A Comparative Study of Cloud-Based Deep Learning Models for Intrusion Detection in Computer Networks

Rayan Firdous[1], Aniket Patil[2]
[1,2]*Student, Bharati Vidyapeeth (Deemed to Be University) College of Engineering, Pune*

***Abstract*—Cloud computing provides elastic and on-demand computing resources, but the scale, heterogeneity, and continuously changing nature of cloud network traffic make intrusion detection increasingly difficult. Deep learning models can learn nonlinear and sequential patterns from network-flow data and can be deployed as centralized or distributed security services. This paper presents a comparative, literature-based study of cloud-oriented deep learning approaches for network intrusion detection, focusing on Deep Neural Networks (DNN), Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), CNN-LSTM hybrids, and more recent recurrent-attention architectures. Public benchmark datasets, including CIC-IDS2017, CSE-CIC-IDS2018, and UNSW-NB15, are examined with respect to traffic diversity, feature representation, class imbalance, and deployment relevance. Reported studies indicate that hybrid architectures generally provide stronger feature extraction and temporal modeling than isolated CNN or LSTM models, while cloud-native deployments add important dimensions such as scalability, inference latency, throughput, privacy, and operational cost. The comparison also shows that accuracy alone is insufficient for selecting an IDS model because benchmark protocols, class distributions, preprocessing, and train-test splits vary substantially. The study therefore recommends evaluating detection quality together with false-positive rate, F1-score, inference latency, throughput, and scalability. The resulting analysis provides a practical framework for selecting deep learning architectures for cloud-based intrusion detection without treating reported benchmark scores as directly interchangeable.**

***Index Terms*—Cloud computing, deep learning, intrusion detection, CNN, LSTM, network security, anomaly detection**

I. INTRODUCTION

Cloud computing has become a dominant model for delivering computing, storage, networking, and application services. Under the NIST definition, cloud computing provides convenient, on-demand network access to a shared pool of configurable resources that can be rapidly provisioned and released with limited management effort [1]. This elasticity is valuable for organizations, but it also changes the security problem. Resources appear and disappear dynamically, traffic volumes vary rapidly, workloads are distributed, and multiple services may share the same underlying infrastructure. Consequently, a security mechanism designed for a fixed network perimeter may not be sufficient for a cloud environment.

An Intrusion Detection System (IDS) monitors network or host activity and identifies events that may represent unauthorized or malicious behavior. Network-based IDS approaches are especially relevant to cloud environments because large volumes of network-flow and telemetry data can be collected centrally or at distributed observation points. Recent reviews identify deep learning as an important direction for IDS research because neural architectures can learn nonlinear representations directly from traffic features and can model patterns that are difficult to capture with manually engineered rules [2], [3].

The central question of this study is not simply which deep learning model produces the highest reported accuracy. Instead, it asks which model family offers the most appropriate balance among detection effectiveness, ability to represent network traffic, computational requirements, and cloud deployment characteristics. This distinction is important because published results are usually obtained under different datasets, preprocessing pipelines, class definitions, and validation protocols.

The objectives of this study are: (1) to compare major deep learning architectures used for network intrusion detection; (2) to examine the benchmark datasets commonly used for evaluation; (3) to analyze reported detection metrics and deployment characteristics; and (4) to identify the architecture or architecture class that provides the strongest practical trade-off for cloud-based deployment.

The remainder of the paper reviews relevant work, introduces the mathematical foundations used for comparison, defines the comparative methodology, summarizes reported results, discusses cloud deployment implications, and concludes with limitations and future research directions.

II. LITERATURE REVIEW

Research on intelligent intrusion detection has progressed from statistical and rule-based techniques to machine learning and deep learning. A recent survey of deep learning IDS approaches identifies CNN, RNN, LSTM, autoencoder, DNN, and hybrid architectures as major families used for traffic classification and anomaly detection [2]. A broader review of cloud-security research likewise reports increasing use of machine learning and deep learning to address large-scale and heterogeneous cloud threats [3].

