Rethinking Graph-Based Approaches: An Empirical Study of Feature Engineering in Network Intrusion Detection

Richard Chukwuebuka Nwachukwu
Department Natural and Applied Sciences, Ignatius Ajuru University of Education, Port Harcourt Nigeria
* Corresponding author : iamrichardcn@gmail.com

Abstract

Although graph-based feature engineering has become widely used in network intrusion detection systems (NIDS), there is a severe lack of empirical research on determining whether the addition of network topology features results in a real positive improvement over the operation of the system or it simply adds complexity to the system. Our paper gets into this gap by critically assessing the performance of graph-based methods as compared to conventional statistical features via systematic comparative analysis across several machine learning paradigms. Using the UNSW-NB15 dataset, we employed a graph-theoretic characteristics that included measures of centrality, the community structure identification and the topological analysis, which were compared to traditional traffic-based characteristics. Results revealed a counterintuitive finding where incorporating graph features consistently degraded detection performance across all algorithms, with statistically significant accuracy reductions observed in multiple classifiers. Random Forest experienced modest performance decline, while Support Vector Machines and RBF Networks showed more substantial degradation. Computational analysis also demonstrated that graph feature extraction imposed substantial overhead compared to traditional feature computation, representing approximately nineteen-fold increase in processing time.

Keywords- Intrusion Detection, Graph Theory, Network Security, Machine Learning, Feature Engineering, Cybersecurity

1. INTRODUCTION

The exponential growth in cyber threats has necessitated continuous evolution in network intrusion detection systems (NIDS), driving researchers toward increasingly sophisticated feature engineering approaches. Traditional NIDS rely primarily on statistical features extracted from network traffic flows, including temporal characteristics (connection duration, inter-arrival times), volumetric measures (packet counts, byte volumes), and behavioral indicators (protocol distributions, service usage patterns) [1,2]. On the contrary, the cybersecurity community is progressively incorporating graph-theoretic techniques in their research, under the hypothesis that network topology and communication patterns add further discriminative value for malicious traffic versus benign traffic[3,4,5].

Graph-based intrusion detection views network communications as mathematical structures G = (V, E), whereby vertices V denote hosts and edges E stand for connections, allowing complex structural feature extraction, such as centrality measures, clustering coefficients, and community detection metrics. The fundamental assumption underpinning this line of research posits that attackers differ in communication patterns that translate into discernible graph-theoretic signatures [6,7]. Several studies, consequently, report an uplift in performance when graph features are used alongside machine learning models for intrusion detection [8,9,10].

However, this prevailing narrative suffers from critical limitations. First, the assumption that simply adding more features to a system does not guarantee its improvement goes against general principles of machine learning that speak of the curse of dimensionality and feature relevance [11]. Second, the computational overhead and the complexity of implementing graph-based approaches have raised very serious questions regarding their real-world utility, especially in deployment scenarios where latency and scalability are of utmost importance. Third, publication bias might have kept the literature thriving on positive results without going into critical discussions about conditions where graph features are to the detriment.

DOI : https://doi.org/10.21063/jtif.2026.V14.1.1-10

This study addresses these critical gaps through systematic empirical investigation designed to challenge conventional assumptions about graph-based intrusion detection effectiveness. Our research objectives are thus:

(1) Conduct comprehensive performance comparison between traditional traffic-based features and graph enhanced feature sets across multiple machine learning paradigms

(2) Quantify computational overhead and scalability implications of graph-based approaches

The primary contributions of this work are threefold: (1) empirical evidence challenging the assumed superiority of graph-based intrusion detection approaches, (2) a comprehensive comparative framework evaluating traditional versus graph-enhanced features across multiple algorithms, and (3) practical insights regarding the trade-offs between feature complexity and detection performance in network security applications.

