iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** VIII **Month of publication:** August 2026

DOI: https://doi.org/10.22214/ijraset.2026.84513

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue VIII Aug 2026- Available at www.ijraset.com

A Comparative Evaluation of Deep Learning Architectures for Binary Network Intrusion Detection Using the NSL-KDD Dataset

Ketki Naik[1], Sanjeev Ghosh[2]
[1]*Department of Electronics and Telecommunication Engineering,* [2] *Department of Computer Science and Engineering (IoT),* [1,2]
Thakur College of Engineering and Technology, Mumbai, India, Email: sanjeevnghosh@gmail.com

Abstract: ***The rapid growth of digital communication technologies, cloud computing, and Internet of Things (IoT) devices has increased both the frequency and sophistication of cyber-attacks, making effective intrusion detection an essential component of modern cybersecurity systems. Traditional signature-based intrusion detection systems (IDS) are effective against known attacks but fail to detect previously unseen or evolving threats. This study investigates the application of deep learning models for binary network intrusion detection using the NSL-KDD benchmark dataset. Three standalone architectures, Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM) networks, and Deep Neural Networks (DNN), are implemented and evaluated, alongside a CNN-LSTM Hybrid model that integrates spatial and sequential learning, and a DNN-LSTM Ensemble model that combines independently trained DNN and LSTM predictions through weighted averaging. Following data cleaning, categorical encoding, normalization, and Random Forest-based feature selection (41 features reduced to 20), all models were trained and evaluated under identical conditions using Accuracy, Precision, Recall, F1-Score, ROC-AUC, training time, and inference time. The standalone DNN model achieved the best overall performance, with 80.98% accuracy, 97.08% precision, 68.66% recall, 80.43% F1-score, and 96.11% ROC-AUC, while also requiring the shortest training time (39.69 s). The CNN-LSTM Hybrid model attained the highest precision (97.23%) but did not outperform the standalone architectures overall, and the DNN-LSTM Ensemble produced balanced but not superior results. These findings indicate that carefully designed standalone architectures can match or exceed the performance of more complex hybrid and ensemble models for binary intrusion detection, while incurring substantially lower computational cost. The study contributes a controlled, common-framework comparison of five deep learning architectures and provides practical guidance for selecting computationally efficient models for anomaly-based intrusion detection.***

Keywords: Intrusion Detection System; Deep Learning; NSL-KDD; CNN; LSTM; DNN; Hybrid Model; Ensemble Learning; Anomaly Detection; Cybersecurity.

I. INTRODUCTION

The widespread adoption of cloud computing, IoT devices, mobile applications, online banking, and smart infrastructure has substantially increased the volume and complexity of network traffic, expanding the attack surface available to cybercriminals. Threats such as Distributed Denial-of-Service (DDoS) attacks, malware, ransomware, phishing, botnets, insider attacks, and advanced persistent threats (APTs) can compromise the confidentiality, integrity, and availability of information systems, resulting in financial loss, reputational damage, and disruption of critical services.

Intrusion Detection Systems (IDS) play a central role in identifying malicious network activity. Traditional IDS approaches are broadly categorized as signature-based, which match traffic against known attack patterns, or anomaly-based, which model normal behavior and flag deviations. Signature-based systems provide high accuracy for known attacks but cannot detect zero-day or evolving threats, motivating growing interest in machine learning and, more recently, deep learning approaches capable of automatically learning complex, non-linear traffic patterns.

Deep learning architectures such as CNNs, RNNs, LSTMs, and Autoencoders have been successfully applied to intrusion detection. However, most existing studies rely on a single architecture evaluated under study-specific preprocessing and feature selection choices, making cross-study comparison difficult and leaving open the question of whether architectural complexity (e.g., hybrid or ensemble models) reliably improves detection performance. This paper addresses that gap through a controlled comparative study of five deep learning architectures, CNN, LSTM, DNN, a CNN-LSTM Hybrid, and a DNN-LSTM Ensemble, trained and evaluated under an identical experimental framework on the NSL-KDD benchmark dataset.

The remainder of this paper is organized as follows. Section 2 reviews related work on signature-based, machine learning-based, and deep learning-based intrusion detection. Section 3 describes the proposed methodology, including preprocessing, feature selection, and model architectures. Section 4 details the experimental setup. Section 5 presents and discusses the results. Section 6 concludes the paper and outlines directions for future work.