Benchmark selection is a major factor in this literature. CIC-IDS2017 was designed to provide labeled benign and attack traffic representing common attack scenarios and includes flow features generated using CICFlowMeter [4]. The dataset covers five days of captured activity and includes attacks such as brute force, DoS, DDoS, web attacks, infiltration, botnet, and Heartbleed. UNSW-NB15 was constructed using a combination of real normal activity and synthetic contemporary attacks and provides 49 generated features across nine attack categories [5]. CSE-CIC-IDS2018 extends the CIC approach to a larger environment and is explicitly distributed through AWS; it includes network captures, system logs, and more than 80 traffic features [12]. These datasets are useful for benchmarking, but their construction and label distributions differ, so numerical scores should not be treated as a universal ranking.

Sun et al. introduced DL-IDS, a CNN-LSTM hybrid architecture evaluated on CICIDS2017. Their study reported 98.67% overall accuracy and compared the hybrid approach with CNN-only, LSTM-only, and other machine-learning models [6]. The result supports the architectural argument that local feature extraction and sequential modeling can complement each other. However, the reported score is tied to the study's own preprocessing, feature selection, and experimental protocol.

Sajid et al. developed a hybrid approach using XGBoost or CNN for feature extraction followed by LSTM classification and evaluated the approach across CIC-IDS2017, UNSW-NB15, NSL-KDD, and WSN-DS [7]. Their work illustrates another important direction: combining feature-selection mechanisms with temporal models rather than relying on a single deep learning architecture.

Bamber et al. compared ANN, LSTM, BiLSTM, CNN-LSTM, GRU, and BiGRU models on NSL-KDD after feature selection with Recursive Feature Elimination and a decision-tree classifier [8]. Their CNN-LSTM model achieved 95% accuracy, 0.89 recall, and 0.94 F1-score under their experimental protocol. The study is particularly useful for comparison because several model families were tested within one experimental setting.

More recent cloud-oriented work has moved beyond simple CNN-LSTM structures. Lokhande and Sonawane proposed an IDS-as-a-Service framework combining BiLSTM, GRU, attention, a modified gating mechanism, and GAN-assisted synthetic anomaly generation, with experiments conducted in an AWS deployment environment [9]. Their reported multi-class results were 97.91% accuracy, 98.67% precision, 97.93% recall, and 98.18% F1-score. These results demonstrate the importance of evaluating the deployment environment itself, rather than considering the classifier in isolation.

The literature therefore reveals three recurring themes. First, hybrid models often perform well because different architectures encode different properties of traffic. Second, class imbalance and feature redundancy remain important sources of misleadingly high overall accuracy. Third, cloud deployment introduces additional evaluation dimensions, including latency, throughput, resource consumption, scalability, and data privacy. These themes form the basis of the comparative framework used in this paper.

III. MATH REVIEW

1. Data Normalization

Network-flow features may have very different numerical ranges. Min-Max normalization maps a feature to a bounded interval and can be written as:

$$x' = (x - x_{min}) / (x_{max} - x_{min}) \quad (1)$$

Normalization is important because a feature with a larger numerical scale can otherwise dominate gradient-based optimization. Recent IDS implementations commonly apply scaling or standardization before neural-network training [7].

2. Neural Activation and Classification

For binary intrusion classification, the sigmoid function converts a model output into a value between zero and one:

$$\sigma(z) = 1 / (1 + e^{(-z)}) \quad (2)$$

For multi-class intrusion classification, the softmax function is commonly used to obtain a probability distribution over M classes:

$$\text{Softmax}(z_j) = e^{(z_j)} / \Sigma_{(k=1 \text{ to } M)} e^{(z_k)} \quad (3)$$

3. Evaluation Metrics

Accuracy measures the fraction of correctly classified traffic records. Precision measures the proportion of predicted attacks that are actually attacks, while recall measures the proportion of actual attacks that are detected. F1-score provides the harmonic mean of precision and recall. For an IDS, recall and false-positive rate are especially important because missed attacks and excessive false alarms have different operational costs.

