero-Day, which span multiple stages or presented special cases). In all such cases, the framework's rule to select the earliest representative stage produced agreement between reviewers and consistent results across datasets such as CICIDS2017, UNSW-NB15, and CSE-CIC-IDS2018. These empirical outcomes indicate that the framework is not only theoretically coherent but also operationally stable when applied to real network threat taxonomies and benchmark datasets.

Zero-Day represents previously unknown or novel exploitations that have not been observed before. Within the proposed framework, it is mapped to Initial Access (TA0001) and the Exploitation stage of the Cyber Kill Chain, reflecting its earliest, potential role as an entry point in network-based adversarial campaigns. Thus, these events are treated as threats consistent with the framework's proactive detection orientation and earliest stage mapping principle. This ensures methodological coherence across all 56 threat types.

Table 7. Mapping of threats covered in the review according to the network threat classification framework.

Threat Type (Grouped)	MITRE ATT&CK Tactic ID(s)	MITRE ATT&CK Tactic Name(s)	Cyber Kill Chain Stage	Classification
Reconnaissance Group				
Reconnaissance	TA0043	Reconnaissance	Reconnaissance	Threat
Analysis	TA0043	Reconnaissance	Reconnaissance	Threat
Fuzzers	TA0043	Reconnaissance	Reconnaissance	Threat
Portscan	TA0043	Reconnaissance	Reconnaissance	Threat
Probing	TA0043	Reconnaissance	Reconnaissance	Threat
Probing-IP sweep	TA0043	Reconnaissance	Reconnaissance	Threat
Probing-Nmap	TA0043	Reconnaissance	Reconnaissance	Threat
Probing-Port sweep	TA0043	Reconnaissance	Reconnaissance	Threat
Darknet Activity	TA0043	Reconnaissance	Reconnaissance	Default to Threat
Initial Access Group				
Initial Access	TA0001	Initial Access	Delivery/Exploitation	Dual-context
Initial Compromise	TA0001	Initial Access	Delivery/Exploitation	Dual-context
Infiltration	TA0001	Initial Access	Delivery/Exploitation	Threat
Shellcode	TA0001	Initial Access	Exploitation	Dual-context
R2L	TA0001	Initial Access	Delivery/Exploitation	Dual-context
SQL Injection	TA0001	Initial Access	Exploitation	Dual-context
Unauthorized access	TA0001	Initial Access	Delivery/Exploitation	Dual-context
Zero-Day	TA0001	Initial Access	Exploitation	Threat
Persistence Group				
Persistence	TA0003	Persistence	Installation	Dual-context
Backdoor	TA0003	Persistence	Installation	Dual-context
Privilege Escalation Group				
Privilege Escalation	TA0004	Privilege Escalation	Installation/Exploitation	Dual-context
Process Injection	TA0004	Privilege Escalation	Exploitation	Dual-context
Defense Evasion Group				
Defense Evasion	TA0005	Defense Evasion	Installation/Exploitation	Dual-context
Fileless Malware	TA0005	Defense Evasion	Installation/Exploitation	Default to Threat
Credential Access Group				
Credential Access	TA0006	Credential Access	Credential Access	Dual-context
Bruteforce	TA0006	Credential Access	Credential Access	Threat
FTP-Patator	TA0006	Credential Access	Credential Access	Threat
FTP-Bruteforce	TA0006	Credential Access	Credential Access	Threat
SSH-Patator	TA0006	Credential Access	Credential Access	Threat
SSH-Bruteforce	TA0006	Credential Access	Credential Access	Threat
HTTP-Bruteforce	TA0006	Credential Access	Credential Access	Threat
Unauthorized login attempts	TA0006	Credential Access	Credential Access	Threat

Table 7. *Cont.*

Threat Type (Grouped)	MITRE ATT&CK Tactic ID(s)	MITRE ATT&CK Tactic Name(s)	Cyber Kill Chain Stage	Classification
Discovery Group				
Anomalous TCP Connections	TA0007	Discovery	Reconnaissance/Discovery	Default to Threat
Lateral Movement Group				
Lateral Movement	TA0008	Lateral Movement	Lateral Movement	Dual-context
Lateral Movement via Anomalous Authentications	TA0008	Lateral Movement	Lateral Movement	Dual-context
Command and Control Group				
Botnets	TA0011	Command and Control	Command and Control	Threat
P2P Botnets	TA0011	Command and Control	Command and Control	Threat
Kelihos botnet (P2P Botnets)	TA0011	Command and Control	Command and Control	Threat
C2 Botnets	TA0011	Command and Control	Command and Control	Threat
C2 Traffic	TA0011	Command and Control	Command and Control	Threat
Others				
Impersonation	TA0005, TA0006	Defense Evasion, Credential Access	Reconnaissance/Delivery/ Credential Access	Dual-context
Data Exfiltration	TA0010	Exfiltration	Exfiltration	Default to Threat
Multi-stage/APT-style threats				
APT	TA0001, TA0008	Initial Access, Lateral Movement	Reconnaissance/Delivery	Threat
APT related login activities	TA0006, TA0008	Credential Access, Lateral Movement	Credential Access/Lateral Movement	Threat
APT (Initial access, Privilege escalation, Persistence, Stealth operations, Data theft)	TA0001, TA0004, TA0003, TA0005, TA0010	Initial Access, Privilege Escalation, Persistence, Defense Evasion, Exfiltration	Reconnaissance → Exploitation → Installation	Threat
APT (Lateral Movement, C2 Traffic)	TA0008, TA0011	Lateral Movement, Command and Control	Lateral Movement → Command & Control	Threat
APT (Probing, R2L, U2R)	TA0043, TA0001, TA0004	Reconnaissance, Initial Access, Privilege Escalation	Reconnaissance/Delivery	Threat
APT (Process creation & manipulation, File access and modification, Inter-process communication (IPC) hijacking, Network connection hijacking)	TA0004, TA0010, TA0005, TA0008	Privilege Escalation, Exfiltration, Defense Evasion, Lateral Movement	Installation/Exploitation/Lateral Move	Threat
APT (Rootkit installation, Process injection, Kernel-level malware usage)	TA0003, TA0004	Persistence, Privilege Escalation	Installation/Exploitation	Threat
APT (Unauthorized access, Malware implantation, Data tampering, Misuse of system resources)	TA0001, TA0003, TA0010, TA0005	Initial Access, Persistence, Exfiltration, Defense Evasion	Delivery/Installation/Exploitation	Threat

Table 7. *Cont.*

Threat Type (Grouped)	MITRE ATT&CK Tactic ID(s)	MITRE ATT&CK Tactic Name(s)	Cyber Kill Chain Stage	Classification
APT(Malware download, Malicious payload execution, Remote code execution, System command injection)	TA0001, TA0005	Initial Access, Defense Evasion	Delivery/Exploitation	Threat
Multi-stage APT with advanced evasion techniques	TA0001, TA0005, TA0008	Initial Access, Defense Evasion, Lateral Movement	Delivery/Installation/Lateral Movement	Threat
Multi-stage APT with attack paths	TA0001, TA0008	Initial Access, Lateral Movement	Delivery/Lateral Movement	Threat
Insider Threats				
Insider Threats	TA0006, TA0008, TA0010	Credential Access, Lateral Movement, Exfiltration	Credential Access/Lateral/Exfiltration	Threat
Insider Threats (AB-I) (data exfiltration)	TA0010	Exfiltration	Exfiltration	Default to Threat
Insider Threats (AB-II) (credential theft)	TA0006	Credential Access	Credential Access	Dual-context
Insider Threats (AB-III) (malicious tunneling)	TA0011	Command and Control	Command & Control	Threat

The framework's practical reliability stems from its deterministic classification rules and dual-context mapping. When faced with ambiguous or compound labels, the framework prescribes selecting the earliest relevant stage in the attack lifecycle to preserve its proactive detection orientation. Thus, even if a study's terminology implies an "attack," if the behavior corresponds to an early-stage reconnaissance, initial access, or privilege escalation attempt, it is consistently treated as a Threat or Dual-context event. This ensures that early-warning behaviors, those most relevant for pre-attack detection are not conflated with realized attacks, a distinction often blurred in prior literature.

Because it aligns with two widely adopted taxonomies (MITRE ATT&CK and the Cyber Kill Chain), the framework can be replicated or extended in other analytical contexts. Its steps canonical label normalization, tactic stage mapping, earliest stage selection, and classification into "Threat"/"Dual-context"/"Default to Threat" are transparent, making it straightforward for other researchers or analysts to follow without subjective interpretation. Furthermore, its distinction between potential threats, preparatory actions, and late-stage dual-context events remained conceptually stable across all 36 reviewed studies, even though the source terminology varied.

Nonetheless, the framework's practical application still depends on the level of detail available in primary datasets. Several public benchmarks lack explicit indication of whether behaviors represent attempted or successful compromises. In these cases, the framework's rule to default to Threat maintains methodological coherence while transparently acknowledging uncertainty. Future dataset design should ideally record this distinction to improve transparency and strengthen the reproducibility of network threat related research.

Therefore, this evidence demonstrates that the proposed methodological classification framework can achieve high practical consistency. By offering deterministic classification rules, a dual technical–chronological structure, and clear mechanisms for resolving terminological ambiguity, it enables consistent interpretation of cyber threat activities across heterogeneous sources.

4.4. GNN-Based Studies

Nineteen distinct GNN variants/approaches have covered 139 threat detection instances (note that GCN + RL approach is shared by both GNN and RL studies, though it is being counted here) of threat detection across 48 distinct threat types, as elaborated in Figure 6 and Table 8. Among these GNNs, there are well-known variants such as GCN, GAT, GraphSAGE, etc., whereas other GNN approaches, such as Graph Transformer, GCN + GraphSAGE or GCN + RL hybrid, are also categorized into their distinct categories.

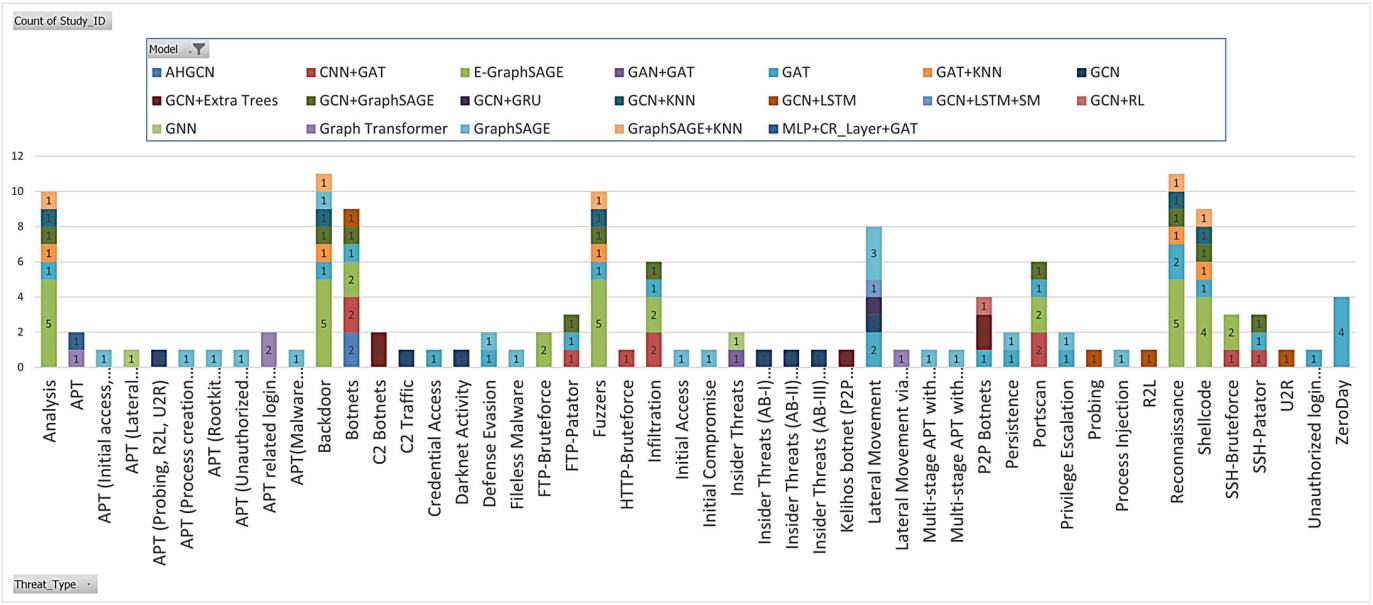

Figure 6. Distribution of threat evaluation instances across GNN models and their hybrid approaches (See Table 7 for full threat names).

Table 8. Frequency of GNN models and their hybrid approaches across studies.

Model/Hybrid Approach	Unique Study Count
GAT	4
GCN	4
GraphSAGE	3
E-GraphSAGE	2
GNN	2
AHGCN	1
CNN + GAT	1
GAN + GAT	1
GAT + KNN	1
GCN + Extra Trees	1
GCN + GraphSAGE	1
GCN + GRU	1
GCN + KNN	1
GCN + LSTM	1
GCN + LSTM + SM	1
GCN + RL	1
GraphSAGE + KNN	1
MLP + CR_Layer + GAT	1
Graph Transformer	1

Among the distinct threat types detected by GNNs, the Backdoor and Reconnaissance are the most frequently evaluated (11 instances each), followed by Fuzzers and Analysis

(10 instances each), Botnets (9 instances), Shellcode (9 instances), and Lateral Movement (8 instances). There are 26 distinct threat types evaluated a single time, including Credential Access (by GAT), Initial Compromise (by GraphSAGE), HTTP-Bruteforce (by CNN + GAT), etc. Among the 19 GNN models/hybrid approaches utilized across the studies, GAT and GCN are the most frequently utilized, by 4 studies each, followed by GraphSAGE, E-GraphSAGE and GNN by 2 studies each, as shown in Table 8.

4.5. Transformer-Based Studies

There are 13 distinct Transformer model architecture approaches employed by the included studies, where these models evaluated 22 distinct network threat types in a total number of 129 threat evaluation instances, as shown in Table 9 and Figure 7. Among these models, those that followed conventional Transformer architectures with customizations to their architectures were categorized as "Transformer", whereas those that used the original architecture introduced by [14] were categorized as "Vanilla Transformer". Among different Transformer architecture types and hybrid approaches, the "Transformer" category was employed in 20 threat evaluation instances, followed by "Transformer + MLP" in 15 instances. Among the distinct threat types evaluated by different Transformer architectures, Botnets were the most frequently evaluated, with 13 evaluation instances, followed by Infiltration, which was evaluated 11 times. Analysis, Backdoors, Fuzzers, Probing, R2L, Reconnaissance, Shellcode, and U2R were each evaluated in 9 instances, shown in Figure 7.