2. METHODS AND DATA

2.1. Experimental Design Framework

Our experimental design is based on a strict comparative system that aims to isolate the effect of graph features on the performance of intrusion detection. Figure 1 represents the experimental pipeline and it involves four main steps, namely, data preprocessing, feature extraction, model training, and performance evaluation. The pipeline initiates with raw network flow data of the UNSW-NB15 set, and the subsequent parallel generation of feature extraction directions of classic traffic-based features and graph-theoretic features. These sets of features are used separately and together so that the performance of any set can be compared directly to the other one. The experimental architecture uses systematic influences to ensure that varying performance based on feature composition and not on the variation in algorithms or implementations is achieved. The quality assurance mechanisms are included in each of the stages such as data validation, feature normalization as well as cross-validation protocols that guarantee the level of reproducibility and statistical rigor within the entire experimental process.

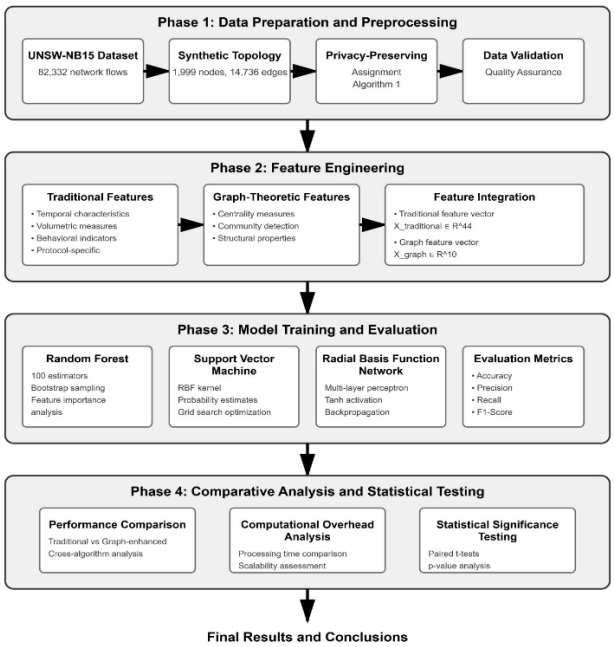

Fig. 1. Experimental Pipeline Architecture

2.2. Dataset Characteristics and Preprocessing

DOI : https://doi.org/10.21063/jtif.2026.V14.1.1-10

UNSW-NB15 data contains records of network flows obtained in real-world networks, and it is one of the first improvements of former datasets in terms of relevance in modern context and diversity of attacks [12,13]. Our experiment sample retains the original class distribution with reasonable representation of the normal and attack traffic flows. The data offers a rich list of original characteristics of features that cover various characteristics of traffic.

Time dynamics Temporal properties measure time varying aspects such as connection time and inter-arrival time, which characterize the time dynamics of network interactions. Volumetric measures are used to measure data transfer patterns in terms of number of packets (source packets and destination packets) and number of bytes (source packets and destination packets), which can give an insight into the intensity of the traffic and communication patterns. Behavioral indicators include transmission rates and load characteristics (source load, destination load), the behavior of network entities. Protocol-specific properties are TCP flags and window sizes and type of service, which contain protocol-level protocol details needed to detect attacks based on a protocol. The loss rates, jitter measurements (Quality of Service parameters that indicate the changes in the delay of packets), and transitions in the state are statistical aggregates and these are all used to describe the metrics of network reliability and performance. Loss rates are used to measure the number of drops of packets that may indicate that the network is overloaded or faces malicious interference, jitter measures the deviation of the time the packets arrive, which is one of the most important Quality of Service (QoS) parameters to identify abnormal timing patterns that can be used to signal an attack such as timing-based reconnaissance or traffic manipulation.

Traditional feature-vector mathematically represented:

$$X_{traditional} = [dur, spkts, dpkts, sbytes, dbytes, rate, sload, dload, \ldots]^T \in R^{19} \quad (1)$$

2.3. Synthetic Network Topology Generation

To overcome the privacy issues and still achieve the research reproducibility, we used a synthetic topology generation procedure which is specifically based on generating realistic and privacy preserving network representations. It is done through the systematic assigning of IP addresses, in which observed patterns of service and communication practices are used in the original dataset. This methodology has two purposes: to safeguard the privacy of original network entities and to produce topologically meaningful graphs that can be analyzed using graph methods.