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN) \quad (4)$$

$$\text{Precision} = TP / (TP + FP) \quad (5)$$

$$\text{Recall} = TP / (TP + FN) \quad (6)$$

$$F1 = 2 \times \text{Precision} \times \text{Recall} / (\text{Precision} + \text{Recall}) \quad (7)$$

A comparison based only on accuracy can therefore be misleading when attack classes are imbalanced. Macro-F1, per-class recall, confusion matrices, and false-positive rate should be examined alongside accuracy [10].

4. Cloud Performance Measures

In a cloud deployment, the model must also be evaluated by inference latency, throughput, resource utilization, and scalability. A detector with slightly lower classification accuracy may be operationally preferable if it processes substantially more flows per second at lower resource cost. Conversely, a complex recurrent-attention model may be justified when the security requirement prioritizes detection quality over compute cost.

IV. PROBLEM STATEMENT AND COMPARATIVE METHODOLOGY

The problem addressed in this study is the selection of a deep learning architecture for intrusion detection in cloud-connected computer networks. The selection must account for both predictive performance and deployment characteristics. Instead of conducting a new experimental run with unverified hardware or a privately unavailable dataset, this paper uses a structured comparative methodology based on published experimental studies and official dataset documentation.

1. Comparative Criteria

Five model families are compared: DNN/MLP, CNN, LSTM/GRU, CNN-LSTM hybrids, and attention-enhanced recurrent hybrids. The comparison considers four dimensions: representation capability, detection performance, computational complexity, and cloud suitability.

2. DNN / MLP

A feed-forward DNN is a natural baseline because it can model nonlinear relationships among flow features without assuming a particular spatial or temporal structure. It is relatively straightforward to deploy and can be efficient when the feature vector is compact. Its limitation is that a standard feed-forward architecture does not explicitly model the ordering or temporal dependency of traffic observations.

3. CNN

One-dimensional CNNs can learn local patterns from ordered feature representations or traffic sequences using convolutional filters. Their parallelizable operations make them attractive for high-throughput inference. CNNs are effective feature extractors, but by themselves they may be less suited to long-range dependencies across traffic sequences.

4. LSTM and GRU

LSTM networks use gated memory to preserve information over sequences and are therefore useful when attack behavior develops across multiple time

steps. GRU networks provide a related recurrent mechanism with fewer gates and can reduce computational complexity. Their primary disadvantage is that recurrent processing is less naturally parallel than convolutional processing and may increase training and inference cost for long sequences.

5. CNN-LSTM Hybrid

CNN-LSTM models combine the strengths of convolution and recurrence. The CNN stage learns local representations, while the LSTM stage models dependencies among the extracted features. Sun et al. reported strong performance from this combination on CICIDS2017 [6], and Bamber et al. reported the CNN-LSTM model as the strongest among several compared deep learning models on NSL-KDD [8].

6. Attention-Enhanced Recurrent Hybrid

Attention-enhanced recurrent systems attempt to emphasize the most informative parts of a traffic sequence. Recent cloud research has combined attention with BiLSTM and GRU components and, in some cases, synthetic anomaly generation to address class imbalance [9]. These systems can be powerful but introduce additional model complexity and therefore require careful cloud resource planning.

7. Cloud Deployment Pipeline

A practical cloud-based IDS can be represented as: Traffic Collection → Flow/Feature Extraction → Preprocessing → Feature Selection → Model Inference → Alert Generation → Storage and Monitoring. Cloud elasticity can provide additional compute during traffic spikes, while model serving can be separated from storage and monitoring services. The same architecture also creates security requirements: telemetry must be protected in transit and at rest, access to model artifacts must be controlled, and the detector itself should be monitored for drift and adversarial manipulation.