II. RELATED WORK

A. Signature-Based and Traditional IDS

Signature-based systems such as Snort and Suricata detect attacks by matching observed traffic against predefined rule sets. Khairi et al. [1] demonstrated the effectiveness of a Snort-based IDS for securing an e-voting environment, and a related study in Applied Sciences [2] reported satisfactory detection of web attacks using signature-based methods. Despite reliable detection of known threats, these systems cannot identify previously unseen attacks and require continuous signature maintenance, limiting their adaptability to the evolving threat landscape.

B. Machine Learning and Deep Learning-Based IDS

Machine learning algorithms such as Support Vector Machines, Random Forests, Decision Trees, and K-Nearest Neighbors have improved detection over rule-based systems but typically depend on handcrafted feature engineering [3]. Deep learning approaches address this limitation through automatic hierarchical feature learning. Artificial Neural Networks (ANNs) provided early gains in detection accuracy given sufficient training data [4], while Deep Neural Networks (DNNs) extended this capability with multiple hidden layers, achieving strong performance on NSL-KDD and UNSW-NB15 at the cost of increased training complexity [4].

Convolutional Neural Networks, originally developed for image processing, have been adapted to extract local spatial correlations among traffic features, reporting superior accuracy relative to conventional machine learning in several studies [5], [6], though they are less capable of capturing long-range temporal dependencies. Recurrent Neural Networks (RNNs) were introduced to model sequential traffic behavior [7] but suffer from vanishing/exploding gradients on long sequences. Long Short-Term Memory (LSTM) networks address this limitation using gated memory cells and have become one of the most widely used architectures for modeling temporal attack patterns [8], [9], albeit with higher computational cost. Autoencoders provide an unsupervised alternative, learning normal traffic representations and flagging anomalies via reconstruction error, which is particularly useful for zero-day detection when labeled attack data is scarce [10].

C. Hybrid and Advanced Architectures

Because CNNs and LSTMs capture complementary characteristics (spatial versus temporal), hybrid CNN-LSTM architectures have been proposed to combine both strengths, generally reporting improved accuracy and lower false-positive rates relative to standalone models [13], [6]. Attention mechanisms have further been integrated into CNN-LSTM pipelines to emphasize the most discriminative features, improving performance on high-dimensional and multiclass tasks [14]. Autoencoder-based hybrids combine unsupervised feature learning with DNN, CNN, or ensemble classifiers to reduce computational complexity while retaining strong detection of unknown attacks [15]. More recently, Transformer-based architectures leveraging self-attention [16] and Graph Neural Networks that model topological relationships among hosts and flows [17] have shown promising results, though often at increased implementation and computational cost.

D. Research Gap

Across this body of work, individual studies typically evaluate a single architecture using dataset- and study-specific preprocessing, feature selection, and evaluation protocols, which makes it difficult to draw reliable conclusions about the relative effectiveness of different deep learning architectures. In particular, it remains unclear whether the added complexity of hybrid and ensemble models yields consistent performance gains over well-tuned standalone architectures when all models are trained and evaluated under identical conditions. This study addresses that gap by comparing CNN, LSTM, DNN, CNN-LSTM Hybrid, and DNN-LSTM Ensemble models within a single, controlled experimental framework on a common feature set.

III. METHODOLOGY

The proposed framework consists of five stages: dataset acquisition, data preprocessing, feature selection, model development, and comparative performance evaluation, all applied identically across the five candidate architectures (Fig. 1 describes the overall pipeline conceptually).