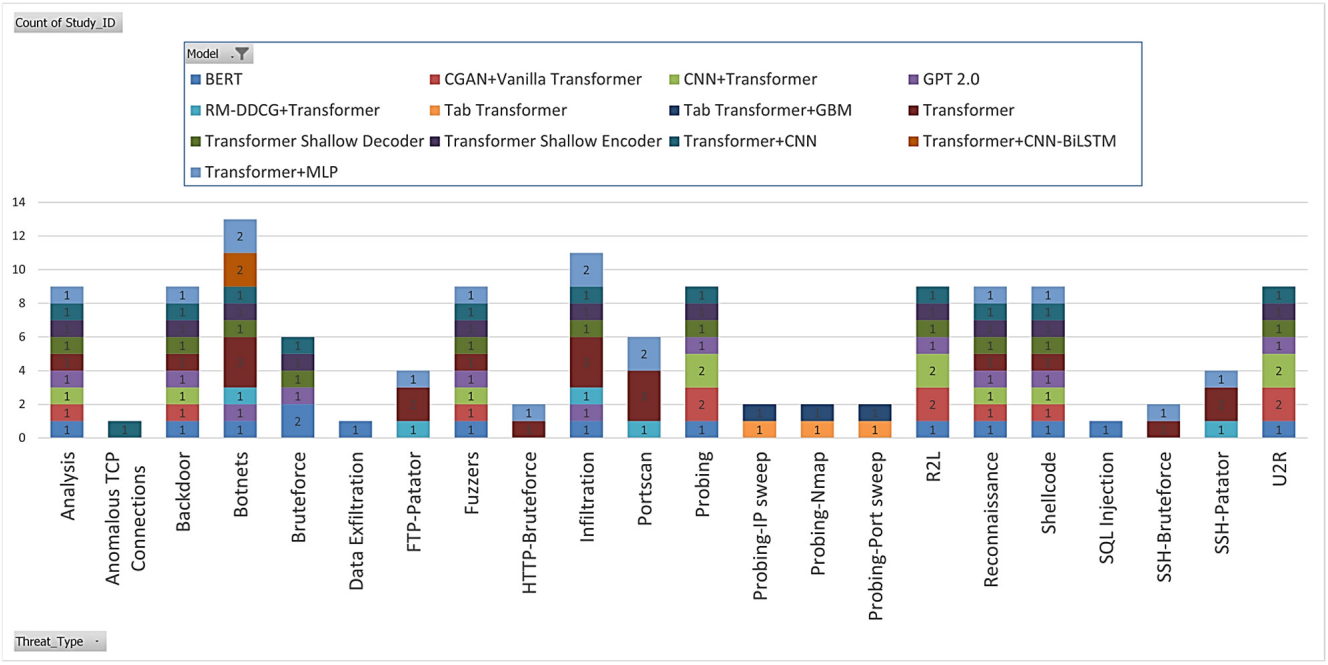

Figure 7. Distribution of threat evaluation instance counts across Transformer models and their hybrid approaches.

Transformer architectures and their hybrid approaches vary greatly across studies, where the Transformer and BERT architectures were each used by two studies and the rest of them by a single study, as shown in Table 9.

Table 9. Frequency of Transformer models and their hybrid approaches across studies.

Model/Hybrid Approach	Unique Study Count
Transformer	2
BERT	2
Transformer + MLP	1
Transformer + CNN	1
Transformer + CNN-BiLSTM	1
Tab Transformer	1
Tab Transformer + GBM	1
CGAN + Vanilla Transformer	1
RM-DDCG + Transformer	1
Transformer Shallow Encoder	1
Transformer Shallow Decoder	1
GPT 2.0	1
CNN + Transformer	1

4.6. RL-Based Studies

There are 5 distinct model types and approaches to RL-based network threat detection studies explored in this review, performing a total of 26 threat detection instances across 13 different network threat types, as shown in Figure 8. These threat types are Analysis, Backdoor, Botnets, Bruteforce, Fuzzers, impersonation, Portscan, Probing, R2L, Reconnaissance, Shellcode, U2R, and Zero-Day. The frequency of RL models and their hybrid approaches across the included studies is displayed in Table 10.

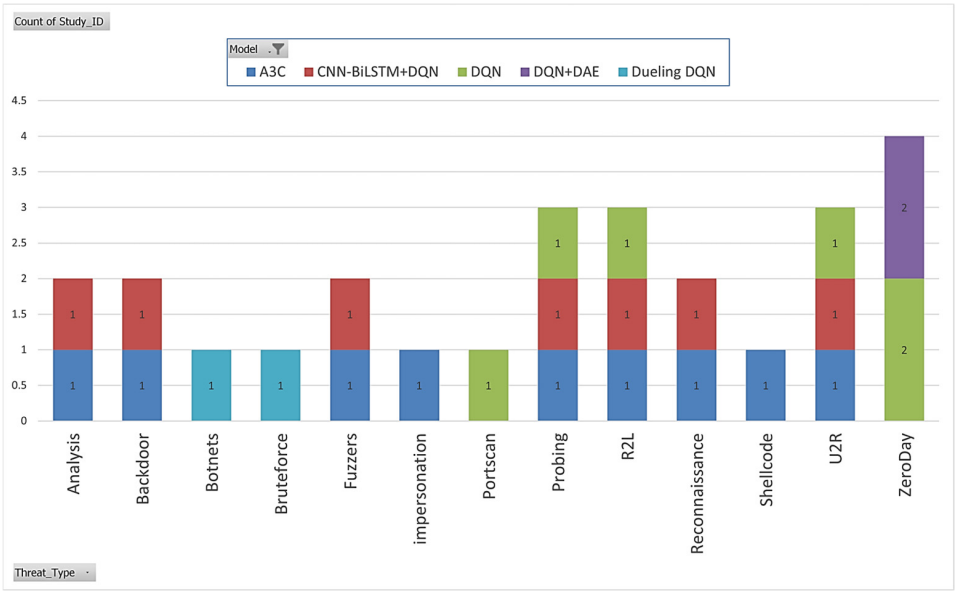

Figure 8. Distribution of threat evaluation instance counts across RL models and their hybrid approaches.

Table 10. Frequency of RL models and their hybrid approaches across studies.

Model/Hybrid Approach	Unique Study Count
CNN-BiLSTM + DQN	1
DQN	1
DQN + DAE	1
A3C	1
Dueling DQN	1

5. Analysis and Discussion

5.1. Model Performance Analysis

Among the included studies, some evaluated the network threats as a binary classification problem, labeling the merged class as "attacks", "anomalous", or similar. Few studies, including S22 [74] and S36 [36], conducted multi-class evaluations and thus recorded Macro F1 scores or class-wise performance metrics. A number of other studies evaluated standalone threat types, e.g., S4 [58], S6 [60], S10 [64], S13 [66], S15 [68], S19 [72], S20 [73], S23 [75], S35 [85]. Regarding the studies that merged threats into a single malicious class, their reported overall performance metrics were extracted and used in the analysis. This approach is unavoidable and well justified since merging class labels and binary classification is a fairly common practice across cybersecurity literature. Hence, in this SLR, only the threat-relevant labels (e.g., brute force, reconnaissance, botnet) of such instances were documented and synthesized. However, the reported metrics may sometimes reflect performance across a broader set of classes, which is acknowledged. The data extraction and synthesis approach for all such instances across traditional ML/DL models, GNNs, Transformers, and RL algorithms was the same, further justifying the performance comparisons across them.

5.2. Performance of GNNs

5.2.1. Overall Success Rate of GNNs

As evident in the analysis, GNNs are highly successful in detecting network threats. Their performance depends on how the graph is constructed to model the threat behavior, the overall model architecture, and the nature of the dataset (graph-native data vs. tabular and sequential data). The following is a breakdown of GNNs' overall performance and effectiveness.

This analysis categorizes GNN model performances into three categories based on performance metrics: Top performers ($\geq$90% F1-score/Accuracy/AUC & Other supporting Metrics), Moderate Performers (80–89% F1-score/Accuracy/AUC & Other supporting Metrics) and Underperformers (<80% F1-score/Accuracy/AUC & Other supporting Metrics). Note that Accuracy, Precision, and Recall are abbreviated as A, P, and R, respectively.

The percentage of top-performing GNNs is 74.2%, with studies recording high accuracies/F1 scores > 90%. The percentage of moderate performing GNNs is 16.1%, recording F1 scores between 79–85% and mixed metrics. The percentage of underperforming GNNs is 9.7%, displaying the high applicability and the potential of GNNs in the network threat detection domain.

5.2.2. GNN Top Performers

According to the reported metrics (accuracy, precision, recall and F1-score) in the included studies, the highest performing GNN-based studies demonstrate that they excel at network threat detection when provided and modeled with graph-structured network data. This is evident by the top performing studies such as S5 [59], S12 [26], S21 [53] and S1 [56]. S5 [59] achieves near-perfect performance: 99.97% accuracy, 99.96% F1-score and 99.97% recall detecting botnets and brute force threats in CICIDS2017. S5 achieves this by utilizing its GAT architecture. GAT converts raw network packets into Traffic Interaction Graphs (TIGs), where the edges of the graphs encode meaningful information like protocol-level interactions, and the attention mechanism calculates how much attention should be given to the critical edges, such as the ones presented with SSH spikes. The performance values demonstrate that learning graph-structured information directly from raw network packet data, rather than preprocessed statistics is vastly successful.

GCN + GraphSAGE utilized in S12 [26] reaches 99.76% accuracy, 99.26% precision, 99.51% recall and 99.76% F1-score in the CICIDS2017 dataset and 98.65% accuracy, 98.01% precision, 98.97% recall and 98.65% F1-score in UNSW-NB15. The hybrid model constructs graph-structured data by mapping network flows into graph nodes and mapping shared IPs to graph edges. GraphSAGE leverages aggregation of features from neighboring nodes by only sampling relevant neighbors rather than processing the whole graph to capture distributed threat patterns like the ones presented with botnets, making it highly scalable and efficient for large networks.

S21 [53] combines GCNs with Policy Gradient RL to detect p2p botnets at 99.78% accuracy, evaluated on a custom dataset named "ISCX-Botnet + CAIDA + Synthetic P2P traffic". The study further reported 98.89% precision, 98.92% recall, and 98.91% F1-score. The proposed GCN maps hosts to its nodes and the communication topology of the network into edges. First, the GCN flags suspicious nodes and then the RL agent prioritizes high-risk nodes for deep packet inspection. According to the study, this method reduces false positives by 90% when compared to random sampling and demonstrates the strength of combining RL algorithms with the power of GNNs.

The GAT employed in study S1 [56] detects zero-day threats at 99.48% accuracy, 99.43% precision, 99.4% recall, and 99.42% F1-score. The proposed model is evaluated in four different datasets: CICIDS2018 + synthetic zero-day traffic, CTU-13 + synthetic zero-day traffic, UJS-IDS2022 + synthetic zero-day traffic and USTC-TFC2016 + synthetic zero-day traffic, and records its highest performance under the last-mentioned dataset. The proposed GAT constructs graph nodes using Hosts/IPs and the edges using traffic flows. The model dynamically refines the graph structure during the inference by iteratively updating its adjacency matrix using Laplacian regularization. This iterative updating helps take down the adversarial edges introduced by evasion attacks such as SPAN. Moreover, the native attention mechanism tied into the GAT allows it to focus on important and meaningful traffic flows while downweighing suspicious connections, hence its success and resilience towards zero-day threats combined with adversarial tactics.

S24 [76] detects insider threats by evaluating 6 different datasets/subsets: DARPA TC#3 Cadets, DARPA TC#3 Fivedirections, DARPA TC#3 Theia, DARPA TC#3 Trace, DARPA TC#5 Theia, DARPA TC#5 Trace and Unicorn SC-2 dataset. The paper records its highest performance by evaluating insider threats using the Unicorn SC-2 dataset, where the detection accuracy was 94%, 93% precision, 96% recall and 94% F1-score. The proposed model achieves this high performance by integrating GraphSAGE with attention mechanisms and Jumping Knowledge (JK) Networks. Their methodology involves constructing provenance graphs by mapping processes to nodes and dependencies to graph edges, and further using attention mechanisms to weigh the importance of critical node-edge relationships, such as those presented with unusual process forks. The JK layer helps preserve features and prevent information loss of the GNN across different graph depths during the aggregation process. According to the study, this specialized architecture is effective at APT detection, where capturing long-range attack and threat patterns matters, e.g., detecting privilege escalation chains while keeping false positives at a lower rate.

Study S28 [54] employs a unique Graph Transformer architecture, which can be considered as a partial GNN-Transformer hybrid. Unlike traditional GAT, it utilizes a Transformer-style multi-head attention to analyze authenticate traces over the entire network to identify anomalies such as credential stuffing and lateral movement. Moreover, the model utilizes residual connections to filter noise and preserve long-term contexts, thus maintaining a low false-positive rate of 6.8%. Consequently, the model captures long-range dependencies associated with APT-related login behavior and exposes lateral movement using multi-head attention, multi-hop sub graphs to expose lateral movement paths, and by

preserving long-term context via residual connections. And thus, the Graph Transformer detects APT-related logins in CERT r5.2 at a 93.52% F1 score.

5.2.3. GNN Underperformers

Despite the decent performance of the E-GraphSAGE model in S19 [72] on common threats like Fuzzers and Reconnaissance in the UNSW-NB15 dataset, it fails to detect rare attack types like Backdoor (F1 score of 13.35%) and Analysis (F1 score of 17.32%). The root cause of this failure, according to the authors, is the very few samples related to Backdoor and Analysis threats. Thus, the GraphSAGE aggregator fails to identify meaningful edges to identify local anomalies during the training. Another cause of failure is relying on static graphs, which cannot adapt to the evolving nature of dynamic threats.

The GCN's low performance (F1-score of 70.45%, low Jaccard of 57.99%) in identifying APT threats defined and evaluated in the UNSW-NB15 dataset in S27 [79] is due to the problematic transfer techniques and assumptions taken during its methodology. The study employs the same hypergraph framework that it used to benchmark citation networks in a network threat detection context in a similar approach. It maps hosts/IPs as nodes and communications among them as edges, and artificially groups "apt tactics" as hyperedges. However, the hypergraph construction does not take temporal dynamics into account and thus does not align with the nature of evolving network threats. Moreover, there are no edge features, such as protocol flags, traffic volume dynamics, and stage metadata, which are also ignored. Furthermore, the synthetic hyperedge creation using Adaptive Synthetic (ADASYN) sampling introduces bias as they do not reflect real APT tactics such as the ones defined in the MITRE attack kill-chains (e.g., infiltration—lateral movement—data exfiltration), which collectively cause the observed underperformance.

The poor performance of some of the evaluations done with GNNs in S6 detecting lateral movement is caused by their inherent architectural limitations. Vanilla GNNs are not natively designed to handle temporal sequences of data but to model and solve graph-structured problems. On the other hand, lateral movement involves temporal patterns across evolving host connections, which static graph representations fail to model. Data extracted from S6 [60] revealed that pure GAT without any temporal modeling components achieved a high TPR of 99.88% with a significantly high FPR of 23.174% which is above the tolerated threshold in cybersecurity. Furthermore, the precision of 0.01% is catastrophically low. Thus, vanilla/static GNNs' inherent inability to understand temporal patterns explains their effectiveness in identifying malicious nodes, while struggling to contextualize them in sequential attack progressions such as those presented with lateral movement. According to the paper, another prevailing cause of the underperformance is extreme class imbalance presented in the "LANL 2015 Comprehensive Multi-Source Cyber Security Events dataset" regarding lateral movement-related edges vs. benign edges. However, S6 demonstrates that this failure is sequentially overcome by integrating distinct architectural components. First, adding an LSTM network to a GCN provides the necessary temporal context, dramatically reducing the FPR. The ultimate solution, exemplified by the GCN + LSTM + SM model, incorporates a dedicated Softmax decoder to directly combat the class imbalance. This full hybrid achieves a precision of 98.6% with an FPR of 0.013% on the OpTC dataset, proving that the complete integration of spatial (GCN), temporal (LSTM), and imbalance-aware (SM) components is required for high-fidelity lateral movement detection. Moreover, the following GNN studies that detect lateral movement successfully can further provide valuable insights into the failure of vanilla GNN only S6 [60] evaluations. S9 [63] introduces a metapath-aggregated GNN in its Intrahost Provenance Graph/IPG to tackle APT kill-chains (e.g., infiltration $\rightarrow$ lateral movement $\rightarrow$ exfiltration) and achieves top performance on the same LANL dataset. GraphSAGE in

S13 combines unified provenance graphs with Word2Vec temporal encoding to preserve attack progression presented with lateral movement. GAN in S28 [54] uses Transformers like global attention to link lateral movement via anomalous authentications across time, and updates edge relationships in real-time for authentication graphs. GAT in S18 [71] uses dynamic edge weight optimizations to capture evolving threats, including lateral movement. Thus, the above top performers confirm that GNNs are indeed powerful at detecting time-aware temporal threats when coupled with the right temporal-aware techniques. Top performers, moderate performers and underperformers of GNN-related studies are displayed in Tables 11–13, respectively.

5.3. Performance of Transformers

5.3.1. Overall Success Rate of Transformers

This analysis categorizes Transformer model performances into three categories based on recorded performance metrics: 1. Top performers (≥90% F1-score/Accuracy/AUC & Other supporting Metrics), 2. Moderate Performers (80–89% F1-score/Accuracy/AUC & Other supporting Metrics) and 3. Underperformers (<80% F1-score/Accuracy/AUC & Other supporting Metrics). Note that Accuracy, Precision, and Recall are abbreviated as A, P, and R, respectively. There are 34 distinct Transformer evaluation scenarios against various network threat datasets. Among those 43 evaluations, 32 scenarios or 74.41% can be categorized into the top performer category, 8 scenarios or 18.60% into the moderate performers, whereas only 6.97% fall into the underperformers.

5.3.2. Transformer Top Performers

All the 20 top-performing evaluation instances which reported accuracy values exceeded 90%, including 10 of them achieving accuracies above 99%. Transformers combined with other deep learning models such as MLP, GAN or CNN dominated among the top performers, particularly in Botnets, Portscans and Bruteforce threats. Transformer + MLP in S25 had exceptionally high accuracy of 99.98% and 99.8% F1 score with a 99.77% TPR and 99.99% TNR on the ISCX-IDS2012 dataset, detecting SSH-Bruteforce, HTTP-Bruteforce, Infiltration, Botnets, and Portscan. It achieved 99.97% accuracy and 99.69% F1-score for FTP-Patator, SSH-Patator, Botnets, PortScan, and Infiltration threats in CICIDS2017, where some Transformer only versions were categorized under moderate-performing approaches.

RM-DDCG + Transformer (RM stands for Random Masked Data blocks and DDCG stands for Deep Convolutional Generative Adversarial Network) version in S26 [78] detected FTP-Patator, SSH-Patator, Botnets, PortScan, and Infiltration classes in CICIDS2017 in a multiclass classification yielding 98.12% accuracy with a high weighted F1-score of 98.46%.

Proposed CNN + Transformer approach in S36 [36] achieves exceptionally high performance across multiple network threat types, including challenging ones such as U2R, R2L, and Fuzzers. On the NSL-KDD dataset, the proposed CNN + Transformers model excelled at detecting probing attacks, achieving 99.64% accuracy and a 99.30% F1-score. Notably, it achieved exceptional results on R2L and U2R threats, with accuracies exceeding 99.7% and F1-scores of 98.63% and 99.14%, respectively. R2L and U2R threats are particularly challenging due to their subtlety and low frequency. On the UNSW-NB15 dataset, the model maintained its high accuracy against backdoors, with an accuracy of 99.10%, and fuzzers, with an accuracy of 97.40%, complemented by strong recall scores of 98.10% and 99.00%, demonstrating the robustness of the approach. Like R2L and U2R, Fuzzers are also known for being challenging to detect, as they generate highly variable and semi-random traffic with no fixed signatures or patterns, to find network vulnerabilities.

Table 11. Dataset evaluation instances of top-performing GNNs.

ID	Model	Dataset	Threat Type	Performance %
S1	GAT	CICIDS2018 + synthetic zero-day traffic	Zero-Day	A: 99.12, P: 99.18, R: 99.09, F1: 99.13, FPR: 0.35
S1	GAT	CTU-13 + synthetic zero-day traffic	Zero-Day	A: 98.90, P: 98.85, R: 98.82, F1: 98.83, FPR: 0.31
S1	GAT	USTC-TFC2016 + synthetic zero-day traffic	Zero-Day	A: 99.48, P: 99.43, R: 99.40, F1: 99.42, FPR: 0.30
S1	GAT	UJS-IDS2022 + synthetic zero-day traffic	Zero-Day	A: 98.12, P: 98.18, R: 98.00, F1: 97.13, FPR: 0.25
S2	MLP + CR_Layer + GAT	Malware Capture CTU-13 dataset	APT	A: 99, P: 100, R: 99, F1: 99
S3	GCN	CERT v4.2	Insider Threats (AB-I: Data Exfiltration)	A: 98.67, P: 99.61, F1: 98.59, AUC: 98.66, TPR: 97.7, FPR: 0
S3	GCN	CERT v4.2	Insider Threats (AB-II: Credential Theft)	A: 99.56, P: 99.54, F1: 99.53, AUC: 99.56, TPR: 99.56, FPR: 0
S3	GCN	CERT v4.2	Insider Threats (AB-III: Malicious Tunneling)	A: 99.67, P: 99.14, F1: 99.13, AUC: 98.63, TPR: 99.15, FPR: 0
S4	GCN + LSTM	CICIDS2017	Probing, U2R, R2L	A: 99.24, P: 98.50, R: 98.62, F1: 98.72
S4	GCN + LSTM	CTU-13	Botnets	A: 99.31, P: 98.04, R: 99.53, F1: 99.47
S5	CNN + GAT	ISCX-IDS2012	SSH-Bruteforce, HTTP-Bruteforce, Infiltration, Botnets, Portscan	A: 99.75, P: 99.78, R: 99.73, F1: 99.76
S5	CNN + GAT	CICIDS2017	FTP-Patator, SSH-Patator, Botnets, PortScan, Infiltration	A: 99.97, P: 99.94, R: 99.97, F1: 99.96
S6	GCN + LSTM + SM	OpTC	Lateral Movement	P: 98.6, R: 94.9, AUC: 99.4, TPR: 94.9, FPR: 0.013
S10	GCN + Extra Trees	CTU-13	C2 Botnets, P2P Botnets	A: 98.85, R: 92.90, F1: 91.33
S10	GCN + Extra Trees	ISCX-2014	C2 Botnets, P2P Botnets	A: 99.10, R: 94.66, F1: 91.86
S10	GCN + Extra Trees	CICIDS2017	Kelihos botnet (P2P Botnets)	A: 98.23, R: 93.77, F1: 95.38
S12	GCN + GraphSAGE	CICIDS2017	FTP-Patator, SSH-Patator, Botnets, PortScan, Infiltration	A: 99.76, P: 99.26, R: 99.51, F1: 99.76
S12	GCN + GraphSAGE	UNSW-NB15	Fuzzers, Analysis, Backdoor, Reconnaissance, Shellcode	A: 98.65, P: 98.01, R: 98.97, F1: 98.65
S13	GraphSAGE	DARPA E3—Cadets	Initial Compromise, Persistence, Privilege Escalation	P: 94, R: 99, F1: 96, FPR: 0.072
S13	GraphSAGE	DARPA E3—Trace	Lateral Movement, Defense Evasion	P: 95, R: 99, F1: 97, FPR: 0.009
S13	GraphSAGE	DARPA E3—Theia	Process Injection, Fileless Malware	P: 92, R: 99, F1: 95, FPR: 0.017
S13	GraphSAGE	DARPA OpTC (Threat 1)	Initial Access, Lateral Movement	P: 91, R: 94, F1: 92, FPR: 0.0009
S13	GraphSAGE	DARPA OpTC (Threat 3)	Backdoor	P: 92, R: 92, F1: 92, FPR: 0.0002
S17	GAT	PeerRush + CTU-13 + MAWIWorkingGroupTrafficArchive	P2P Botnets	P: 90.5, R: 100, F1: 95, FPR: 0.6
S18	GAT	UNSW-NB15	Fuzzers, Analysis, Backdoor, Reconnaissance, Shellcode	F1: 93.41, A: 93.17, R: 88.12

Table 11. *Cont.*

ID	Model	Dataset	Threat Type	Performance %
S18	GAT	CampusNet-Logs	Reconnaissance, Unauthorized login attempts, Persistence, Privilege Escalation, Defense Evasion, Credential Access, Lateral Movement	F1: 99.13, A: 98.62, R: 98.62
S20	AHGCN	Custom Honeypot Dataset	Botnets	P: 98.81, R: 97.65
S20	AHGCN	CTU-13	Botnets	P: 98.24, R: 98.31
S21	GCN + RL	ISCX-Botnet + CAIDA + Synthetic P2P	P2P Botnets	A: 99.78, P: 98.89, R: 98.92, F1: 98.91
S22	E-GraphSAGE	NF-UNSW-NB15-v2 (0% contamination)	Fuzzers, Analysis, Backdoor, Reconnaissance, Shellcode	A: 98.74, Macro-F1: 92.72
S22	E-GraphSAGE	NF-UNSW-NB15-v2 (4% contamination)	Fuzzers, Analysis, Backdoor, Reconnaissance, Shellcode	A: 98.74, Macro-F1: 92.85
S22	E-GraphSAGE	NF-CSE-CIC-IDS2018-v2 (0% contamination)	Brute Force FTP, SSH, Infiltration, Botnets, PortScan	A: 98.54, Macro-F1: 98.06
S22	E-GraphSAGE	NF-CSE-CIC-IDS2018-v2 (4% contamination)	Brute Force FTP, SSH, Infiltration, Botnets, PortScan	A: 96.36, Macro-F1: 95.28
S22	E-GraphSAGE	NF-UNSW-NB18-v2 (0% contamination)	Fuzzers, Analysis, Backdoor, Reconnaissance, Shellcode	A: 98.54, Macro-F1: 96.36
S22	E-GraphSAGE	NF-UNSW-NB18-v2 (4% contamination)	Fuzzers, Analysis, Backdoor, Reconnaissance, Shellcode	A: 98.06, Macro-F1: 95.28
S24	GraphSAGE	Unicorn SC-2 dataset	APT (Malware download, Remote code execution, etc.)	A: 94, P: 93, R: 96, F1: 94, FPR: 0.072
S24	GraphSAGE	DARPA TC#3 Theia	APT (Unauthorized access, Malware implantation)	P: 89, R: 99, F1: 94, FPR: 0.009
S24	GraphSAGE	DARPA TC#3 Trace	APT (Rootkit installation, Process injection)	P: 76, R: 99, F1: 86, FPR: 0.017
S24	GraphSAGE	DARPA TC#3 Cadets	APT (Process manipulation, Network hijacking)	P: 79, R: 99, F1: 88, FPR: 0.0009
S24	GraphSAGE	DARPA TC#3 Fivedirections	APT (Initial access, Data theft)	P: 79, R: 89, F1: 84, FPR: 0.0002
S24	GraphSAGE	DARPA TC#5 Theia	Multi-stage APT with attack paths	P: 86, R: 76, F1: 80, FPR: 0.074
S24	GraphSAGE	DARPA TC#5 Trace	Multi-stage APT with advanced evasion	P: 87, R: 99, F1: 93, FPR: 0.065
S28	Graph Transformer	CERT r4.2	APT-related login activities	F1: 92.44, AUC: 94.39, TPR: 92.53, FPR: 7.74
S28	Graph Transformer	CERT r5.2	APT-related login activities	F1: 93.52, AUC: 95.35, TPR: 93.52, FPR: 6.81
S28	Graph Transformer	PicoDomain	APT, Lateral Movement via Anomalous Authentications	F1: 88.89, AUC: 75, TPR: 100, FPR: 25
S29	GNN	CERT v6.2	Insider Threats (Espionage, Data theft, Policy violations)	F1: 95.58, AUC: 96.32, P: 95.52, R: 96.79

The symbol "#" refers to the engagement number used in the DARPA Transparent Computing dataset (e.g., "DARPA TC #3 THEIA" refers to the third engagement, or "E3," in the THEIA threat scenario).

Table 12. Dataset evaluation instances of moderately performing GNNs.

ID	Model	Dataset	Threat Type	Performance %
S6	GCN	LANL	Lateral Movement	P: 0.66, AUC: 99.16, FPR: 0.47
S6	GCN + GRU	LANL	Lateral Movement	P: 0.54, R:86.1, AUC: 99.12, TPR: 86.1, FPR: 0.5698
S7	GCN	TOR-nonTOR	C2 Traffic, Darknet Activity	A: 88
S8	GAN + GAT	CERT v4.2	Insider Threats	F1: 84.9, AUC: 90
S9	GNN	LANL	APT (Lateral Movement, C2 Traffic)	F1: 83, AUC: 90
S11	GCN + KNN	UNSW-NB15	Fuzzers, Analysis, Backdoor, Reconnaissance, Shellcode	F1: 81, A: 80
S11	GAT + KNN	UNSW-NB15	Fuzzers, Analysis, Backdoor, Reconnaissance, Shellcode	F1: 79, A: 80
S11	GraphSAGE + KNN	UNSW-NB15	Fuzzers, Analysis, Backdoor, Reconnaissance, Shellcode	F1: 83, A: 81
S18	GAT	CICIDS2017	FTP-Patator, SSH-Patator, Botnets, PortScan, Infiltration	F1: 83.66, A: 69.82
S19	E-GraphSAGE	UNSW-NB15	Reconnaissance	F1: 84.17, P: 92.37

Table 13. Dataset evaluation instances of underperforming GNNs.

ID	Model	Dataset	Threat Type	Performance %
S6	GraphSAGE	LANL	Lateral Movement	P: 0.01, FPR: 24.57
S6	GAT	LANL	Lateral Movement	P: 0.02, FPR: 23.17
S19	E-GraphSAGE	UNSW-NB15	Analysis	F1: 17.32, R: 9.98
S19	E-GraphSAGE	UNSW-NB15	Backdoor	F1: 13.35, R: 7.51
S19	E-GraphSAGE	UNSW-NB15	Fuzzers	F1: 69.17, P: 74.58
S27	GCN	UNSW	APT (DoS, Probing, R2L, U2R)	F1: 70.45, Jaccard: 57.99

S36 [36] further reported that the proposed model employed several architectural innovations, such as multi-scale feature extraction supported by parallel CNN layers (with 1×1, 3×3, 5×5 kernels) to capture much granular, local packet-level network patterns and global flow-level network patterns, and fusion of multi-scale features using Cross Feature Enrichment before passing them to the Transformer. Furthermore, the Transformer uses "focal loss" loss function to deal with dataset class imbalances. The success of this Transformer in flagging probing activities with a 99.3% F1-score is explained by its ability to accurately model sequential port scans, such as those performed with Nmap. Unlike S23, the success of R2L/U2R, with F1-scores of 98.6–99.1%, highlights the effectiveness of the CNN + Transformer in capturing low-frequency threat signatures by combining CNNs' local feature detection with Transformers' long-term dependency modeling. Moreover, the study also demonstrates the effectiveness of using focal loss to prioritize rare threat samples.

All Transformer variants in S30 [80], which are Transformer shallower encoder, GPT 2.0, and BERT, yielded 98+% F1-score in the NSL-KDD dataset, detecting network threats such as Probing, R2L, and U2R. The same model variants achieved F1-scores ranging from 95.41% to 97.05% when evaluated against CSE-CIC-IDS2018, detecting Bruteforce, Botnets, and Infiltration. Similarly, the BERT model in S31 [81] achieved 99.7% accuracy and 99.2% F1-score for Brute Force, SQL Injection, and Data Exfiltration.

In the study S14 [67], Transformer + CNN-BiLSTM excelled at detecting botnets in CTU with 99.84% accuracy, 99.7% precision, and 99.85% recall. The same model recorded 91.92 accuracy, 91.45 precision, and 91.45 recall in detecting botnets in the ISCX-2014 dataset. In study S16 [69], the Tab Transformer + GBM is successful at detecting probing in firewall logs with an 90.5% accuracy by detecting sequential port access patterns.

5.3.3. Transformer Moderate Performers

Transformers in this category achieved accuracy between 80% and 90% with occasional drops in recall and precision. For example, the CGAN + Vanilla Transformer in S23 [75] detected probing threats in the KDDcup99 dataset with an 87.5% F1-score, but struggled to detect R2L and U2R in a class-wise classification, resulting in F1-scores of 24.2% and 15.5%, respectively. Thus, the latter two scenarios, R2L and U2R detection, were categorized under the "underperformers" category.

BERT in S30 [80] showed a moderate 76.23% F1 score on UNSW-NB15, detecting Fuzzers, Analysis, Backdoor, Reconnaissance, and Shellcode, despite the high detection rates of 98.9%. Moreover, the FAR was elevated up to 2.98% as well, whereas the GPT 2.0 version had 1.07% FAR with a slightly above detection rate of 99.82%.

5.3.4. Transformer Underperformers

CGAN + Vanilla Transformer in S23 [75] fails to detect R2L and U2R threats in KDDcup99, where it achieved a lower Recall of 13.9% and 13.2% and a lower F1-Score of 24.2% and 15.5%, respectively. Lastly, the Transformer only version in S25 [77] could only achieve 37.9% of F1-score for detecting Fuzzers, Analysis, Backdoor, Reconnaissance, and Shellcode network threats in UNSW-NB15.

The failure of underperforming Transformers is not due to any inferiority of their architecture; similar to GNN underperformers, the failure is due to incorrect application of them. Although S23 [73] used CGAN oversampling, the R2L and U2R threats often occur instantly, and thus their synthesized samples may not have presented a meaningful sequential context for the Transformer to excel. This is confirmed by S23's [75] success at detecting Probing, which is a threat that has a temporal aspect due to its sequential nature of unauthorized scanning of the victim network and S25's [77] exceptional success in

bruteforce, botnets and portscans. Moreover, unlike S36's [36] successful R2l/U2R detecting hybrid CNN architecture, the CGAN + Transformer approach in S23 [73] may lack the mechanism to capture granular packet-level anomalies in its generated data. To further worsen the S23's [75] performance, KDDCup99 is an extremely unbalanced dataset, with only < 0.01% U2R records, and no special class imbalance dealing technique like Focal Loss has been used at the classifier level. On the other hand, S25 [77] suffers from a lack of learnable sequential patterns when dealing with Fuzzers, which basically send random bursts of noisy garbage traffic.

The comparison between outperformers and top performers provides valuable insights into Transformer-based approaches for network threat detection. Some of those insights are the importance of hybrid designs to deal with stateless threats, such as the one proposed in S36 [36], and the effectiveness of class balancing solutions, such as Focal Loss in S36 [36] or GAN oversampling in S26 [78]. Top performers, moderate performers and under performers are displayed in Tables 14–16, respectively.

5.4. Performance of Reinforcement Learning Algorithms

75% of the unique threat evaluation instances done with RL studies demonstrated top performance backed by ≥90% F1-score/Accuracy/AUC & other relevant metrics. 25% of evaluation instances demonstrated moderate performance, supported by 80–89% F1-score/Accuracy/AUC & other relevant metrics. And there are no underperformers categorized under the <80% performance metrics range.

All the top-performing RL methods share some similar patterns and design strategies. A common pattern emerges among these top performers, which is combining them with DL models to compensate for general and standalone RL limitations. S15 combines DQN with BiLSTM and CNN to extract temporal and spatial features from network traffic. S33 combines DQN with DAE (Denoising Autoencoder), where DAE helps filter adversarial attack noise. The study shows that after adding DAE, the F1-score drop was reduced to 1.2% from 14%.

Moreover, S15 [68], S33 [83] and S34 [84] employ optimal reward functions to suit the detection scenario. S15 [68] balances exploration and exploitation by assigning +1 and −1 for correct and incorrect classifications, respectively, and 0 for requesting human intervention, which is particularly innovative. 0 reward-driven exploration motivates the agent to ask for help when the label is unclear with high uncertainty, while avoiding over-reliance, and thus is particularly effective in detecting zero-day threats. Over time, the agent learns to make predictions without relying on humans as it learns patterns associated with unknown threats. S34's [84] reward strategy is based on TCP connection failures, such as those presented with SYN-ACK handshake anomalies. The positive rewards are given to the agent for blocking malicious network flows, and negative rewards are given if the agent allows them. This particular approach of "protocol level failure" based rewards makes the agent robust to data manipulation tactics common with adversarial attacks and thus, enables the A3C model to remain accurate at 98.68%. Top performers and moderate performers are displayed in Tables 17 and 18, respectively.

Table 14. Dataset evaluation instances of top-performing Transformers.

ID	Model	Dataset	Threat Type	Performance %
S14	Transformer + CNN-BiLSTM	CTU-13	Botnets	A: 99.84, P: 99.7, R: 99.85
S14	Transformer + CNN-BiLSTM	ISCX-2014	Botnets	A: 91.92, P: 91.45, R: 91.45
S16	TabTransformer + GBM	NAD firewall logs	Probing-IP sweep, Port sweep, Nmap	Validation Fβ Score: 0.905 (not in %)
S23	CGAN + Vanilla Transformer	KDDcup99	Probing, R2L, U2R	A: 93.07, F1: 93.68
S25	Transformer + MLP	ISCX-IDS2012	SSH/HTTP-Bruteforce, Infiltration, Botnets, Portscan	A: 99.98, F1: 99.8, MCC: 0.9977, TPR: 99.77, TNR: 99.99
S25	Transformer + MLP	UNSW-NB15	Fuzzers, Analysis, Backdoor, Reconnaissance	A: 98.77, F1: 92.37
S25	Transformer + MLP	CICIDS2017	FTP/SSH-Patator, Botnets, PortScan	A: 99.97, F1: 99.69, MCC: 0.9967, TPR: 99.96, TNR: 99.97
S25	Transformer	ISCX-IDS2012	SSH/HTTP-Bruteforce, Infiltration, Botnets	A: 99.07, F1: 91.7, MCC: 0.9149, TPR: 99, TNR: 99.07
S26	RM-DDCG + Transformer	CICIDS2017 Multiclass	FTP/SSH-Patator, Botnets, PortScan	A: 98.12, R: 97.86, Weighted F1: 98.46
S30	Transformer Shallow Encoder	CSE-CIC-IDS2018	Bruteforce, Botnets, Infiltration	F1: 97.05, Detection Rate: 95.9, FAR: 0.1
S30	Transformer Shallow Encoder	NSL-KDD	Probing, R2L, U2R	F1: 98, Detection Rate: 97.68, FAR: 1.29
S30	Transformer Shallow Encoder	UNSW-NB15	Fuzzers, Analysis, Backdoor, Shellcode	F1: 90.45, Detection Rate: 99.89, FAR: 1.08
S30	Transformer Shallow Decoder	CSE-CIC-IDS2018	Bruteforce, Botnets, Infiltration	F1: 96.02, Detection Rate: 95.21, FAR: 0.26
S30	Transformer Shallow Decoder	NSL-KDD	Probing, R2L, U2R	F1: 98.29, Detection Rate: 92.77, FAR: 2.68
S30	Transformer Shallow Decoder	UNSW-NB15	Fuzzers, Analysis, Backdoor	F1: 88.94, Detection Rate: 99.79, FAR: 1.22
S30	GPT 2.0	CSE-CIC-IDS2018	Bruteforce, Botnets, Infiltration	F1: 96.5, Detection Rate: 95.86, FAR: 0.11
S30	GPT 2.0	NSL-KDD	Probing, R2L, U2R	F1: 98, Detection Rate: 97.87, FAR: 1.02
S30	GPT 2.0	UNSW-NB15	Fuzzers, Analysis, Backdoor	F1: 89.95, Detection Rate: 99.82, FAR: 1.07
S30	BERT	CSE-CIC-IDS2018	Bruteforce, Botnets, Infiltration	F1: 95.41, Detection Rate: 94.27, FAR: 0.43
S30	BERT	NSL-KDD	Probing, R2L, U2R	F1: 94.51, Detection Rate: 90.77, FAR: 1.03
S31	BERT	cve-attack-ttp-kaggle	Brute Force, SQL Injection, Data Exfiltration	A: 99.7, P: 99.4, R: 98.8, F1: 99.2, FPR: 1.1
S32	Transformer + CNN	Network Intrusion-kaggle	Anomalous TCP Connections	A: 98.54, P: 98.2, R: 97.302, F1: 97.5, Detection Rate: 97.5
S32	Transformer + CNN	NSL-KDD	Probing, R2L, U2R	A: 97.75, P: 96.33, R: 97.12
S36	CNN + Transformer	NSL-KDD	Probing, R2L, U2R	A: 99.25, P: 99.07, R: 99.02, F1: 99.05
S36	CNN + Transformer	NSL-KDD	Probing	A: 99.64, P: 99.38, R: 99.22, F1: 99.3
S36	CNN + Transformer	NSL-KDD	R2L	A: 99.82, P: 98.24, R: 99.03, F1: 98.63

Table 14. *Cont.*

ID	Model	Dataset	Threat Type	Performance %
S36	CNN + Transformer	NSL-KDD	U2R	A: 99.73, P: 98.9, R: 99.38, F1: 99.14
S36	CNN + Transformer	UNSW-NB15	Analysis	A: 98.7 R: 98.5
S36	CNN + Transformer	UNSW-NB15	Backdoor	A: 99.1 R: 98.1
S36	CNN + Transformer	UNSW-NB15	Fuzzers	A: 97.4 R: 99
S36	CNN + Transformer	UNSW-NB15	Reconnaissance	A: 97 R: 99.7
S36	CNN + Transformer	UNSW-NB15	Shellcode	A: 97.5 R: 99

Table 15. Dataset evaluation instances of moderately performing Transformers.

ID	Model	Dataset	Threat Type	Performance %
S16	TabTransformer	NAD firewall logs	Probing-IP sweep, Port sweep, Nmap	Validation Fβ Score: 0.833 (not in %)
S23	CGAN + Vanilla Transformer	UNSW-NB15	Fuzzers, Analysis, Backdoor, Reconnaissance	A: 89.38, P: 90.12, R: 89.38, F1: 89.75
S23	CGAN + Vanilla Transformer	KDDcup99	Probing	P: 84, R: 91.3, F1: 87.5
S25	Transformer	CICIDS2017	FTP/SSH-Patator, Botnets, PortScan	A: 96.27, F1: 68, MCC: 0.6972, TPR: 96.7, TNR: 96.25
S26	Transformer	CICIDS2017 Multiclass	FTP/SSH-Patator, Botnets, PortScan	A: 89.53, R: 89.76, Weighted F1: 89.6
S30	BERT	UNSW-NB15	Fuzzers, Analysis, Backdoor	F1: 76.23, Detection Rate: 98.9, FAR: 2.98
S32	Transformer + CNN	CSE-CIC-IDS2018	Bruteforce, Botnets, Infiltration	A: 81.87, P: 96.45, R: 70.11
S32	Transformer + CNN	UNSW-NB15	Fuzzers, Analysis, Backdoor	A: 85.55, P: 86.24, R: 85.55

Table 16. Dataset evaluation instances of underperforming Transformers.

ID	Model	Dataset	Threat Type	Performance %
S23	CGAN + Vanilla Transformer	KDDcup99	R2L	P: 91.6, R: 13.9, F1: 24.2
S23	CGAN + Vanilla Transformer	KDDcup99	U2R	P: 19, R: 13.2, F1: 15.5
S25	Transformer + MLP	UNSW-NB15	Fuzzers, Backdoor	A: 82.49, F1: 37.9, MCC: 0.3559, TPR: 71.44, TNR: 83.39

Table 17. Dataset evaluation instances of top-performing RL algorithms.

ID	Model	Dataset	Threat Type	Performance %
S15	CNN-BiLSTM + DQN	NSL-KDD	Probing	A:93.58, P:90, R:94.6, F1:92.2
S15	CNN-BiLSTM + DQN	NSL-KDD	R2L	A:95.85, P:95.9, R:95.8, F1:94.4
S15	CNN-BiLSTM + DQN	NSL-KDD	U2R	A:96.23, P:96.2, R:96.3, F1:94.8
S15	CNN-BiLSTM + DQN	UNSW-NB15	Reconnaissance	A:88.23, P:90.62, R:91.23, F1:90.92
S15	CNN-BiLSTM + DQN	UNSW-NB15	Analysis	A:90.12, P:89.15, R:90.28, F1:89.71
S33	DQN	NSL-KDD	Probing, R2L, U2R	A:97.4, P:96.5, R:99.1, F1:97.8
S33	DQN	CICIDS2017	PortScan	A:98.7, P:98.6, R:99.4, F1:98.9
S33	DQN + DAE	Custom NSL-KDD (Zero-Day)	Zero-Day	A:96.2, FPR:1.42
S33	DQN + DAE	Custom CICIDS2017 (Zero-Day)	Zero-Day	A:96.5, FPR:1.26
S34	A3C	UNSW + AWID + NSL-KDD	Shellcode, Reconnaissance, etc.	A:98.68, P:98.4, R:98.9
S35	Dueling DQN	CICIDS2018	Botnets	P:99.79, R:98.86, F1:99.32
S35	Dueling DQN	CICIDS2018	Bruteforce	P:98.7, R:99.91, F1:99.31

Table 18. Dataset evaluation instances of moderately performing RL algorithms.

ID	Model	Dataset	Threat Type	Performance %
S15	CNN-BiLSTM + DQN	UNSW-NB15	Fuzzers	A:88.46, P:89.52, R:88.85, F1:89.18
S15	CNN-BiLSTM + DQN	UNSW-NB15	Backdoor	A:89.58, P:89.98, R:86.29, F1:88.1
S33	DQN	Custom NSL-KDD (Zero-Day)	Zero-Day	A:85.2, FPR:2.32
S33	DQN	Custom CICIDS2017 (Zero-Day)	Zero-Day	A:85.2, FPR:4.0

5.5. Key Applications of GNNs

The top-performing GNN studies employ their GNN methods to model network traffic, system logs, and sophisticated network threats into graph-structured data by representing spatial interrelationships between the relevant entities. These entities are commonly defined by hosts, IPs or sessions. Some of the key applications of top-performing GNN studies are as follows:

5.5.1. Malicious Traffic Detection

S1 [56] successfully detects adversarial perturbations in network traffic with adaptive GNNs and dynamic attention. GAT in S5 [59] utilizes raw packet bytes to model traffic interaction graphs (TIGs), where the edges encode meaningful information like protocol-level interactions and utilize an attention mechanism to calculate the importance of critical edges. S12 [26] combines the strengths of both GraphSAGE and GCN and maps network flows into graph nodes and shared IPs into edges. And thus, almost all the successful GNN studies outperform traditional ML and DL models (e.g., SVMs and CNNs) by capturing hidden structural patterns in network traffic left by sophisticated network threats.

5.5.2. APT and Insider Threat Detection

The GCN model in S3 [29] constructs associated session graphs (ASG) to detect insider threats. S28 [54] designs a unique Graph Transformer architecture with multi-headed attention to analyze "authenticate traces" across the entire network to identify anomalies left by APTs. It also employs multi-hop subgraphs to expose lateral movement paths. S29 [55] applies temporal GNNs to analyze long-term threat behaviors in provenance graphs. Therefore, successful GNN studies demonstrate the importance of inter-host dependencies and the use of additional techniques to preserve long-term dependencies, enhancing the detection of multistage network threats.

5.5.3. Botnet Detection

S17 [70] uses local graph attention to analyze host neighborhoods and excels at detecting p2p botnets. S20 [73] uses heterogeneous graphs made with nodes such as IPs, protocols and ports to detect botnet clusters. S21 employs a hybrid architecture designed with RL to detect botnets via topology and traffic co-verification. And thus, Botnets' graph-like communication patterns make them ideal for GNN-based detection.

5.5.4. Malicious Anomaly Detection in Dynamic Network Environments

S6 [60] uses a combination of GNN and RNN to add temporal dynamic attributes to their approach and thus, detect lateral movement in evolving network snapshots. However, the performance remains moderate. Furthermore, S18 [71] leverages its hyperbolic embeddings for successful hierarchical anomaly detection linked to network threats.

5.6. Methodological and Architectural Trends of GNNs vs. Their Limitations

As evident from the detailed analysis, GNNs clearly inherit strong methodological and architectural advances in addressing network-related threats. However, as with any other highly promising technology, there are limitations associated with those GNN advantages, as outlined in Table 19.

5.7. Strengths of GNNs

After careful analysis of GNN-based studies, it is evident that graph-structured relational modeling is the primary and best strength of GNNs. They easily capture hidden dependencies, such as lateral movement (S6 [60]), between hosts better than non-graph models. Furthermore, techniques such as dynamic attention, implemented in S1, and

graph regularization, implemented in S18 [71], can improve adversarial robustness. Temporal GNNs, such as S6 [60] and S29 [55], can update their graph representations as the underlying network changes and are thus adaptable to dynamic networks.

Table 19. Methodological and architectural trends of GNNs vs. their limitations.

GNN Variant	Examples	Methodological & Architectural Trends	Challenges & Limitations
GAT	S1 [56], S8 [62], S17 [70]	Dynamic Node & Edge Weighting with attention mechanisms—Attention mechanisms have been proven to improve robustness in GATs.	Attention calculations are computationally intensive.
GraphSAGE	S12 [26], S19 [72], S22 [74]	Neighborhood Sampling—This approach provides scalability for large graphs, making it ideal for large LAN networks.	May lose global structural information during neighborhood sampling.
Temporal GNNs	S6 [60], S29 [55]	Employing Temporal Techniques/Models- GNNs are inherently agnostic to temporal dynamics. Thus, it requires additional techniques to capture evolving attack patterns.	High memory consumption is guaranteed for long sequences.
Heterogeneous GNNs	S9 [63], S20 [73]	Hierarchical & Attributed Heterogeneous Graph Learning—Employs complex graphs with multiple, distinct node/edge types, such as hosts, processes, and protocols, to natively model the multi-modal nature of conventional LAN networks and threats.	Complex to train.
GCN	S3 [29], S4 [58], S7 [61], S10 [64], S14 [67]	Graph Convolution Operations—This is effective when threats and/or the network are modeled homogeneously.	Can suffer from over-smoothing.

5.8. Key Applications of Transformers

Transformers have demonstrated their capability of modeling long-range network-related dependencies in sequential data, such as the ones presented with network flows or temporal logs. During the investigation of the role of Transformers in network threat detection, some of their key applications have been identified and are as follows.

5.8.1. Payload and Flow-Based Threat Detection

S25 [77] combines tabular features with payload Transformers to deliver exceptionally high performance. Transformer Multi-Receive Field Fusion (MRFF) in S26 [78] analyzes packet headers and payloads and achieves high accuracy in detecting Botnets and Portscans. S30 [80] introduces a novel modular framework for interchangeable Transformer architectures, which can be plugged into a flow-based IDS. And thus, these Transformers outperform RNNS and CNNs in capturing semantic traffic in raw network traffic data.

5.8.2. Log-Based Anomaly Detection

The approach in S23 balances imbalanced log data with Conditional Generative Adversarial Network (CGAN) oversampling before the classification is done by the Transformer.

5.8.3. Zero-Day and Adaptive Threat Detection

In S23 [75], BERT excels at detecting unseen threats by utilizing zero-shot learning and MITRE ATT&CK embeddings, thus excelling at detecting Brute force and SQL injection threats with high performance. In S32 [82], Transformers utilize VAE to regularize input feature extraction and selection, thereby facilitating self-supervised training. S32 [82] excels at detecting both anomalous TCP connections in a Kaggle network intrusion detection dataset and more challenging threats, such as R2L and U2R, in the NSL-KDD dataset. These high-performing studies explain how Transformers can generalize to unknown threats by employing pretrained embeddings such as those provided by BERT and self-supervised learning facilitated by VAE.

5.9. Methodological and Architectural Trends of Transformers vs. Their Limitations

Some of the methodological and inherent architectural advances of Transformer models, along with their limitations, are shown in Table 20.

Table 20. Methodological and architectural trends of Transformers vs. their limitations.

Transformer Variant	Examples	Methodological & Architectural Trends	Challenges & Limitations
BERT-style pre-trained	S31 [81]	LLM integration: Zero-shot detection via semantic embeddings.	Large model and pretraining data size
Shallow variants	S30 [80]	Lite and shallow Transformer Adoption: Reduces parameters (e.g., 2-layer transformers vs. BERT's 12 layers).	Reduced parameters may sacrifice some accuracy.
Character-level Transformer	S25 [77]	Tokenization Free Transformers: Raw network payload analysis without tokenization.	Results longer sequences, thus require high memory and computational cost.
Strided multi-head Transformer	S32 [82]	Strided Multi-Heads: Efficient long-sequence modeling of network traffic.	Reduced attention computation due to the loss of some global feature importance.
Hybrid (CNN + Transformer)	S36 [36], S26 [78]	Combining Transformers with Spatial Models: Combines local spatial features (CNN) and global long-term sequential features (Transformer).	Increased complexity & computationally intensive.
TabTransformer	S16 [69]	Non-Sequential Transformer Variants: Tabular feature analysis, such as NetFlow analysis.	Lose sequential information.
Patch Segmentation Transformer	S26 [78]	Patch Segmentation: Divides packets or payloads into smaller segments/chunks for analysis	Struggles to model local features due to the loss of fine-grained details provided by packet data.

5.10. Strengths of Transformers

According to the comprehensive analysis conducted on Transformer-based studies, it is evident that they excel at modeling long-range dependencies tied to temporal and multi-stage network threats. Transformers often outperformed traditional sequential models in detecting subtle temporal threats and identifying the early stages of multi-stage attacks,

such as S28's [54] malicious login chain detections. Unlike traditional sequential ML models, Transformers use attention mechanisms to correlate with distant events, such as those presented in APT campaigns. To elaborate, Transformers might link initial reconnaissance activity to later data exfiltration, thereby connecting the patterns to highlight multistage APTs with high confidence. This was demonstrated by S31's [81] high performance in mapping real-time network traffic patterns with MITRE ATT&CK tactics, which require correlating distant incidents separated across time and different attack stages. Moreover, Transformers can also handle heterogeneous data such as logs, network packets or graphs, as demonstrated by S26 [78] and S30 [80]. Furthermore, the Transformers' generalizability strength is demonstrated by S22's [74] high F1-score, which was achieved without labeled data.

5.11. Key Applications of RL Algorithms

Reinforcement Learning is fundamentally different from conventional ML and DL algorithms, including GNNs and Transformers. It operates by adaptively learning through trial and error and a sequential decision-making process, rather than static classification of labels. Hence, in the context of network threat detection, the function of RL is diverse and often extends beyond active threat detection. Table 21 explains five distinct functions of RL-based studies.

Table 21. Functions and integration methods of RL algorithms.

RL Function	How RL is Integrated	Examples
Active Learning for Threat Sampling	RL selects high-risk samples (e.g., potential zero-day attacks with high probabilities of being malicious) received from BiLSTM as probability distributions. Such samples are then sent for human expert labeling. This reduces the annotation costs of previously unseen threats, such as the ones presented with zero-day threats.	S15 [68] (DQN + BiLSTM)
Adaptive Network Node Inspection Prioritization	RL agents dynamically decide which nodes/flows should be sent for inspection by the GCN component of the hybrid. This optimizes the detection efficiency of the RL-GCN hybrid.	S21 [53] (Policy Gradient + GCN)
Distributed Multi-Agent Detection	RL trains autonomous agents installed on routers to collectively detect network threats with attention mechanisms provided to them via a NN (Neural Network).	S33 [83] (Multi-agent DQN)
Exploration-Exploitation for Anomalies	RL actively classifies known threats while balancing the exploration of new threat patterns and the exploitation of known threats in real-time.	S34 [84] (A3C)
Reward-Driven Policy Optimization	RL replaces static thresholds with adaptive reward functions, such as penalizing FPs or FNs while actively detecting network threats.	S35 [85] (Dueling DQN)

5.12. Strengths of Reinforcement Learning Algorithms

RL offers the strength of adaptability and autonomous decision-making to conventional network threat detection in various ways. These include adaptive prioritization of nodes and edges to be inspected by other ML/DL models, such as GNNs (S21 [53]), autonomous adjustment of classification thresholds (S33 [83]), adaptive and real-time threat sampling (S15 [68]), and exploration of new threat patterns and exploitation of known

threats in real-time (S34 [84]). Furthermore, RL's adaptive learning strengths are ideal for learning from previously unknown threats and long-developing multistage attack patterns. Moreover, their ability to handle rare class imbalance issues (S35 [85]) with reward engineering and active sampling has also been demonstrated by the reviewed studies.

5.13. Challenges and Limitations of Reinforcement Learning Algorithms

RL faces significant challenges in detecting network threats, which are closely tied to its fundamental operation, including its reliance on Markov Decision Processes (MDPs). This limitation is clearly evident when dealing with large state spaces, large action spaces, and partial observability, which are common with complex network threats. Another challenge is RL's susceptibility to exploitation by threat actors during its initial training stage. RL algorithms are quite vulnerable at this stage, as threat actors can manipulate the rewards and exploration processes. Furthermore, until the agent properly learns to detect threats, the early stage of training leaves the network and its resources vulnerable. However, these challenges can be overcome with adversarial training and hybrid detection, which employs other ML and DL models.

5.14. Dataset Bias

Across the reviewed literature, a systemic bias toward a narrow set of benchmark datasets was observed. This bias arises not from the SLR methodology itself but from the choices made in primary studies, where a limited group of public datasets has become de facto standard. These include the CIC family (CICIDS2017, CSE-CIC-IDS2018, CIC-DDoS2019), the UNSW-NB15 dataset, legacy KDD-based datasets (KDDCup'99, NSL-KDD), and other sources such as CERT Insider Threat datasets. While these benchmarks have advanced IDS research significantly, their inherent design characteristics impose dataset level constraints that affect model generalization, realism, and comparability across studies.

1. CIC Family Datasets (CICIDS2017, CSE-CIC-IDS2018, CIC-DDoS2019)

These datasets dominate modern IDS research due to their comprehensiveness and detailed labeling. However, they share several structural issues.

- Class imbalance: Certain attack types (e.g., Infiltration, Heartbleed, SQL Injection) have very few instances, while benign traffic dominates, leading to skewed accuracy metrics.
- Non-stationary data: Traffic was captured over separate days, creating temporal discontinuities that models may exploit as artificial patterns.
- Controlled lab environment: All CIC datasets were generated using scripted attacks and synthetic normal traffic, lacking the diversity and background noise of real-world networks [86].
- Redundant and correlated features: Flow-based features often overlap, introducing redundancy that can distort feature importance analysis.
- Reproducibility gaps: Variations in preprocessing pipelines across studies (e.g., feature normalization, label consolidation) cause inconsistencies in reported performance.

2. UNSW-NB15

UNSW-NB15 was designed to replace older datasets such as KDDCup'99, but it also exhibits biases.

- Synthetic traffic generation: Network data were produced using the IXIA PerfectStorm tool rather than real enterprise captures [87].
- Limited protocol diversity: Fewer application layer behaviors and uniform normal traffic make models overfit to specific flow characteristics.

- Feature correlation: Certain attributes are highly collinear, reducing discriminative value for ML/DL models.
- Short capture duration: Restricts representation of long-term threat evolution or gradual attack stages such as APTs.

3. Legacy KDD-Based Datasets (KDDCup'99, NSL-KDD).

These remain in limited use for benchmarking simplicity or comparability, but they have severe limitations.

- Obsolete attack types that no longer reflect modern network threats.
- Feature duplication and redundancy leading to inflated classification metrics.
- Over simplified feature space unsuitable for testing complex deep models such as GNNs or Transformers.

4. CERT Insider Threat Datasets

Widely used for evaluating insider threat detection, these datasets are valuable yet constrained.

- Simulated organizational environments with scripted user behavior, lacking true behavioral diversity [88].
- Limited organizational roles and time span, which restrict generalization to real multi-department enterprises.
- Label uncertainty: Some insider behaviors are ambiguous or context dependent, potentially introducing labeling bias.
- Low Utilization of Other Specialized Datasets

Beyond the major benchmark families, several studies employed specialized datasets representing narrower network contexts or targeted threat categories. These include DARPA-style and APT-oriented datasets (e.g., DARPA TC 1998/1999, Theia, and Unicorn SC-2), botnet and traffic capture collections such as CTU-13, ISCX-IDS2012, ISCX-IDS2014, MAWI, and CAIDA, as well as organization-specific or campus datasets and custom honeypot or firewall logs generated for individual experiments. While these sources contribute to the diversity of traffic patterns and enable focused evaluation of certain attack scenarios, their overall adoption across the reviewed literature remains limited. This infrequent use restricts opportunities for consistent cross-study comparison and limits the generalizability of models beyond the dominant benchmark datasets.

5.15. GNNs, Transformers, and RL, Cross-Model Comparison

The following figures present heatmaps in which models are ordered alphabetically and include GNN-, Transformer-, and RL-based approaches. Figure 9 shows the heatmap of F1_Score (%). Figure 10 shows the heatmap of Accuracy (%). In both heatmaps, each cell indicates the rounded metric value for a given model–dataset combination. Missing values indicate that a metric was not reported for that model–dataset pair. Table 22 compares GNN, Transformer, and RL approaches across key aspects such as performance, strengths, limitations, and typical hybrid roles. It highlights where each model family performs best and summarizes their scalability, data needs, and generalization capabilities.

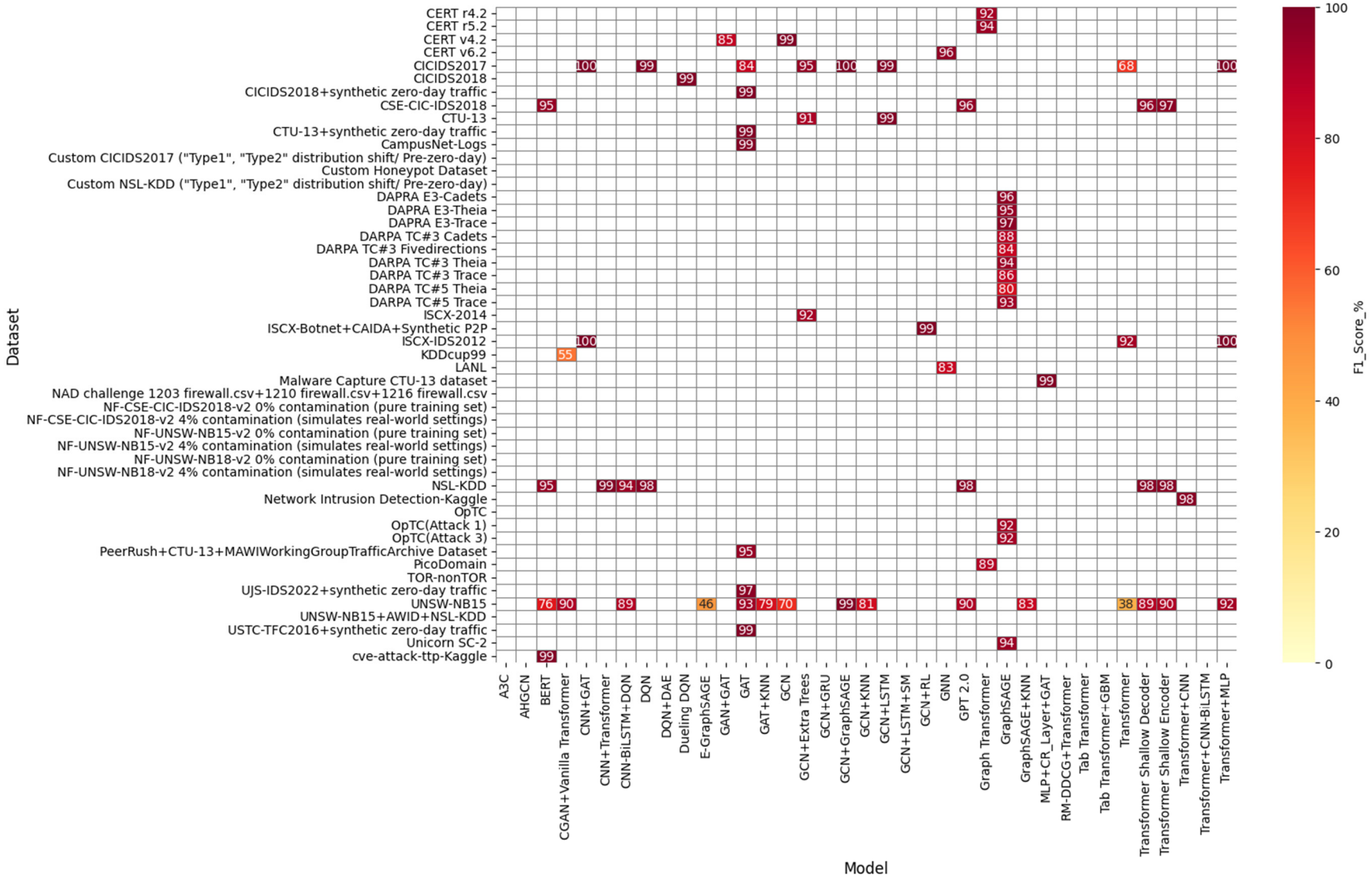

Figure 9. Dataset vs. model performance (F1_Score_%).

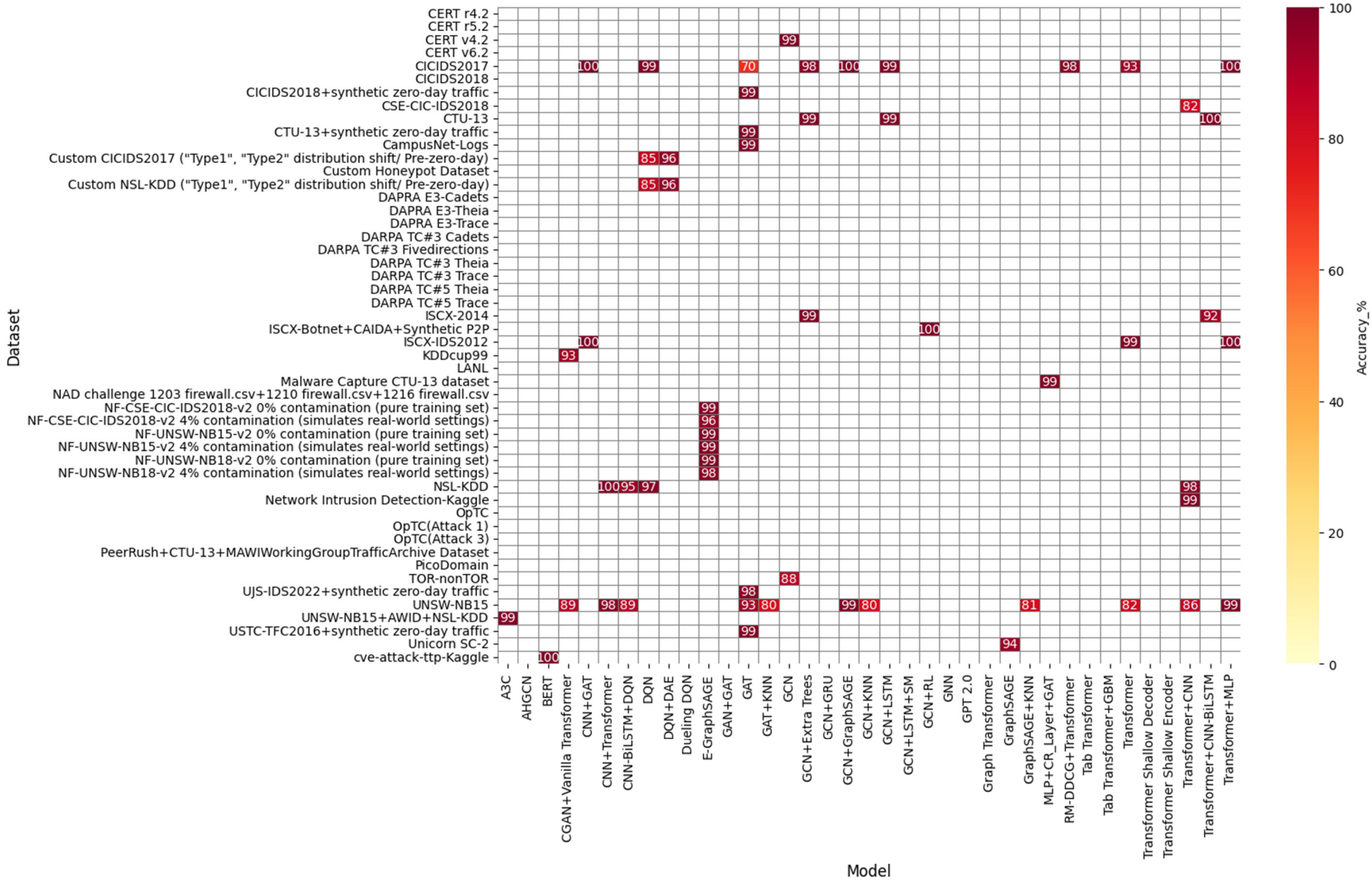

Figure 10. Dataset vs. model performance (Accuracy_%).

Table 22. GNNs, Transformers, and RL, cross-model comparison.

Aspect	Graph Neural Networks	Transformers	Reinforcement Learning
Aggregate performance distribution	74.2% of GNN evaluation instances are top performers, 16.1% are moderate, and 9.7% are underperformers.	74.41% evaluation instances are top performers, 18.60% are moderate, and 6.97% are underperformers.	75% of RL evaluation instances are top performers, 25% are moderate performers, and no underperformers are reported.
Representative top-performing examples	S5 [59] (GAT on CICIDS2017): Accuracy 99.97%, F1 99.96%. S12 [26] (GCN + GraphSAGE on CICIDS2017/UNSW): Accuracy 99.76%/98.65%, respectively. S1 [56] (GAT detecting zero-day): Accuracy up to 99.48%. S21 [53] (GCN + Policy-gradient RL hybrid): Accuracy 99.78% on ISCX-Botnet + CAIDA + synthetic P2P.	S25 [77] (Transformer + MLP on ISCX-IDS2012 & CICIDS2017): Accuracy up to 99.98%, F1 99.8%. S36 [36] (CNN + Transformer on NSL-KDD/UNSW): accuracies & F1-scores in the high-90 s on several threats (including R2L/U2R in reported cases). S31 [81]/S30 [80] (BERT/GPT variants) report F1-scores >95% on several datasets.	S15 [68] (CNN-BiLSTM + DQN on NSL-KDD/UNSW-NB15): Accuracy/F1 mostly in the 90% + range. S33 (DQN/DQN + DAE) reported an Accuracy of 96–97% on custom zero-day setups. S34 [84] (A3C) reported an Accuracy of 98.68%. S35 [85] (Dueling DQN) reported P/R/F1 $\approx$ 99% on CICIDS2018 Botnets/Bruteforce.
Typical strengths	Best strength: explicit graph-relational modeling. Captures inter-host dependencies, lateral movement, botnet topology, and provenance (session/provenance graphs). Dynamic attention/graph regularization helps adversarial robustness, temporal GNNs + hybrid components can model evolving threats.	Best strength: modeling long-range temporal dependencies and correlating distant events (useful for multi-stage APTs, log sequences, zero-day generalization when using pretrained embeddings/VAE/BERT). Combine well with spatial modules (CNN) or MLPs for the best results.	Best strength: Adaptivity & decision-making. RL enables adaptive sampling, prioritization (e.g., select nodes/flows for deeper inspection), exploration–exploitation for unknown/zero-day threats, and reward engineered handling of class imbalance. Often used in hybrids to improve efficiency.
Typical limitations/failure modes	Vanilla GNNs do not inherently capture temporal dynamics (need temporal modules, RNN/LSTM, temporal GNNs); can have high memory cost for long sequences; over-smoothing or training complexity for heterogeneous graphs.	High computational/memory costs; sensitivity to datasets lacking meaningful temporal structure; Transformer-only versions can fail on stateless/noisy threats unless combined with spatial hybrids or data balancing.	Large state/action spaces and partial observability challenge RL; vulnerability during early training to reward-poisoning/adversarial manipulation; needs care (adversarial training, hybridization) before deployment.
Typical hybrid patterns and role in hybrids	Many GNNs are hybridized with LSTM/GRU, CNNs, KNN, or RL agents (e.g., GCN + LSTM, GCN + RL). Hybrids address temporal or imbalance issues and often improve the detection of multistage threats.	Transformers are often combined with CNN or MLPs (Transformer + CNN, Transformer + MLP) or used with GAN/VAE for oversampling/self-supervision; shallow/lite variants or patch/strided designs are used for long sequences.	RL is most effective as a hybrid component (node/flow prioritization feeding GNN or DL classifier; active sampling; reward-based selection). The only full GCN–RL hybrid (S21) is reported, and partial GNN–Transformer hybrids were also observed.
Where each model family is best applied	Graph-native/topology-heavy threats: Botnets, lateral movement, provenance/APT stages, insider detection (when meaningful graphs can be constructed).	Long-range temporal and sequence tasks: Multi-stage APT correlation, log sequence anomaly detection, zero-day generalization (with pretrained embeddings/VAE). Hybrid designs work best for stateless or packet-level anomalies.	Adaptive/operational roles: Active sampling, inspection prioritization, reward-driven anomaly exploration. Most effective when augmenting existing classifiers and managing inspection resources, rather than serving as a standalone detection model.
Explainability & Interpretability	Moderate interpretability via node/edge importance (Graph Attention, explainable message passing). Graph visualization is possible.	Attention weights offer some interpretability, but global Transformer layers often act as black boxes. Visualization is limited to the token/flow-level.	Typically, the least interpretable. Learned policies are opaque and hard to trace to specific features or decisions.
Scalability/Real-time Feasibility	Dynamic graphs can increase cost, but efficient sampling (GraphSAGE, mini-batching) helps.	High parallelization potential via attention blocks, with a large memory footprint. Efficient variants (Lite, Strided, Tiny-Transformer) are emerging.	Requires environment simulation. Iterative training is slow. However, online policies enable adaptive real-time inference once trained.
Data Requirements	Can work with smaller labeled sets via relational generalization. Sensitive to topology quality.	Typically data hungry. Performance improves with large sequential corpora (e.g., flow logs).	Can learn with limited labeled data, but requires well defined reward functions and simulated environments.
Zero-Day & Generalization	Learns structural patterns. Generalizes to unseen topologies and node relations.	Captures sequential dependencies. Generalizes to unseen temporal patterns.	Learns adaptive policies. Generalizes dynamically to new scenarios if the reward is engineered well.

6. Future Research Trends and Directions

Based on the collective insights synthesized from this review, several emerging research trends and strategic recommendations can be identified. Accordingly, the following subsections outline both key future trends and explicit recommendations for advancing research in this domain.

6.1. Development of Hybrid GNN–Transformers

Modern network threats often exhibit both temporal and spatial characteristics. Hence, there is a growing need for comprehensive threat detectors that can model both spatial and temporal characteristics of such advanced and multistage threats. The review confirmed that GNNs are excellent at accurately modeling such spatial characteristics of network threats. Furthermore, conventional networks can be accurately modeled as graphs using GNNs. However, vanilla GNNs cannot inherently model the temporal aspect, which is crucial for detecting the subtle, evolving aspects presented in modern network threats.

Transformers, on the other hand, excel at modeling such temporal and subtle dependencies that are far distant from each other. This review noted a trend that these limitations have sometimes been addressed by combining GNNs with sequential models such as RNNs, and Transformers with spatial models such as CNNs.

Therefore, as GNNs and Transformers already excel in spatial and temporal modeling, respectively, combining them directly without relying on conventional CNNs or RNNs intermediaries would theoretically provide the best coverage of both spatial and temporal aspects mentioned above. Thus, future research into suitable GNN-Transformer hybrids that can complement each other in detecting the advanced, multistage network threats is highly recommended.

6.2. Reinforcement Learning and Transformer Synergy

While RL and Transformers have each demonstrated their strengths in standalone experiments in the evaluated studies, their combined and complementary potential is largely underexplored in the current network threat detection studies. There is a growing recognition that combining RL's adaptive decision making capabilities with Transformer-based feature learning could create highly responsive and context-aware detection systems.

Therefore, further exploration of RL–Transformer hybrid systems is encouraged. Researchers should examine how reinforcement learning can dynamically optimize Transformer model parameters, improve attention mechanisms, and manage trade-offs between detection accuracy, false positives, and computational efficiency in real-time network threat detection.

6.3. Generative AI for Threat Simulation and Augmentation

Generative Artificial Intelligence, which is predominantly powered by Transformer architectures, is gaining attention in cybersecurity.Beyond its role in content generation, GenAI can contribute to network threat detection by generating realistic synthetic attack data for training and by simulating evolving threat behaviors to improve model robustness. In particular, GenAI could complement GNN-Transformer systems by acting as a simulation and augmentation layer, producing plausible future network flows (benign or malicious) that enrich the model's ability to anticipate unseen attacks.