V. IMPLEMENTATION / EXPERIMENT SETUP

1. Dataset Characteristics

Dataset	Primary characteristics	Attack coverage	Cloud relevance
CIC-IDS2017	Labeled network flows; >80 extracted flow features	Brute force, DoS/DDoS, web attacks, infiltration, botnet, Heartbleed	High; widely used for DL benchmarking
CSE-CIC-IDS2018	AWS-hosted traffic and system logs; >80 flow features	Brute force, Heartbleed, botnet, DoS/DDoS, web attacks, infiltration	Very high; explicitly available through AWS
UNSW-NB15	49 generated features; 9 attack families	Fuzzers, analysis, backdoors, DoS, exploits, generic, reconnaissance, shellcode, worms	High; diverse modern benchmark
NSL-KDD	Legacy IDS benchmark with normal and four broad attack families	DoS, Probe, R2L, U2R	Moderate; useful for controlled comparison but less representative of current traffic

CIC-IDS2017 contains benign activity and several common attack categories and was generated over five days with realistic background activity [4]. CSE-CIC-IDS2018 was designed with a larger infrastructure and is explicitly provided through AWS, making it particularly relevant to cloud-based experimentation [12]. UNSW-NB15 contains two million-plus records in four CSV files and provides predefined training and testing sets [5]. NSL-KDD remains useful for controlled comparisons, but recent studies acknowledge that it is an older benchmark and should not be treated as a complete representation of modern network traffic [8].

2. Preprocessing

A reproducible IDS pipeline should remove duplicate or invalid records were justified, encode categorical variables, scale numerical features, and separate training and evaluation data before fitting normalization parameters. Class imbalance should be quantified rather than hidden by accuracy. SMOTE or other synthetic sampling approaches may be used when justified, but synthetic data must be applied only to the training portion to prevent information leakage.

3. Comparative Evaluation Protocol

For a future controlled implementation, all model families should be trained on the same dataset partitions, use identical preprocessing, and be evaluated with the same class definitions. Hyperparameter searches should be restricted to the training/validation data. The final test set should remain untouched until model selection is complete. This protocol is essential because published IDS studies frequently differ in their preprocessing and train-test strategies, which limits direct comparison.

4. Cloud Configuration

A cloud experiment can use a CPU instance for lightweight DNN baselines and GPU-backed instances for CNN, LSTM, and hybrid models. The training layer should be separated from the inference layer so that the production detector can scale independently. Containerization also improves reproducibility by fixing library versions and model dependencies. Monitoring should record inference latency, throughput, CPU/GPU utilization, memory consumption, and alert rates in addition to classification metrics.

VI. RESULTS AND DISCUSSION

Table II summarizes selected published results. These values are reproduced as reported by the cited studies and are not results generated by the present paper. Because the datasets, splits, preprocessing methods, and class definitions differ, the table should be interpreted as an evidence map rather than a universal league table.

Study	Model / approach	Dataset(s)	Reported result	Deployment observation
Sun et al. [6]	CNN-LSTM (DL-IDS)	CICIDS2017	98.67% overall accuracy	Strong hybrid benchmark result; cloud deployment not primary focus
Bamber et al. [8]	CNN-LSTM vs. ANN, LSTM, BiLSTM, GRU, BiGRU	NSL-KDD	95% accuracy; 0.89 recall; 0.94 F1	Shows benefit of hybrid architecture under one controlled comparison
Sajid et al. [7]	XGBoost-LSTM / CNN-LSTM	CIC-IDS2017, UNSW-NB15, NSL-KDD, WSN-DS	High detection performance with low FAR reported	Emphasizes feature extraction plus temporal modeling
Lokhande & Sonawane [9]	Attention BiLSTM-GRU + GAN; IDSaaS	CICIDS-2018, UNSW-NB15, SCAPY, SCADANet	97.91% accuracy; 98.67% precision; 97.93% recall; 98.18% F1	AWS deployment; throughput and latency considered

Observations and Analysis

1) Hybrid architectures show a consistent conceptual advantage. CNN layers can extract local feature relationships while recurrent layers model sequential behavior. The independent studies of Sun et al. and Bamber et al. both report strong CNN-LSTM performance relative to alternative models within their respective experimental settings [6], [8].