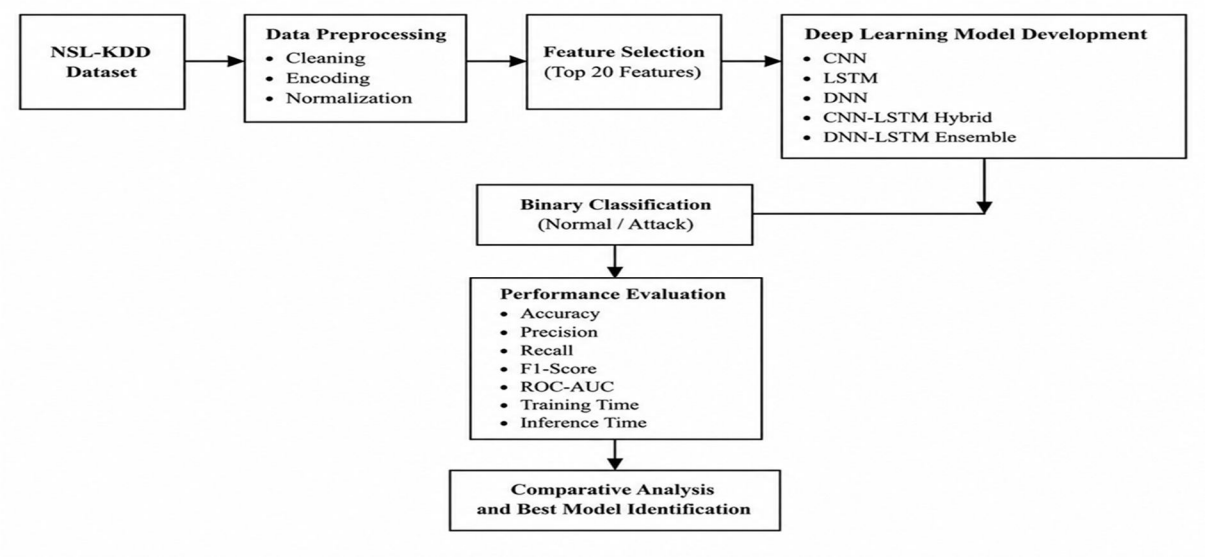


Figure 1. Proposed System Architecture

A. Dataset

The NSL-KDD dataset, an improved version of the KDD Cup 1999 dataset with reduced redundancy and a more balanced class distribution, was used as the benchmark for training, validation, and testing. The dataset comprises 125,973 training records and 22,544 testing records, each described by 41 traffic features plus a class label. Multiple attack categories, Denial of Service (DoS), Probe, Remote-to-Local (R2L), and User-to-Root (U2R), were merged into a single Attack class to formulate a binary classification problem (Normal = 0, Attack = 1).

B. Data Preprocessing

- Binary class conversion of the original multi-category labels into Normal and Attack classes.
- Label encoding of categorical attributes: protocol_type, service, and flag.
- Min-Max normalization of all numerical features to the range [0, 1]: $X_{norm} = (X - X_{min}) / (X_{max} - X_{min})$.
- Partitioning of the data into training, validation, and testing subsets.

C. Feature Selection

To reduce dimensionality and computational overhead while preserving discriminative information, Random Forest feature importance analysis and correlation analysis were used to rank the 41 available features. The top twenty features, dominated by traffic-volume attributes (src_bytes, dst_bytes), service and protocol indicators (same_srv_rate, flag, protocol_type), and host/behavioral statistics (dst_host_same_srv_rate, dst_host_srv_count, logged_in, srv_serror_rate, diff_srv_rate), were retained for model development, a 51.22% reduction from the original feature space (Table 1).

Table 1. Top Ten Features Identified by Random Forest Importance Analysis

Rank	Feature	Importance
1	src_bytes	0.1952
2	dst_bytes	0.1000
3	same_srv_rate	0.0818
4	dst_host_same_srv_rate	0.0686
5	flag	0.0665
6	dst_host_srv_count	0.0652
7	logged_in	0.0525
8	srv_serror_rate	0.0399
9	diff_srv_rate	0.0358
10	protocol_type	0.0325

D. Model Architectures

All five models perform binary classification (sigmoid output) using the same 20 selected input features, with the Adam optimizer, binary cross-entropy loss, and early stopping / checkpointing applied consistently.

1) CNN Model

The CNN uses two 1-D convolutional blocks (32 and 64 filters, kernel size 3, ReLU) each followed by batch normalization and max pooling, then a flatten layer, a 128-unit dense layer with 0.3 dropout, and a sigmoid output layer (31,553 parameters).

2) LSTM Model

Two stacked LSTM layers (64 and 32 units) with batch normalization and 0.3 dropout after each layer are followed by a 64-unit dense layer and a sigmoid output (31,873 parameters).

3) DNN Model

A fully connected network with three hidden dense layers (128, 64, and 32 units, ReLU), batch normalization, and 0.3 dropout after the first two layers, followed by a sigmoid output (13,825 parameters), the most compact of the five architectures.