Synthetic topology generation algorithm is based on intelligent IP address mapping which maintains the structural properties of network communications. Hosts in the internal network are given an IP address based on a specified range within a particular private address space, whereas external parties are allocated an address based on a different range, and the conceptual separation between internal and external network zones is preserved. Given that hosts offering similar services are likely to be in the same coherent subnets, service-based assignment is used to provide the hosts with addresses in reasonable network structures. Consistency maintenance ensures that different flows with the same logical host make repeated references to the same synthetic IP address, which keeps the temporal and relational structure of network communications. A synthetic network topology is created using this method and these networks preserve statistical and structural characteristics of real enterprise networks but does not leave a record of the real network infrastructure.

Algorithm 1: Synthetic IP Generation.
Input: Network flows F, Service patterns S Output: Synthetic topology T with IP assignments
(1) Initialize IP pools:
   a. Internal_IPs ← $\{192.168.x.y \mid x \in [1,255], y \in [1,254]\}$
   b. External_IPs ← $\{10.a.b.c \mid a \in [1,255], b,c \in [1,254]\}$
(2) For each flow $f \in F$:

DOI : https://doi.org/10.21063/jtif.2026.V14.1.1-10

a. If f.service ∈ {'dns', 'http', 'smtp', 'ftp'}:
   - src_ip ← sample(Internal_IPs[1:100]) – dst_ip ← sample(External_IPs)
b. Else:
   - src_ip ← sample(Internal_IPs)
   - dst_ip ← sample(Internal_IPs)

3. Maintain consistency across multiple records
4. Return topology T with synthetic IP assignments

Network Topology Properties. The generated synthetic network exhibits realistic topological characteristics:

a. Nodes: $|V| = 1,999$ unique IP addresses
b. Edges: $|E|$ = 14, 736 connections
c. Density: $\rho = 2|E|/(|V|(|V| - 1))$ = 0.0074
d. Average Path Length: 2.993
e. Diameter: 5 hops

The algorithm structure makes sure that the services that are likely to be used by the users like web browsing, secure communications, domain name resolution, and secure shell connections represent internal-to-external communication patterns as well as the other services represent the internal network communications. This separation gives a realistic topology structure with some services being inherently more central (e.g. DNS servers, web proxies) and others being peer-to-peer or hierarchical internal communication. Network Topology properties establish that the synthetic topology models key structural properties of realistic network settings, which forms a viable substrate to do graph-theoretic feature extraction.

2.4. Graph-Theoretic Feature Extraction

Graph feature extraction entails building of a network topology using traffic flow data and computing the graph-theoretic measures of each network node (IP address). We assume that the network is a directed graph G, (V, E), each of whose vertices, V, is a distinct IP address and whose edges, E, are communication paths between hosts.

Centrality Measures: We compute multiple centrality metrics:

a. Degree Centrality: $CD(v) = \deg(v)/(|V| - 1)$
b. Betweenness Centrality: $CB(v) = Ís \neq v \neq t\, \sigma st(v)/\sigma st$
c. Closeness Centrality: $CC(v) = (|V| - 1)/Íu \in V\, d(v, u)$
d. Eigenvector Centrality: Principal eigenvector component

Community Structure: We use the Louvain algorithm [23] in order to identify community structures of the network, which uncovers groups of nodes that have tight internal ties and loose external ties. The hierarchical clustering method maximizes network modularity and finds structures of the organization that could be associated with functional network segregations or attack patterns.

The complete graph feature vector for each communication flow includes features for both source and destination IP addresses:

$$\text{xgraph} = [\text{xtraditional}; CD(src); CB(src); CC(src); \ldots; CD(dst); CB(dst); \text{CC (dst)}; \ldots]\text{T} \in \text{Rn} + \text{m} \quad (2)$$

2.5 Machine Learning Algorithms

We consider three different machine learning methods representing the following algorithmic paradigms, Radial Basis Function Networks (RBFN): RBFN This is implemented on Multi-Layer Perceptrons based on hyperbolic tangent activation functions, which give smooth nonlinear decision boundaries through

DOI : https://doi.org/10.21063/jtif.2026.V14.1.1-10

radial basis transformations [20-22], Random Forest: Multiple decision tree estimators are used in an ensemble method which takes advantage of bootstrap aggregation to shrink variance and enhance generalization by using varying tree structures [16], Support vector machines: Probability estimates based, kernel-based classifiers (Radial Basis Function kernel) with Radial Basis Function kernels, which build the best separating hyperplane in high-dimensional (feature) space [19].

3. RESULTS AND DISCUSSION

Analysis of network topology shows that there are distinct IP addresses that can be depicted as a graph of connection where there are many edges and low density similar to a real enterprise network. The network is small-world with small average path lengths and small diameter, which are typical of real-world communication networks. Community detection finds many individual communities whose modularity scores are large, meaning that there is a high degree of community organization that is characteristic of functional organization of enterprise networks [23].

In Table 1, there is significant asymmetric feature of the graphs between the normal and attack traffic analyzed statistically. The centrality on source is consistent with rise in the attack cases whereas destination-side aspects indicate negative correlations. Particularly, the attackers are mostly located at the highly central positions in the network as they are characterized by significant changes in the degree centrality of sources and betweenness centrality between the attack periods [2,5,10]. This trend is an indication that malicious actors place themselves in strategic points of network bottlenecks to ensure they can reach and operate as many points of attack as possible.

Table 1. Graph Feature Comparison by Traffic Type

Feature	Normal Mean	Attack Mean	Difference	Ratio	Correlation with Label
Degree centrality src	0.018	0.0264	+0.0081	1.44	+0.273
Betweenness centrality src	0.003	0.0067	+0.0028	1.70	+0.265
Eigenvector centrality src	0.018	0.0267	+0.0079	1.42	+0.268
Pagerank src	0.001	0.0018	+0.0006	1.48	+0.218
Degree centrality dst	0.010	0.0075	-0.0027	0.73	-0.164
Betweenness centrality dst	0.001	0.0010	-0.0005	0.67	-0.098
Eigenvector centrality dst	0.019	0.0186	-0.0007	0.96	-0.047
Pagerank dst	0.000	0.0005	-0.0002	0.75	-0.112

On the other hand, destination-side centrality measures do not decrease systematically in the course of attacks, and there are significant losses in both degree centrality and betweenness centrality. The trend shows that the attackers give preference to peripheral nodes which are less connected and are probably vulnerable endpoints with fewer monitoring and defending measures [2,14,29].

3.1. Comprehensive Performance Evaluation

The overall analysis demonstrates some mind-shifting trends that contradict traditional beliefs regarding the efficacy of graph features, which is in Table 2. Traditional approaches consistently outperform or match graph-enhanced variants across all algorithms and metrics. Random Forest proves to be stable in its performance, with a slight deterioration in accuracy. The effect of addition of graph features on the Support Vector Machines is more pronounced in loss of performance. Radial Basis Function Networks

DOI : https://doi.org/10.21063/jtif.2026.V14.1.1-10

have the greatest degradation with the accuracy tending to fall significantly when graph features are added.