2) Recurrent models remain important when the ordering of traffic events carries information. LSTM and GRU architectures explicitly preserve state across observations, making them suitable for multi-step or evolving attack patterns. Their computational cost, however, becomes important at cloud scale, particularly when sequence lengths increase.

3) Attention improves selectivity but increases architectural complexity. The 2026 IDSaaS study combines BiLSTM and GRU with attention, synthetic anomaly generation, and cloud deployment, reporting strong multi-class performance and high throughput [9]. This demonstrates a shift from benchmark-only evaluation toward operational metrics.

4) Dataset choice can dominate apparent model performance. CIC-IDS2017 and CSE-CIC-IDS2018 contain more contemporary attack categories than legacy benchmarks, while UNSW-NB15 offers a different feature and attack taxonomy. NSL-KDD remains useful for comparison but should not be used alone to claim modern cloud readiness [5], [8].

5) Accuracy is not sufficient for deployment decisions. A model may obtain a high accuracy score because benign traffic dominates the test set. For security operations, false-positive rate, minority-class recall, macro-F1, and inference latency can be more informative. Systematic reviews have also identified inconsistent validation methodology and dataset limitations as recurring problems in AI-based NIDS research [10].

6) Cloud deployment changes the optimization target. A model that is computationally expensive but accurate may be practical during offline training yet unsuitable for real-time inference at very high traffic volumes. Conversely, a lightweight CNN or DNN may offer better throughput but miss longer temporal relationships. The best design is therefore application-dependent rather than universally determined by one metric.

VII. CONCLUSION

This paper presented a comparative study of cloud-based deep learning approaches for network intrusion detection using published evidence and official benchmark documentation. The comparison shows that DNNs provide a useful baseline, CNNs offer efficient local feature extraction, LSTM and GRU models capture sequential dependencies, and CNN-LSTM hybrids combine complementary representations. Attention-enhanced recurrent architectures extend this idea by dynamically emphasizing relevant traffic information, although they require greater computational and implementation complexity.

The reviewed studies provide evidence that hybrid architectures can achieve strong detection performance. CNN-LSTM models have produced high reported accuracy on CICIDS2017 and NSL-KDD under their respective experimental protocols [6], [8]. More recent cloud-native research has demonstrated that IDS can be delivered as a service and evaluated using both classification metrics and operational measures such as throughput [9].

The principal conclusion is therefore not that one architecture is universally superior. For a cloud IDS, CNN-LSTM is a strong general-purpose choice when both feature extraction and sequential context matter. LSTM/GRU variants are appropriate when temporal behavior is dominant, while lighter DNN/CNN models can be preferable when throughput, cost, and low latency are the primary constraints. Attention-based recurrent hybrids are promising for complex and evolving environments but should be justified by the available compute budget and operational requirements.

Most importantly, future comparisons should use common preprocessing, identical splits, macro-averaged metrics, false-positive analysis, latency, throughput, and resource utilization. Without these controls, differences in reported accuracy cannot reliably be interpreted as differences in model capability.

VIII. FUTURE WORKS

Future research should move from isolated benchmark comparisons toward reproducible cloud experiments using multiple contemporary datasets. An especially useful direction is cross-dataset evaluation, in which a model trained on one environment is tested on a different environment to measure generalization and dataset bias.

Another direction is lightweight model design. Knowledge distillation, pruning, quantization, and efficient recurrent or convolutional architectures can reduce inference cost while retaining detection quality. This is particularly relevant when an IDS must scale horizontally with traffic volume.