4) CNN-LSTM Hybrid Model

A single 1-D convolutional layer (64 filters, kernel size 3) with batch normalization, max pooling, and dropout feeds a 64-unit LSTM layer, followed by a 32-unit dense layer and a sigmoid output (35,649 parameters), integrating spatial feature extraction with sequential learning within one network.

5) DNN-LSTM Ensemble Model

Rather than a single joint network, the ensemble combines the sigmoid output probabilities of two independently trained models, the standalone DNN and the standalone LSTM, using weighted averaging: P_ensemble = w_DNN·P_DNN + w_LSTM·P_LSTM. Three weight configurations were evaluated on the validation set (Table 2), and the 0.7/0.3 (DNN/LSTM) configuration was selected as it produced the highest F1-score.

Table 2. Ensemble Weight Configuration Evaluation

DNN Weight	LSTM Weight	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
0.5	0.5	79.81	96.83	66.71	78.99
0.6	0.4	79.95	96.84	66.95	79.17
0.7	0.3	80.31	96.89	67.58	79.62

IV. EXPERIMENTAL SETUP

Experiments were implemented in Python 3.x using TensorFlow/Keras, Scikit-learn, Pandas, and NumPy within a Google Colaboratory environment with GPU acceleration. All models were trained and evaluated on identical training/validation/testing splits of the NSL-KDD dataset using the same twenty selected features to ensure that performance differences reflect architectural characteristics rather than experimental variation.

Model effectiveness was assessed using Accuracy, Precision, Recall (Detection Rate), F1-Score, and ROC-AUC, derived from the confusion matrix components (True Positive, True Negative, False Positive, False Negative):

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad \text{……(1)}$$

$$\text{Precision} = \frac{TP}{TP+FP} \quad \text{……(2)}$$

$$\text{Recall} = \frac{TP}{TP+FN} \quad \text{……(3)}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision+Recall} \quad \text{……(4)}$$

$$\text{False Positive Rate} = \frac{FP}{FP+TN} \quad \text{……(3)}$$

- TP = True Positives (correctly predicted relevant items)
- TN = True Negatives (correctly predicted irrelevant items)
- FP = False Positives (incorrectly predicted relevant items)
- FN = False Negatives (incorrectly predicted irrelevant items)

Computational efficiency was additionally assessed via training time and inference time on the test set.

V. RESULTS AND DISCUSSION

A. *Individual Model Performance*

Table 3 summarizes the classification performance of all five models, and Table 4 summarizes their computational performance.

Table 3. Comparative Classification Performance of Deep Learning Models

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	ROC-AUC (%)
CNN	77.10	96.01	62.36	75.61	91.06
LSTM	80.43	95.74	68.67	79.98	90.65
DNN	80.98	97.08	68.66	80.43	96.11
CNN-LSTM Hybrid	78.36	97.23	63.80	77.04	95.47
DNN-LSTM Ensemble	80.31	96.89	67.58	79.62	94.03

Table 4. Computational Performance of Deep Learning Models

Model	**Training Time (s)**	**Inference Time (s)**
CNN	103.56	1.10
LSTM	104.11	3.31
DNN	39.69	1.93
CNN-LSTM Hybrid	120.16	2.06
DNN-LSTM Ensemble	N/A*	5.24

**The ensemble reuses the previously trained standalone DNN and LSTM models and therefore required no additional training.*

The CNN model attained the highest precision-to-simplicity trade-off among the spatial-only architectures (96.01% precision, 91.06% ROC-AUC) but the lowest recall (62.36%), indicating that spatial feature extraction alone misses a substantial share of attack instances. Introducing temporal modeling in the LSTM improved recall to 68.67% and F1-score to 79.98%, confirming that sequential dependencies carry useful discriminative information for this task.

The standalone DNN achieved the best overall balance, with the highest accuracy (80.98%), F1-score (80.43%), and ROC-AUC (96.11%), while also requiring the least training time (39.69 s) — roughly one-third that of the CNN or LSTM and one-quarter that of the CNN-LSTM Hybrid. This suggests that, given a well-chosen reduced feature set, a fully connected architecture can model the relevant non-linear relationships in NSL-KDD traffic without the added spatial or temporal machinery of CNNs and LSTMs.