Table 2. Comprehensive Performance Comparison

Algorithm	Feature Set	Accuracy	Precision	Recall	F1-Score	Performance Change
RBFN	Traditional	0.91	0.952	0.896	0.923	-
RBFN	Enhanced	0.84	0.859	0.865	0.862	-7.09%
Random Forest	Traditional	0.934	0.954	0.925	0.940	-
Random Forest	Enhanced	0.926	0.955	0.909	0.931	-0.82%
SVM	Traditional	0.860	0.898	0.842	0.869	-
SVM	Enhanced	0.842	0.871	0.837	0.853	-1.87%

The break-even analysis, done in detail using metrics, discloses the consistency in the values of the precision, recall and F1-score. Models with graphs tend to be less accurate which means that they have higher false positive rates and hence will overload security analysts with a lot of false alarms during operational deployments. Recall metrics also reduce in the same direction as graph enhanced variants, indicating that they are not able to detect real attacks. F1-score which is the harmonic mean of recall and precision always gives preference to traditional feature approaches in all three algorithms used [28,30].

3.2. Computational Overhead Analysis

Graph feature extraction adds significant computational load over a traditional feature computation, as quantified in Table 3. It is done through topology building of networks, calculation of centrality measures and community detection operations with linear to cubic complexity based on the measure.

Table 3. Computational Performance Metrics

Process	Traditional Features	Graph-Enhanced Features
Feature Extraction	0.8 seconds	15.3 seconds
Overhead Factor	-	19.1×

Our implementation took a significantly longer time to construct the graph and extract features within the experimental dataset than preparing features using the traditional method. This is a computational cost that is a serious practical issue to real-time intrusion detection systems in which latency constraints are severe.

This computation statistical burden is especially troublesome in the context of high-throughput network environment where network systems are required to serve thousands of millions of flows per second. Graph construction involves keeping and updating network topology in real-time, the computation of centrality scales poorly with network size and detection of communities in a network involves the use of iterative optimization.

3.3. Discussion of Findings

The experimental results challenge the prevailing assumption that graph features automatically improve intrusion detection performance. Several factors contribute to this counterintuitive finding. Graph characteristics can record network topological properties which are perpendicular to malicious behaviour patterns. While attackers do exhibit communication patterns, these may be better captured by traditional traffic statistics rather than global network topology metrics [2,14,27]. The time and volume properties

DOI : https://doi.org/10.21063/jtif.2026.V14.1.1-10

of the attack traffic, which are recorded in the form of the number of packets, the number of bytes and the length of a connection, are an overture to malicious activity. Conversely, graph-theoretic measures are higher-order abstractions that can even degenerate the discriminative signal [7,15,28].

From an information theory viewpoint, adding graph feature to the feature space raises the entropy of the feature space without correspondingly boosting the mutual information with the target variable [14,26]. The dimensionality curse occurs when the extra features do not have any discriminatory value, and machine learning methods have problems determining the patterns of relevance in the larger feature space. It is especially problematic when graph observables are noisy and not signal-like and algorithms are fitting spurious correlations between features instead of real indicators of an attack [24,26,28].

The topology generation process of synthetic networks, as much as required in order to protect the privacy, can unknowingly add in artificial patterns that will confuse the learning algorithms. The graph characteristics calculated using artificial topologies can be inaccurate with the actual structural characteristic of real network communications [27]. Real networks have organic topology formation, management rules and behavioral patterns that influence real-world networks and topology in a manner that is hard to imitate by synthetical formation. As a result, the features of a graph obtained using synthetic topologies can be the artifact of the generation process instead of the actual structure-related attacks.

The asymmetric features of the centrality measures as observed are valuable in the attack strategies and vulnerability of the networks. The attackers who have a central position have innate reconnaissance, cross-cutting, and extensive effects. High betweenness centrality represents being placed on a large number of shortest paths, allowing enemies to intercept, manipulate, or monitor large amounts of network traffic [2,10,29]. On the other hand, the strategy of attacking peripheral nodes is consistent with the familiar patterns of attacks that target endpoints with low levels of security monitoring, including workstations and Internet of Things devices or older systems that do not have modern security controls [29,30].

3.4. Practical Considerations for Deployment

The results have far reaching implications on the realistic application of intrusion detection system in the operational fields. Conventional feature methods have high computational benefits, they take considerably less time and memory than graph-based methods. This is an efficiency saving that is critical in high throughput network situations where real time processing forms a fundamental need [17,24,25]. Millions of flows in network environments make the computational cost of graph construction and feature extraction unfeasible. The nineteenfold or so direct proportional change in processing time results in an increase in latency, a decrease in throughput or a proportional increase in the necessary powerful hardware infrastructure. When resources are limited to the operations of an organization or when the organization has to handle large volumes of traffic, the conventional methods offer more viable and cost-efficient solutions [25,26].

3.5 Limitations and Future Work

Several limitations affect the generalizability of our findings:
Dataset Specificity: Results are based on the UNSW-NB15 dataset, which may not represent all network environments or attack patterns. Future work should evaluate similar hypotheses across multiple datasets and network architectures.
Synthetic Topology Impact: The use of synthetic network topologies, while necessary for privacy protection, may have influenced graph feature quality. Real network topologies might yield different results.
Algorithm Selection: Our evaluation focuses on three representative algorithms. Additional algorithms, particularly deep learning approaches and specialized graph neural networks, warrant investigation.
Future research directions include developing principled feature selection methods for intrusion detection, investigating the conditions under which graph features provide value, and exploring hybrid approaches that selectively incorporate relevant graph metrics.

DOI : https://doi.org/10.21063/jtif.2026.V14.1.1-10

4. CONCLUSION

This study questions one of the core assumptions in modern cybersecurity research and practice by conducting strict empirical research on the topic of graph-based network intrusion detection system feature engineering. A thorough comparison of three different machine learning paradigms which are, one, Random Forest, two, Support Vector Machines, and three, Radial Basis Function Networks, all shows a tendency of consistent evidence that adding graph-theoretic features does not only not improve detection performance but in fact, improves the performance of all algorithms tested.

With regard to our first research objective, the end-to-end performance analysis of traditional traffic characteristics and graph-enhanced characteristics sets reveals the evident high-performance of standard methods. All of traditional statistical properties based on network flows, such as time-related properties, volume metrics, and behavioral properties, are always significantly superior to graph-augmented counterparts in all conventional classification measures. Random Forest classifiers also have relatively small, yet statistically significant accuracy losses with the addition of graph features. Support Vector Machines demonstrate greater performance deterioration. Most radically, Radial Basis Function Networks show grave degradation of accuracy, where differences become statistically significant. This finding is consistent in terms of precision, recall, and F1-score values, meaning that the performance deterioration is reflected in both false negative and false positive values, posing operational difficulties to the deployed systems.

In terms of the second research goal, computational overhead quantification shows that graph-based methods have large and practically infeasible computational burdens. Extraction of graph features takes about nineteen times as long as the traditional features to compute, and this is a basic scaling bottleneck of real-time intrusion detection. This computation overhead is due to several sources such as network topology construction, centrality measure computation as well as community detection algorithms. This overhead, directly proportional to the throughput of the network environment measuring in millions of flows/second, directly leads to system capacity limitations, a higher latency, or much higher power and cost requirements in hardware infrastructure. Scalability is not just a matter of processing time but also memory usage, complexity of algorithm and maintenance cost which, all combined, makes graph-based solutions not very practical in practice when it comes to operational deployment.

Its consequences are not limited to cybersecurity, but to the machine learning community as a whole, by pointing to the paramount importance of empirical validation of intuitively appealing yet ultimately unproductive methods. The study shows that the topological complexity is not necessarily proportional to the detection ability. Rather, when feature complexity is increased but there is no corresponding increase in the discriminative value, it results in a deteriorated performance due to the curse of dimensionality, addition of noise, instead of signal, and computational costs that far exceed any possible gains.

ACKNOWLEDGEMENTS

The author acknowledges the creators of the UNSW-NB15 dataset for making this comprehensive network intrusion dataset publicly available for research purposes. The author also thanks the developers of the open-source machine learning libraries that made this comparative study possible.

REFERENCES

[1] D. E. Denning, "An intrusion-detection model," IEEE Transactions on Software Engineering, vol. SE-13, no. 2, pp. 222-232, Feb. 1987. http://dx.doi.org/10.1109/TSE.1987.232894

[2] T. Bilot, N. El Madhoun, K. Al Agha, and A. Zouaoui, "Graph neural networks for intrusion detection: A survey," IEEE Access, vol. 11, pp. 49114-49139, 2023. http://dx.doi.org/10.1109/ACCESS.2023.3275789

[3] W. Lo, S. Layeghy, M. Sarhan, M. Gallagher, and M. Portmann, "E-GraphSAGE: A graph neural network based intrusion detection system for IoT," in Proc. NOMS 2022–2022 IEEE/IFIP Network Operations and Management Symposium, 2022, pp. 1-9. http://dx.doi.org/10.1109/NOMS54207.2022.9789878
[4] D. H. Tran and M. Park, "Graph embedding for graph neural network in intrusion detection system," in Proc. International Conference on Information Networking (ICOIN), 2024, pp. 1-6. http://dx.doi.org/10.1109/ICOIN59985.2024.10572048
[5] D. Pujol-Perich, J. Suarez-Varela, M. Ferriol-Galmes, S. Xiao, B. Cabellos-Aparicio, P. Barlet-Ros, and A. Cabellos-Aparicio, "Unveiling the potential of graph neural networks for robust intrusion detection," ACM SIGMETRICS Performance Evaluation Review, vol. 49, no. 4, pp. 111-117, 2022. http://dx.doi.org/10.1145/3543146.3543171
[6] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," Computers & Security, vol. 31, no. 3, pp. 357-374, 2012. http://dx.doi.org/10.1016/j.cose.2011.12.012
[7] Y. K. Saheed, A. A. Usman, F. D. Sukat, and M. Abdulrahman, "A novel hybrid autoencoder and modified particle swarm optimization feature selection for intrusion detection in the internet of things network," Frontiers in Computer Science, vol. 5, p. 997159, 2023. http://dx.doi.org/10.3389/fcomp.2023.997159
[8] S. Axelsson, "Intrusion detection systems: A survey and taxonomy," Department of Computer Engineering, Chalmers University of Technology, Tech. Rep. 99-15, 2000. Available at: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=7a15948bdcb530e2c1deedd8d22dd9b54788a634
[9] H. Debar, M. Dacier, and A. Wespi, "Towards a taxonomy of intrusion-detection systems," Computer Networks, vol. 31, no. 8, pp. 805-822, 1999. http://dx.doi.org/10.1016/S1389-1286(98)00017-6
[10] M. Anoop, L. W. Mary, A. J. Wilson, et al., "Optimized graph transformer with molecule attention network based multi class attack detection framework for enhancing privacy and security in WSN," Multimedia Tools and Applications, vol. 83, pp. 1-32, 2024. http://dx.doi.org/10.1007/s11042-024-19516-x
[11] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in Proc. IEEE Symp. Computational Intelligence for Security and Defense Applications, 2009, pp. 1-6. http://dx.doi.org/10.1109/CISDA.2009.5356528
[12] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in Proc. Military Communications and Information Systems Conference, 2015, pp. 1-6.
[13] N. Moustafa and J. Slay, "The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," Information Security Journal: A Global Perspective, vol. 25, no. 1-3, pp. 18-31, 2016. http://dx.doi.org/10.1080/19393555.2015.1125974
[14] A. M. Alsaffar, M. Nouri-Baygi, and H. M. Zolbanin, "Shielding networks: Enhancing intrusion detection with hybrid feature selection and stack ensemble learning," Journal of Big Data, vol. 11, no. 1, p. 133, 2024. http://dx.doi.org/10.1186/s40537-024-00994-7
[15] [15] U. Ahmed, Z. Jiangbin, A. Almogren, M. Sadiq, A. U. Rehman, M. T. Sadiq, and J. Choi, "Hybrid bagging and boosting with SHAP based feature selection for enhanced predictive modeling in intrusion detection systems," Scientific Reports, vol. 14, no. 1, p. 30532, 2024. http://dx.doi.org/10.1038/s41598-024-81151-1
[16] L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5-32, 2001. http://dx.doi.org/10.1023/A:1010933404324
[17] M. Nanjappan, K. Pradeep, G. Natesan, A. Samydurai, and G. Premalatha, "DeepLG SecNet: Utilizing deep LSTM and GRU with secure network for enhanced intrusion detection in IoT environments," Cluster Computing, pp. 1-13, 2024. http://dx.doi.org/10.1007/s10586-024-04503-4
[18] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," Annals of Statistics, vol. 29, no. 5, pp. 1189-1232, 2001. http://dx.doi.org/10.1214/aos/1013203451
[19] B. Schölkopf and A. J. Smola, "Learning with kernels: Support vector machines, regularization, optimization, and beyond," MIT Press, 2002. http://dx.doi.org/10.7551/mitpress/4175.001.0001

DOI : https://doi.org/10.21063/jtif.2026.V14.1.1-10

[20] A. Heidari, N. J. Navimipour, and M. Unal, "A secure intrusion detection platform using blockchain and radial basis function neural networks for internet of drones," IEEE Internet of Things Journal, vol. 10, no. 10, pp. 8445-8454, 2023. http://dx.doi.org/10.1109/JIOT.2023.3237661
[21] S. Haykin, "Neural networks: A comprehensive foundation," 2nd ed., Prentice Hall, 1999.
[22] M. J. D. Powell, "Radial basis functions for multivariable interpolation: A review," in Algorithms for Approximation, Oxford University Press, 1987, pp. 143-167.
[23] V. D. Blondel, J. L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," Journal of Statistical Mechanics: Theory and Experiment, vol. 2008, no. 10, p. P10008, 2008. http://dx.doi.org/10.1088/1742-5468/2008/10/P10008
[24] Y. K. Saheed, O. H. Kayode, Abdulganiyu, and Taha Ait Tchakoucht, "Modified genetic algorithm and fine-tuned long short-term memory network for intrusion detection in the internet of things networks with edge capabilities," Applied Soft Computing, vol. 155, p. 111434, 2024. http://dx.doi.org/10.1016/j.asoc.2024.111434
[25] H. Hazman, C. Guezzaz, A. Benkirane, S. et al., "Enhanced IDS with deep learning for IoT-based smart cities security," Tsinghua Science and Technology, vol. 29, no. 4, pp. 929-947, 2024. http://dx.doi.org/10.26599/TST.2023.9010033
[26] N. Nath, "A generalized lightweight intrusion detection model with unified feature selection for internet of things networks," International Journal of Network Management, vol. 34, no. 4, p. e2291, 2024. http://dx.doi.org/10.1002/nem.2291
[27] S. M. Kasongo and Y. Sun, "Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset," Journal of Big Data, vol. 7, no. 1, p. 105, 2020. http://dx.doi.org/10.1186/s40537-020-00379-6
[28] A. M. Almotairi, S. Atawneh, and A. Osama, "Enhancing intrusion detection in IoT networks using machine learning-based feature selection and ensemble models," Systems Science & Control Engineering, vol. 12, no. 1, p. 2321381, 2024. http://dx.doi.org/10.1080/21642583.2024.2321381
[29] A. Qaddos, M. A. Jan, M. Hussain, M. Mastorakis, and C. X. Mavromoustakis, "A novel intrusion detection framework for optimizing IoT security," Scientific Reports, vol. 14, p. 21789, 2024. http://dx.doi.org/10.1038/s41598-024-72733-0
[30] M. Farhan, H. W. ud din, S. Ullah, and M. S. Hussain, "Network-based intrusion detection using deep learning technique," Scientific Reports, vol. 15, p. 8770, 2025. http://dx.doi.org/10.1038/s41598-025-08770-0

DOI : https://doi.org/10.21063/jtif.2026.V14.1.1-10