Privacy-preserving learning is also important for distributed cloud environments. Federated learning can reduce the need to centralize raw traffic data, while robust aggregation and secure communication can address additional risks. Explainable AI should also be integrated so that security analysts can understand why an event was classified as malicious.

Finally, future IDS research should evaluate adversarial robustness, concept drift, zero-day detection, and continuous model monitoring. Cloud environments change rapidly, so a detector that performs well on a static benchmark may degrade after deployment. Continuous evaluation and careful retraining are therefore essential for a reliable production system.

ACKNOWLEDGMENT

We express sincere gratitude to the Department of Electrical and Computer Engineering, Bharati Vidyapeeth (Deemed to Be University) College of

Engineering, Pune, for providing the academic environment and resources required for this study.

REFERENCES

[1] Mell, P., and T. Grance, "The NIST definition of cloud computing," NIST Special Publication 800-145, National Institute of Standards and Technology, 2011, doi: 10.6028/NIST.SP.800-145.

[2] Al-Safaar, D., and W. L. Al-Yaseen, "A survey of network intrusion detection systems based on deep learning approaches," Scientific and Technical Journal of Information Technologies, Mechanics and Optics, vol. 23, no. 2, pp. 352–363, 2023, doi: 10.17586/2226-1494-2023-23-2-352-363.

[3] Alzoubi, Y. I., A. Mishra, and A. E. Topcu, "Research trends in deep learning and machine learning for cloud computing security," Artificial Intelligence Review, vol. 57, Art. no. 132, 2024, doi: 10.1007/s10462-024-10776-5.

[4] Sharafaldin, I., A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in Proc. 4th International Conference on Information Systems Security and Privacy, 2018.

[5] Moustafa, N., and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems," in Proc. Military Communications and Information Systems Conference, 2015.

[6] Sun, P., P. Liu, Q. Li, C. Liu, X. Lu, R. Hao, and J. Chen, "DL-IDS: Extracting features using CNN-LSTM hybrid network for intrusion detection system," Security and Communication Networks, 2020, Art. no. 8890306, doi: 10.1155/2020/8890306.

[7] Sajid, M., K. R. Malik, A. Almogren, T. S. Malik, A. H. Khan, J. Tanveer, and A. U. Rehman, "Enhancing intrusion detection: A hybrid machine and deep learning approach," Journal of Cloud Computing, vol. 13, Art. no. 123, 2024, doi: 10.1186/s13677-024-00685-x.

[8] Bamber, S. S., A. V. R. Katkuri, S. Sharma, and M. Angurala, "A hybrid CNN-LSTM approach for intelligent cyber intrusion detection system," Computers & Security, vol. 148, Art. no. 104146, 2025, doi: 10.1016/j.cose.2024.104146.

[9] Lokhande, U. B., and K. Sonawane, "Adaptive intrusion detection system for cloud security using deep learning," Journal of Cloud Computing, vol. 15, Art. no. 103, 2026, doi: 10.1186/s13677-026-00955-w.

[10] Maseer, M., et al., "Meta-analysis and systematic review for anomaly network intrusion detection systems: Detection methods, dataset, validation methodology, and challenges," IET Networks, vol. 13, nos. 5–6, pp. 339–376, 2024, doi: 10.1049/ntw2.12128.

[11] Jain, V., K. Singh, S. Joshi, L. Malviya, R. Patel, V. Gupta, et al., "Optimizing intrusion detection using a hybrid CNN LSTM deep learning framework on the NSL KDD dataset," Discover Computing, vol. 29, Art. no. 194, 2026, doi: 10.1007/s10791-026-10056-6.

[12] Canadian Institute for Cybersecurity, "CSE-CIC-IDS2018 on AWS," University of New Brunswick, online dataset documentation, accessed Oct. 8, 2026.