The CNN-LSTM Hybrid achieved the highest precision (97.23%) and a strong ROC-AUC (95.47%), reflecting the benefit of combining spatial and temporal feature extraction for minimizing false positives. However, its recall (63.80%) and overall F1-score (77.04%) remained below those of the standalone LSTM and DNN, and its training time (120.16 s) was the highest among all models, indicating that the added architectural complexity did not translate into an overall performance advantage under this experimental setup. The DNN-LSTM Ensemble produced balanced, competitive results (80.31% accuracy, 79.62% F1-score, 94.03% ROC-AUC) that closely approached but did not surpass the standalone DNN, while incurring the highest inference time (5.24 s) since both constituent models must be evaluated at prediction time. This indicates that combining complementary predictions through weighted averaging is a robust strategy but did not provide a decisive advantage over the best standalone model in this configuration.

B. *Comparative Discussion*

Considered jointly, the results show that recall was the primary limiting factor for all five architectures, with the highest recall (LSTM, 68.67%) still leaving roughly one-third of attack instances undetected at the default decision threshold, despite ROC-AUC scores above 0.90 for every model — indicating strong underlying ranking ability that is not fully exploited by the default classification threshold. Precision was uniformly high (>95% for all models), reflecting a low false-positive rate that is desirable for reducing analyst alert fatigue in operational settings.

A key finding is that increasing architectural complexity did not yield consistent performance gains: the CNN-LSTM Hybrid and DNN-LSTM Ensemble models, despite integrating two learning mechanisms, did not exceed the standalone DNN in accuracy, F1-score, or ROC-AUC, while requiring substantially more computation (training time for CNN-LSTM was roughly 3× that of DNN; ensemble inference time was roughly 2.7× that of DNN).

This suggests that, for the twenty selected NSL-KDD features used in this study, the standalone DNN captured the dominant discriminative structure of the data, and additional spatial or temporal machinery introduced complexity without a corresponding accuracy benefit.

C. Comparison with Existing Literature

Table 5 situates the best-performing model from this study (DNN) against representative results reported in recent literature. Reported accuracies in prior work range from roughly 90% to above 97%, generally exceeding the results obtained here.

Table 5. Comparison with Representative Results in the Literature

Study	Model	Dataset	Reported Performance
Kim et al. (2022)	CNN-based IDS	NSL-KDD	Accuracy: 90–95%
Optimized LSTM (2025)	LSTM	NSL-KDD	Accuracy > 90%
Bamber et al. (2025)	CNN-LSTM Hybrid	NSL-KDD	Accuracy > 95%
Attention CNN-LSTM (2025)	Attention CNN-LSTM	NSL-KDD / Bot-IoT	Accuracy: 94.8–97.5%
Transformer-Based IDS (2025)	Transformer	UNSW-NB15	F1-Score: 92%
Present Work	DNN	NSL-KDD	Acc.: 80.98%, F1: 80.43%, ROC-AUC: 96.11%

This gap is attributable primarily to methodological differences rather than architectural weakness: many higher-accuracy studies use the complete 41-feature set, extensive hyperparameter tuning, attention mechanisms, or dataset-specific optimization aimed at maximizing accuracy on a single dataset. In contrast, the present study deliberately used a reduced 20-feature subset under a common, unoptimized-per-model pipeline in order to isolate the effect of architecture choice from confounding preprocessing and tuning differences. The high ROC-AUC (96.11%) obtained by the DNN despite the reduced feature space indicates that strong discriminative information was retained, and that the accuracy gap could plausibly be narrowed through decision-threshold tuning, class-imbalance handling, or expanded feature sets — directions noted for future work.

VI. CONCLUSION AND FUTURE WORK

This study presented a controlled comparative evaluation of five deep learning architectures — CNN, LSTM, DNN, a CNN-LSTM Hybrid, and a DNN-LSTM Ensemble — for binary network intrusion detection on the NSL-KDD dataset, using identical preprocessing, a common twenty-feature subset selected via Random Forest importance analysis, and consistent evaluation metrics. The standalone DNN achieved the best overall performance (80.98% accuracy, 97.08% precision, 68.66% recall, 80.43% F1-score, 96.11% ROC-AUC) with the lowest training time among all models, while the CNN-LSTM Hybrid achieved the highest precision and the LSTM achieved the highest recall. Notably, the hybrid and ensemble models did not outperform the standalone DNN despite their added architectural complexity and computational cost, indicating that model complexity is not, by itself, a reliable predictor of intrusion detection performance.