6.4. Exploration of Agentic AI for Automated Threat Detection Pipelines

Closely tied to advancements in Transformers, Agentic AI is also emerging, enabling systems to act more autonomously and adaptively.In the context of network threat detection,

such systems can potentially orchestrate data pre-processing, model retraining, and decision making with minimal human oversight.

7. Conclusions & Recommendations

7.1. Conclusions

This Systematic Literature Review analyzed and synthesized 36 high-quality empirical studies published between the years 2017 and 2025 regarding the use of Graph Neural Networks, Transformers, and Reinforcement Learning algorithms in the context of conventional LAN-based network threat detection. The included studies consisted of 23 journal articles and 13 conference papers. All the primary and secondary research questions were addressed during the analysis and synthesis of this review. The key contributions made during the reviewing process include:

1. A novel conceptual threat classification framework: A novel framework for classifying network threats and distinguishing them from network attacks, inspired by the MITRE ATT&CK tactics and Lockheed Martin cyber kill chain, was developed. This classification framework, along with the analysis and the synthesis, provided a deeper understanding into the stages of the cyber kill chain that GNNs, Transformers and RL models focus on and which early stages need attention to prevent network attacks.
2. Comparative synthesis: This review conducted a comprehensive, comparative analysis and qualitative synthesis with quantitative elements of the application of GNNs, Transformers and RL algorithms in conventional network threat detection. The preliminary literature review showed that an SLR conducted regarding all three of these models is rare or nonexistent. Thus, this SLR fills a significant gap in the current cybersecurity literature.
3. Trend identification: Significant trends were identified and documented for each of the model types. For GNNs and Transformers, there is a trend of increasing interest in hybrid architectures for network threat detection, as some GNNs were combined with traditional sequential models and some Transformer architectures were integrated with traditional spatial models to overcome their inherent architectural limitations.
4. Limitations & challenges identification: The review identified and documented limitations and challenges in the current literature while highlighting the solutions presented by the complementary studies.

7.2. Recommendations

1. Extended network threat detection SLRs for Cloud, IoT and flat network architectures: As organizations increasingly migrate operations to cloud environments, integrate with IoT networks, and adopt flat network architectures to simplify connectivity, reduce latency, and enhance scalability, future research should focus on systematic literature reviews and empirical studies addressing threat detection in these environments. Such reviews would support the adaptation and optimization of advanced threat detection frameworks for modern enterprise networks with minimal segmentation and distributed cloud resources.
2. Explainability and interpretability: Many DL models, including GNNs and Transformers, are considered black boxes. Therefore, it is recommended that future research investigates techniques to improve the transparency of these models, making their detection more transparent and interpretable. This can benefit cybersecurity experts and other authorities in understanding the reasoning process behind model detections, which is crucial for privacy-critical scenarios such as insider threat detection.
3. Utilizing lightweight Transformers: Despite their effectiveness in modeling temporal aspects, Transformers are often considered computationally intensive algorithms.

Further research is needed to develop lightweight Transformers that can produce decent performance in network threat detection tasks.

4. Meta-Analysis and Standardized Reporting: Meta-analyses are recognized for their robust statistical insights. Therefore, it is recommended to conduct a meta-analysis on the applicability of GNNs, Transformers, and RL algorithms in network threat detection. However, to make it feasible, the research community must collectively adhere to more standardized reporting practices, including unified evaluation metrics (e.g., a clear indication of the type of multi-class F1-score used, macro or micro).

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/electronics14214163/s1, PRISMA 2020 Checklist.

Author Contributions: Conceptualization, T.P.D.G. and J.A.G.; methodology, T.P.D.G.; validation, T.P.D.G.; formal analysis, T.P.D.G.; investigation, T.P.D.G.; writing—original draft preparation, T.P.D.G.; writing—review and editing, T.P.D.G., J.A.G., and S.K.R.; supervision, J.A.G. and S.K.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: No new datasets were created in this systematic literature review. Thus, data sharing is not applicable.

Acknowledgments: The authors have reviewed and edited the output and take full responsibility for the content of this publication.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AIDS	Anomaly-Based Intrusion Detection System
APT	Advanced Persistent Threat
AUC	Area Under the Curve
BERT	Bidirectional Encoder Representations from Transformers
BiLSTM	Bidirectional Long Short-Term Memory
CASP	Critical Appraisal Skills Programme
CIC	Canadian Institute for Cybersecurity
CNN	Convolutional Neural Network
DDoS	Distributed Denial of Service
DL	Deep Learning
DNN	Deep Neural Network
DRL	Deep Reinforcement Learning
DQN	Deep Q-Network
FAR	False Alarm Rate
FN	False Negative
FPR	False-Positive Rate
FP	False Positive
Gen-AI	Generative Artificial Intelligence
GAN	Generative Adversarial Network
GAT	Graph Attention Network
GCN	Graph Convolutional Network
GNN	Graph Neural Network
GPT	Generative Pre-trained Transformer
GRU	Gated Recurrent Unit
IDS	Intrusion Detection System
IP	Internet Protocol

LAN	Local Area Network
LLM	Large Language Model
LSTM	Long Short-Term Memory
MAC	Media Access Control
MCC	Matthews Correlation Coefficient
MDP	Markov Decision Process
ML	Machine Learning
MLP	Multilayer Perceptron
NIDS	Network Intrusion Detection System
OSF	Open Science Framework
P2P	Peer-to-Peer
PCA	Principal Component Analysis
PRISMA	Preferred Reporting Items for Systematic Reviews and Meta-Analyses
RL	Reinforcement Learning
RNN	Recurrent Neural Network
RoB	Risk of Bias
SDN	Software-Defined Network
SIDS	Signature-Based Intrusion Detection System
SLR	Systematic Literature Review
SMOTE	Synthetic Minority Over-sampling Technique
SOTA	State of the Art
SVM	Support Vector Machine
TCP/IP	Transmission Control Protocol/Internet Protocol
TNR	True-Negative Rate
TPR	True-Positive Rate
UAV	Unmanned Aerial Vehicle
VAE	Variational Autoencoder
VLAN	Virtual Local Area Network
ViT	Vision Transformer

Appendix A

Table A1. ROB list.

Study	Data Quality	Model Transparency	Reproducibility	Methodology	External Validity	Overall RoB
S1	Yes	Yes	No	Yes	Yes	Low
S2	No	Yes	Unclear	Yes	No	High
S3	Yes	Yes	Yes	Yes	Yes	Low
S4	Yes	Yes	No	Yes	Unclear	Low
S5	Yes	Yes	No	Yes	Yes	Low
S6	Yes	Yes	Yes	Yes	Yes	Low
S7	Unclear	Yes	No	No	No	High
S8	Yes	Yes	No	Yes	No	Moderate
S9	Yes	Yes	No	Yes	Unclear	Low
S10	Yes	Yes	No	Yes	Yes	Low
S11	Yes	Yes	No	Yes	No	Moderate
S12	Yes	Yes	No	Yes	Unclear	Low
S13	Yes	Yes	Yes	Unclear	Unclear	Moderate
S14	Yes	Yes	Unclear	Yes	No	Moderate
S15	Yes	Yes	No	Yes	No	Moderate
S16	Unclear	Partial	No	Yes	No	High
S17	Yes	Yes	No	Yes	Unclear	Low
S18	Yes	Yes	No	Yes	Yes	Low
S19	Yes	Yes	No	Yes	No	Moderate
S20	Yes	Yes	No	Yes	Yes	Low

Table A1. *Cont.*

Study	Data Quality	Model Transparency	Reproducibility	Methodology	External Validity	Overall RoB
S21	Yes	Yes	Yes	Yes	Yes	Low
S22	Yes	Yes	No	Yes	Unclear	Low
S23	Yes	Yes	No	Yes	No	Moderate
S24	Yes	Yes	No	Yes	Unclear	Low
S25	Yes	Partial	No	Yes	No	High
S26	Yes	Yes	No	Yes	No	Moderate
S27	Yes	Yes	No	Yes	Unclear	Low
S28	Yes	Yes	No	Yes	Yes	Low
S29	Yes	Yes	No	Yes	Yes	Low
S30	Yes	Yes	Yes	Yes	Unclear	Low
S31	Yes	Yes	No	Yes	Yes	Low
S32	Yes	Partial	No	Yes	Unclear	High
S33	Yes	Partial	No	Yes	No	High
S34	Yes	No	No	Yes	No	High
S35	Yes	Yes	No	Yes	No	Moderate
S36	Yes	Yes	No	Yes	Unclear	Low

References

1. García-Teodoro, P.; Díaz-Verdejo, J.; Maciá-Fernández, G.; Vázquez, E. Anomaly-based network intrusion detection: Techniques, systems and challenges. *Comput. Secur.* **2009**, *28*, 18–28. [CrossRef]
2. Díaz-Verdejo, J.; Muñoz-Calle, J.; Estepa Alonso, A.; Estepa Alonso, R.; Madinabeitia, G. On the Detection Capabilities of Signature-Based Intrusion Detection Systems in the Context of Web Attacks. *Appl. Sci.* **2022**, *12*, 852. [CrossRef]
3. Ali, M.L.; Thakur, K.; Schmeelk, S.; Debello, J.; Dragos, D. Deep Learning vs. Machine Learning for Intrusion Detection in Computer Networks: A Comparative Study. *Appl. Sci.* **2025**, *15*, 1903. [CrossRef]
4. Newsome, J.; Karp, B.; Song, D. Polygraph: Automatically generating signatures for polymorphic worms. In Proceedings of the 2005 IEEE Symposium on Security and Privacy (S&P'05), Oakland, CA, USA, 8–11 May 2005; pp. 226–241.
5. Samrin, R.; Vasumathi, D. Review on anomaly based network intrusion detection system. In Proceedings of the 2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT), Mysuru, India, 15–16 December 2017; pp. 141–147.
6. Khraisat, A.; Gondal, I.; Vamplew, P.; Kamruzzaman, J. Survey of intrusion detection systems: Techniques, datasets and challenges. *Cybersecurity* **2019**, *2*, 20. [CrossRef]
7. Ramzan, M.; Shoaib, M.; Altaf, A.; Arshad, S.; Iqbal, F.; Castilla, Á.K.; Ashraf, I. Distributed Denial of Service Attack Detection in Network Traffic Using Deep Learning Algorithm. *Sensors* **2023**, *23*, 8642. [CrossRef] [PubMed]
8. Hu, Z.; Shukla, K.; Karniadakis, G.E.; Kawaguchi, K. Tackling the Curse of Dimensionality with Physics-Informed Neural Networks. *Neural Netw.* **2024**, *176*, 106369. [CrossRef]
9. Caviglione, L. Trends and Challenges in Network Covert Channels Countermeasures. *Appl. Sci.* **2021**, *11*, 1641. [CrossRef]
10. Zhou, J.; Cui, G.; Hu, S.; Zhang, Z.; Yang, C.; Liu, Z.; Wang, L.; Li, C.; Sun, M. Graph neural networks: A review of methods and applications. *AI Open* **2020**, *1*, 57–81. [CrossRef]
11. Pujol-Perich, D.; Suarez-Varela, J.; Cabellos-Aparicio, A.; Barlet-Ros, P. Unveiling the potential of Graph Neural Networks for robust Intrusion Detection. *ACM Sigmetrics Perform. Eval. Rev.* **2022**, *49*, 111–117. [CrossRef]
12. Kundacina, O.; Gojic, G.; Cosovic, M.; Miskovic, D.; Vukobratovic, D. Scalability and Sample Efficiency Analysis of GraphNeural Networks for Power System State Estimation. *arXiv* **2023**, arXiv:2303.00105.
13. Guan, M.; Iyer, A.P.; Kim, T. Dynagraph: Dynamic graph neural networks at scale. In Proceedings of the 5th ACM SIGMOD Joint International Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA), Philadelphia, PA, USA, 12 June 2022.
14. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 6000–6010.
15. Xu, P.; Zhu, X.; Clifton, D.A. Multimodal Learning with Transformers: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 12113–12132. [CrossRef] [PubMed]
16. Qiang, W.; Zhongli, Z. Reinforcement learning model, algorithms and its application. In Proceedings of the 2011 International Conference on Mechatronic Science, Electric Engineering and Computer (MEC), Jilin, China, 19–22 August 2011; pp. 1143–1146.

17. Neufeld, E.A.; Bartocci, E.; Ciabattoni, A.; Governatori, G. Enforcing ethical goals over reinforcement-learning policies. *Ethics Inf. Technol.* **2022**, *24*, 43. [CrossRef]
18. Shakya, A.K.; Pillai, G.; Chakrabarty, S. Reinforcement learning algorithms: A brief survey. *Expert Syst. Appl.* **2023**, *231*, 120495. [CrossRef]
19. Khan, N.M.; Madhav, C.N.; Negi, A.; Thaseen, I.S. Analysis on Improving the Performance of Machine Learning Models Using Feature Selection Technique. In Proceedings of the International Conference on Intelligent Systems Design and Applications, Online, 12–15 December 2020; pp. 69–77.
20. Disha, R.A.; Waheed, S. Performance analysis of machine learning models for intrusion detection system using Gini Impurity-based Weighted Random Forest (GIWRF) feature selection technique. *Cybersecurity* **2022**, *5*, 1. [CrossRef]
21. Lee, J.; Pak, J.; Lee, M. Network Intrusion Detection System using Feature Extraction based on Deep Sparse Autoencoder. In Proceedings of the 2020 International Conference on Information and Communication Technology Convergence (ICTC), Jeju Island, Korea, 21–23 October 2020; pp. 1282–1287.
22. Jing, D.; Chen, H.B. SVM Based Network Intrusion Detection for the UNSW-NB15 Dataset. In Proceedings of the 2019 IEEE 13th International Conference on ASIC (ASICON), Chongqing, China, 29 October–1 November 2019; pp. 1–4.
23. Zhang, C.; Jia, D.; Wang, L.; Wang, W.; Liu, F.; Yang, A. Comparative research on network intrusion detection methods based on machine learning. *Comput. Secur.* **2022**, *121*, 102861. [CrossRef]
24. Dash, N.; Chakravarty, S.; Rath, A.K.; Giri, N.C.; AboRas, K.M.; Gowtham, N. An optimized LSTM-based deep learning model for anomaly network intrusion detection. *Sci. Rep.* **2025**, *15*, 1554. [CrossRef]
25. Udurume, M.; Shakhov, V.; Koo, I. Comparative Analysis of Deep Convolutional Neural Network—Bidirectional Long Short-Term Memory and Machine Learning Methods in Intrusion Detection Systems. *Appl. Sci.* **2024**, *14*, 6967. [CrossRef]
26. Tran, D.H.; Park, M. FN-GNN: A Novel Graph Embedding Approach for Enhancing Graph Neural Networks in Network Intrusion Detection Systems. *Appl. Sci.* **2024**, *14*, 6932. [CrossRef]
27. Hu, G.; Sun, M.; Zhang, C. A High-Accuracy Advanced Persistent Threat Detection Model: Integrating Convolutional Neural Networks with Kepler-Optimized Bidirectional Gated Recurrent Units. *Electronics* **2025**, *14*, 1772. [CrossRef]
28. Wu, Z.; Zhang, H.; Wang, P.; Sun, Z. RTIDS: A Robust Transformer-Based Approach for Intrusion Detection System. *IEEE Access* **2022**, *10*, 64375–64387. [CrossRef]
29. Ding, J.; Qian, P.; Ma, J.; Wang, Z.; Lu, Y.; Xie, X. Detect Insider Threat with Associated Session Graph. *Electronics* **2024**, *13*, 4885. [CrossRef]
30. Sun, Z.; Teixeira, A.M.H.; Toor, S. GNN-IDS: Graph Neural Network based Intrusion Detection System. In Proceedings of the 19th International Conference on Availability, Reliability and Security, Vienna, Austria, 30 July–2 August 2024; p. 14.
31. Jianping, W.; Guangqiu, Q.; Chunming, W.; Weiwei, J.; Jiahe, J. Federated learning for network attack detection using attention-based graph neural networks. *Sci. Rep.* **2024**, *14*, 19088. [CrossRef] [PubMed]
32. Ayoub Mansour Bahar, A.; Soaid Ferrahi, K.; Messai, M.-L.; Seba, H.; Amrouche, K. CONTINUUM: Detecting APT Attacks through Spatial-Temporal Graph Neural Networks. *arXiv* **2025**, arXiv:2501.02981. [CrossRef]
33. Ji, Y.; Huang, H.H. NestedGNN: Detecting Malicious Network Activity with Nested Graph Neural Networks. In Proceedings of the ICC 2022—IEEE International Conference on Communications, Seoul, Republic of Korea, 16–20 May 2022; pp. 2694–2699.
34. Mohanraj, S.K.; Sivakrishna, A.M. Spatio-Temporal Graph Neural Networks for Enhanced Insider Threat Detection in Organizational Security. *Authorea Prepr.* **2024**. [CrossRef]
35. Zhong, M.; Lin, M.; Zhang, C.; Xu, Z. A survey on graph neural networks for intrusion detection systems: Methods, trends and challenges. *Comput. Secur.* **2024**, *141*, 103821. [CrossRef]
36. Xi, C.; Wang, H.; Wang, X. A novel multi-scale network intrusion detection model with transformer. *Sci. Rep.* **2024**, *14*, 23239. [CrossRef]
37. Manocchio, L.D.; Layeghy, S.; Portmann, M. FlowTransformer: A flexible python framework for flow-based network data analysis. *Softw. Impacts* **2024**, *22*, 100702. [CrossRef]
38. Abdul Kareem, S.; Sachan, R.C.; Malviya, R.K. Neural Transformers for Zero-Day Threat Detection in Real-Time Cyberse curity Network Traffic Analysis. *Int. J. Glob. Innov. Solut. (IJGIS)* **2024**, *14*, 5–7. [CrossRef]
39. Chen, J.; Zhou, H.; Mei, Y.; Adam, G.; Bastian, N.D.; Lan, T. Real-time Network Intrusion Detection via Decision Transformers. *arXiv* **2023**, arXiv:2312.07696. [CrossRef]
40. De Rose, L.; Andresini, G.; Appice, A.; Malerba, D. VINCENT: Cyber-threat detection through vision transformers and knowledge distillation. *Comput. Secur.* **2024**, *144*, 103926. [CrossRef]
41. Kheddar, H. Transformers and large language models for efficient intrusion detection systems: A comprehensive survey. *Inf. Fusion* **2025**, *124*, 103347. [CrossRef]
42. Oh, S.H.; Kim, J.; Nah, J.H.; Park, J. Employing Deep Reinforcement Learning to Cyber-Attack Simulation for Enhancing Cybersecurity. *Electronics* **2024**, *13*, 555. [CrossRef]

43. Yang, W.; Acuto, A.; Zhou, Y.; Wojtczak, D. A Survey for Deep Reinforcement Learning Based Network Intrusion Detection. *arXiv* **2024**, arXiv:2410.07612.
44. Ren, K.; Zeng, Y.; Cao, Z.; Zhang, Y. ID-RDRL: A deep reinforcement learning-based feature selection intrusion detection model. *Sci. Rep.* **2022**, *12*, 15370. [CrossRef]
45. Merzouk, M.A.; Delas, J.; Neal, C.; Cuppens, F.; Boulahia-Cuppens, N.; Yaich, R. Evading Deep Reinforcement Learning-based Network Intrusion Detection with Adversarial Attacks. In Proceedings of the 17th International Conference on Availability, Reliability and Security, Vienna, Austria, 23–26 August 2022; p. 31.
46. Ren, K.; Zeng, Y.; Zhong, Y.; Sheng, B.; Zhang, Y. MAFSIDS: A reinforcement learning-based intrusion detection model for multi-agent feature selection networks. *J. Big Data* **2023**, *10*, 137. [CrossRef]
47. Issa, M.M.; Aljanabi, M.; Muhialdeen, H.M. Systematic literature review on intrusion detection systems: Research trends, algorithms, methods, datasets, and limitations. *J. Intell. Syst.* **2024**, *33*, 20230248. [CrossRef]
48. MITRE ATT&CK. Enterprise Tactics. Available online: https://attack.mitre.org/tactics/enterprise/ (accessed on 22 October 2025).
49. Hutchins, E.M.; Cloppert, M.J.; Amin, R.M. Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains. Available online: https://www.lockheedmartin.com/content/dam/lockheed-martin/rms/documents/cyber/LM-White-Paper-Intel-Driven-Defense.pdf (accessed on 22 October 2025).
50. Page, M.J.; McKenzie, J.E.; Bossuyt, P.M.; Boutron, I.; Hoffmann, T.C.; Mulrow, C.D.; Shamseer, L.; Tetzlaff, J.M.; Akl, E.A.; Brennan, S.E.; et al. The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. *BMJ* **2021**, *372*, n71. [CrossRef] [PubMed]
51. Kitchenham, B.; Charters, S. Guidelines for Performing Systematic Literature Reviews in Software Engineering. 2007. Available online: https://legacyfileshare.elsevier.com/promis_misc/525444systematicreviewsguide.pdf (accessed on 22 October 2025).
52. casp-uk. CASP Checklists. Available online: https://casp-uk.net/casp-tools-checklists/ (accessed on 22 October 2025).
53. Zhao, Z.; Li, Z.; Li, T.; Zhang, F. A Large-Scale P2P Botnet Detection Framework via Topology and Traffic Co-Verification. In Proceedings of the Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks Workshops, Phoenix, AZ, USA, 2–4 December 2024.
54. Gonçalves, L.; Zanchettin, C. Detecting abnormal logins by discovering anomalous links via graph transformers. *Comput. Secur.* **2024**, *144*, 103944. [CrossRef]
55. Gao, P.; Zhang, H.; Wang, M.; Yang, W.; Wei, X.; Lv, Z.; Ma, Z. Deep Temporal Graph Infomax for Imbalanced Insider Threat Detection. *J. Comput. Inf. Syst.* **2025**, *65*, 108–118. [CrossRef]
56. Akpaku, E.; Chen, J.; Ahmed, M.; Agbenyegah, F.K.; Brown-Acquaye, W.L. RAGN: Detecting unknown malicious network traffic using a robust adaptive graph neural network. *Comput. Netw.* **2025**, *262*, 111184. [CrossRef]
57. Cuong, N.H.; Do Xuan, C.; Long, V.T.; Dat, N.D.; Anh, T.Q. A Novel Approach for APT Detection Based on Ensemble Learning Model. *Stat. Anal. Data Min.* **2025**, *18*, e70005. [CrossRef]
58. Ding, Q.; Li, J. AnoGLA: An efficient scheme to improve network anomaly detection. *J. Inf. Secur. Appl.* **2022**, *66*, 103149. [CrossRef]
59. Han, X.; Zhang, M.; Yang, Z. TBA-GNN: A Traffic Behavior Analysis Model with Graph Neural Networks for Malicious Traffic Detection. In Proceedings of the Wireless Artificial Intelligent Computing Systems and Applications: 18th International Conference, WASA 2024, Qindao, China, 21–23 June 2024; pp. 374–386.
60. King, I.J.; Huang, H.H. Euler: Detecting Network Lateral Movement via Scalable Temporal Link Prediction. *ACM Trans. Priv. Secur.* **2023**, *26*, 35. [CrossRef]
61. Kisanga, P.; Woungang, I.; Traore, I.; Carvalho, G.H.S. Network Anomaly Detection Using a Graph Neural Network. In Proceedings of the 2023 International Conference on Computing, Networking and Communications, ICNC, Honolulu, HI, USA, 20–22 February 2023; pp. 61–65.
62. Li, C.; Li, F.; Yu, M.; Guo, Y.; Wen, Y.; Li, Z. Insider Threat Detection Using Generative Adversarial Graph Attention Networks. In Proceedings of the GLOBECOM 2022–2022 IEEE Global Communications Conference, Rio de Janeiro, Brazil, 4–8 December 2022; pp. 2680–2685.
63. Li, Z.; Cheng, X.; Sun, L.; Zhang, J.; Chen, B. A Hierarchical Approach for Advanced Persistent Threat Detection with Attention-Based Graph Neural Networks. *Secur. Commun. Netw.* **2021**, *2021*, 9961342. [CrossRef]
64. Meng, X.; Lang, B.; Yan, Y.; Liu, Y. Deeply fused flow and topology features for botnet detection based on a pretrained GCN. *Comput. Commun.* **2025**, *233*, 108084. [CrossRef]
65. Rithuh Subhakkrith, S.; Kumar, Y.J.; Harish, R.; Dheeraj, B.V.; Vidhya, S. Graph-Based Approaches to Detect Network Anomalies and to Predict Attack Spread. In Proceedings of the 2024 15th International Conference on Computing Communication and Networking Technologies, ICCCNT 2024, Mandi, India, 18–22 June 2024.

66. Ur Rehman, M.; Ahmadi, H.; Ul Hassan, W. Flash: A Comprehensive Approach to Intrusion Detection via Provenance Graph Representation Learning. In Proceedings of the Proceedings—IEEE Symposium on Security and Privacy, San Francisco, CA, USA, 19–23 May 2024; pp. 3552–3570.
67. Wu, G.; Wang, X.; Lu, Q.; Zhang, H. Bot-DM: A dual-modal botnet detection method based on the combination of implicit semantic expression and graphical expression. *Expert Syst. Appl.* **2024**, *248*, 123384. [CrossRef]
68. Wu, Y.; Hu, Y.; Wang, J.; Feng, M.; Dong, A.; Yang, Y. An active learning framework using deep Q-network for zero-day attack detection. *Comput. Secur.* **2024**, *139*, 103713. [CrossRef]
69. Xu, X.; Zheng, X. Hybrid model for network anomaly detection with gradient boosting decision trees and tabtransformer. In Proceedings of the ICASSP 2021–2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Toronto, ON, Canada, 6–11 June 2021; pp. 8538–8542.
70. Yang, Y.; Wang, L. LGANet: Local Graph Attention Network for Peer-to-Peer Botnet Detection. In Proceedings of the Proceedings—2021 3rd International Conference on Advances in Computer Technology, Information Science and Communication, CTISC 2021, Shanghai, China, 23–25 April 2021; pp. 31–36.
71. Zhang, H.; Zhou, Y.; Xu, H.; Shi, J.; Lin, X.; Gao, Y. Graph neural network approach with spatial structure to anomaly detection of network data. *J. Big Data* **2025**, *12*, 105. [CrossRef]
72. Zhang, L.; Shi, H.; Zhang, K.; Sun, H.; Zhang, W. GraphMal: A Network Malicious Traffic Identification Method Based on Graph Neural Network. In Proceedings of the 2023 International Conference on Networking and Network Applications, NaNA 2023, Qingdao, China, 18–21 August 2023; pp. 262–267.
73. Zhao, J.; Liu, X.; Yan, Q.; Li, B.; Shao, M.; Peng, H. Multi-attributed heterogeneous graph convolutional network for bot detection. *Inf. Sci.* **2020**, *537*, 380–393. [CrossRef]
74. Zoubir, A.; Missaoui, B. Graph Neural Networks with scattering transform for network anomaly detection. *Eng. Appl. Artif. Intell.* **2025**, *150*, 110546. [CrossRef]
75. Yang, Y.; Yao, C.; Yang, J.; Yin, K. A Network Security Situation Element Extraction Method Based on Conditional Generative Adversarial Network and Transformer. *IEEE Access* **2022**, *10*, 107416–107430. [CrossRef]
76. Xu, L.; Zhao, Z.; Zhao, D.; Li, X.; Lu, X.; Yan, D. AJSAGE: A intrusion detection scheme based on Jump-Knowledge Connection To GraphSAGE. *Comput. Secur.* **2025**, *150*, 104263. [CrossRef]
77. Düzgün, B.; Çayır, A.; Ünal, U.; Dağ, H. Network intrusion detection system by learning jointly from tabular and text-based features. *Expert Syst* **2024**, *41*, e13518. [CrossRef]
78. Cao, H. The Detection of Abnormal Behavior by Artificial Intelligence Algorithms under Network Security. *IEEE Access* **2024**, *12*, 118605–118617. [CrossRef]
79. Du, W.; He, Y.; Li, G.; Yang, X.; Li, J.; Ren, G.; Zhou, K. HyperTTC: Hypergraph-Empowered Tactic-Specific Traffic Clustering for Atomized APT Detection. In Proceedings of the 2025 International Conference on Computing, Networking and Communications (ICNC), Honolulu, HI, USA, 17–20 February 2025; pp. 318–322.
80. Manocchio, L.D.; Layeghy, S.; Lo, W.W.; Kulatilleke, G.K.; Sarhan, M.; Portmann, M. FlowTransformer: A transformer framework for flow-based network intrusion detection systems. *Expert Syst. Appl.* **2024**, *241*, 122564. [CrossRef]
81. Santosh, R.P.; Chaudhari, T.; Godla, S.R.; Ramesh, J.V.N.; Muniyandy, E.; Smitha, K.A.; Baker El-Ebiary, Y.A. AI-Driven Transformer Frameworks for Real-Time Anomaly Detection in Network Systems. *Int. J. Adv. Comput. Sci. Appl.* **2025**, *16*, 1121–1130. [CrossRef]
82. Akkepalli, S.; Sagar, K. Copula entropy regularization transformer with C2 variational autoencoder and fine-tuned hybrid DL model for network intrusion detection. *Telemat. Inform. Rep.* **2025**, *17*, 100182. [CrossRef]
83. Malik, M.; Saini, K.S. Network Intrusion Detection System Using Reinforcement Learning Techniques. In Proceedings of the International Conference on Circuit Power and Computing Technologies, ICCPCT 2023, Kollam, India, 10–11 August 2023; pp. 1642–1649.
84. Muhati, E.; Rawat, D.B. Asynchronous Advantage Actor-Critic (A3C) Learning for Cognitive Network Security. In Proceedings of the Proceedings—2021 3rd IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications, TPS-ISA 2021, Atlanta, GA, USA, 13–15 December 2021; pp. 106–113.
85. Luna, L.; Berkowitz, M.P.; Kandel, L.N.; Jansen-Sánchez, S. Dueling Deep Q-Learning for Intrusion Detection. In Proceedings of the Conference Proceedings—IEEE SOUTHEASTCON, Concord, NC, USA, 22–30 March 2025; pp. 1192–1197.
86. Sharafaldin, I.; Habibi Lashkari, A.; Ghorbani, A.A. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In Proceedings of the International Conference on Information Systems Security and Privacy, Funchal, Portugal, 22–24 January 2018.

87. Moustafa, N.; Slay, J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS), Canberra, Australia, 10–12 November 2015; pp. 1–6.
88. Glasser, J.; Lindauer, B. Bridging the Gap: A Pragmatic Approach to Generating Insider Threat Data. In Proceedings of the 2013 IEEE Security and Privacy Workshops, San Francisco, CA, USA, 23–24 May 2013; pp. 98–104.