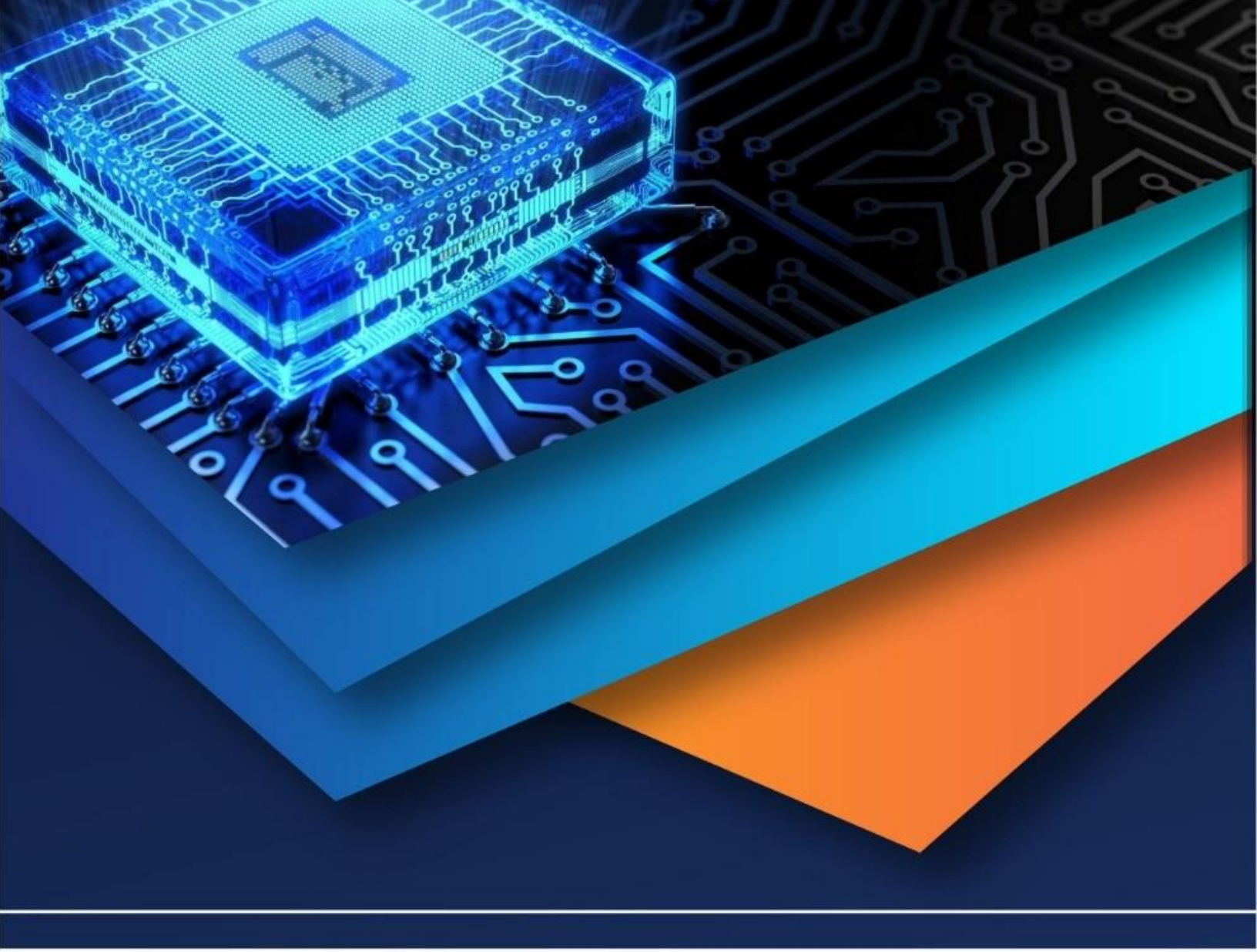
These findings support the practical value of well-designed standalone architectures for binary intrusion detection when computational efficiency is a priority, and provide a controlled reference point for future comparative work. Limitations of this study include reliance on a single benchmark dataset (NSL-KDD), a binary classification scope, a fixed twenty-feature subset, and limited hyperparameter search. Future work will extend this framework to multi-class attack classification (DoS, Probe, R2L, U2R), cross-dataset validation using UNSW-NB15, CICIDS2017, and Bot-IoT, attention-based and Transformer architectures, adaptive/dynamic ensemble weighting, explainable AI techniques (e.g., SHAP, LIME) for interpretability, and evaluation in real-time deployment settings.

REFERENCES

[1] Khairi ASMA, Nugroho EP, Rachman JR. Implementation of signature based intrusion detection system with Snort rule on E-voting system. Journal of Computers for Society. 2023;4(1):17-36. https://doi.org/10.17509/jcs.v4i1.71176

[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. Applied Sciences. 2022;12(2):852. https://doi.org/10.3390/app12020852

[3] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 dataset," Proc. IEEE Symp. Computational Intelligence for Security and Defense Applications, pp. 1–6, 2009.
[4] Y. Xin, L. Kong, Z. Liu, Y. Chen, Y. Li, H. Zhu, M. Gao, H. Hou, and C. Wang, "Machine Learning and Deep Learning Methods for Cybersecurity," IEEE Access, vol. 6, pp. 35365–35381, 2018.
[5] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, "End-to-End Encrypted Traffic Classification with One-Dimensional Convolution Neural Networks," IEEE Int. Conf. Intelligence and Security Informatics, pp. 43–48, 2017.
[6] Vinayakumar R, Alazab M, Soman KP, Poornachandran P, Al-Nemrat A, Venkatraman S. Deep learning approach for intelligent intrusion detection system. IEEE Access. 2019;7:41525-41550. https://doi.org/10.1109/ACCESS.2019.2895334
[7] J. Kim, J. Kim, H. L. T. Thu, and H. Kim, "Long Short Term Memory Recurrent Neural Network Classifier for Intrusion Detection," Int. Conf. Platform Technology and Service, pp. 1–5, 2016.
[8] M. Alauthman, N. Aslam, M. Al-Kasassbeh, S. Khan, A. Al-Qerem, and K. Choo, "An Efficient Reinforcement Learning-Based Botnet Detection Approach," Journal of Network and Computer Applications, vol. 150, 2020.
[9] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A Deep Learning Approach to Network Intrusion Detection," IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 2, no. 1, pp. 41–50, 2018.
[10] J. An and S. Cho, "Variational Autoencoder Based Anomaly Detection Using Reconstruction Probability," Special Lecture on IE, vol. 2, no. 1, pp. 1–18, 2015.
[11] H. Hindy, D. Brosset, E. Bayne, A. Seeam, C. Tachtatzis, R. Atkinson, and X. Bellekens, "A Taxonomy and Survey of Intrusion Detection System Design Techniques, Network Threats and Datasets," IEEE Communications Surveys & Tutorials, vol. 24, no. 2, pp. 1105–1141, 2022.
[12] A. Ferrag, L. Maglaras, H. Janicke, J. Jiang, and T. Shu, "Deep Learning for Cyber Security Intrusion Detection: Approaches, Datasets, and Comparative Study," Journal of Information Security and Applications, vol. 50, 2020.
[13] Y. Li, R. Ma, and R. Jiao, "A Hybrid Malicious Code Detection Method Based on Deep Learning," International Journal of Security and Its Applications, vol. 9, no. 5, pp. 205–216, 2015.
[14] Ullah S, Boulila W, Koubaa A, Ahmad J. Attention-based hybrid deep learning model for intrusion detection in IIoT networks. Procedia Computer Science. 2024;246:3323-3332. https://doi.org/10.1016/j.procs.2024.09.307
[15] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "Deep Learning Approach Combining Autoencoders and Deep Belief Networks for Intrusion Detection," IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 2, no. 1, pp. 41–50, 2018.
[16] A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep Learning for Cyber Security Intrusion Detection: A Review of Transformer-Based Models," Journal of Information Security and Applications, vol. 68, 2022.
[17] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Yu, "A Comprehensive Survey on Graph Neural Networks," IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 1, pp. 4–24, 2021.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY