JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Review on Analysis of Autoencoder Based and NetBERT Based Intrusion Detection System

[1]Shivam Sidharth, [2]Prashant Pranav, [3]Sandip Dutta and [4]Iram Siddiquee

[1]Department of Cybersecurity and Digital Forensics, National Forensic Sciences University, Dharwad Campus, Karnataka-580011, India

[2,3,4]Department of Computer Science and Engineering, Birla Institute of Technology, Mesra

ABSTRACT:

The instant growth of internet technologies and communication systems has led to a major growth in network size and complexity. This expansion has also brought a rise in advanced cyber threats, making intrusion detection a crucial challenge. Intrusion Detection Systems (IDS) act as an important defense tool, continuously monitoring network gathering to maintain its confidentiality, integrity, & availability. Regardless of significant research, IDS solutions still struggle with progressive detection accuracy, decreasing false positives, and identifying new attack patterns. Recently, the mixture of Machine Learning and Deep Learning methods has shown potential in boosting IDS performance. This article examines the basic principles of IDS and highlights innovative ML and DL methods aimed at creating stronger network-based intrusion detection systems. It also provides a thorough review of existing methods, evaluation standards, and data set options that influence the current development of intelligent IDS.

Keywords: NIDS, AutoEncoder, NetBERT, CICDS 2017

1. Introduction

Over the past decade, the need for robust network-based intrusion detection systems has grown sharply as cyberattacks—such as DoS, DDoS, and spoofing—have become more frequent and sophisticated. This threat landscape has accelerated the adoption of AI and machine learning techniques, particularly deep learning, which have improved detection accuracy, system robustness, and the ability to analyze threats at a finer granularity. At the same time, intrusion detection has expanded beyond traditional networks into cloud environments and IoT ecosystems, reflecting how rapidly digital infrastructure is evolving.

The idea of intrusion detection traces back to James Anderson’s 1980 proposal to use system audit logs to spot suspicious activity; that foundational concept underpins modern IDS design. Today’s systems augment that legacy with advanced models—RNNs, CNNs, DBNs, LSTMs—that address key challenges such as reducing dimensionality, detecting anomalies, and recognizing temporal patterns.

Unlike host-based approaches that focus on single endpoints, network-based IDSs are placed at strategic points (for example, gateways, routers, and firewalls) to monitor traffic across entire segments in real time. They commonly employ signature-based detection, which matches traffic to known attack example, and anomaly-based methods, which flag deviations from established norms. Such centralized monitoring is especially valuable for large networks and data centers that require coordinated incident response.

As cybersecurity architectures evolve, NIDS have become essential components of layered defenses alongside firewalls, endpoint protections, and intrusion prevention systems. Trends like zero trust, edge computing, and 5G are driving demand for more adaptive, intelligent NIDS. Nonetheless, persistent challenges remain—inspecting encrypted traffic, lowering false positive rates, and detecting sophisticated evasion tactics—necessitating continued advances in algorithms, dataset quality, and the integration of contextual threat intelligence.

Deployments in important sectors such as energy, healthcare, manufacturing, and transportation highlight the need for prompt detection and response, since brief delays can have severe consequences. Consequently, NIDS are increasingly integrated with IPS, SIEM platforms, and automated orchestration to enable rapid mitigation measures like packet filtering, device isolation, and incident escalation.

Because attackers are themselves using automated, learning-driven methods to probe and alter network behavior, NIDS must move beyond purely reactive detection toward predictive capabilities that leverage historical data, behavioral models, and threat intelligence. In short, network-based intrusion detection has evolved from static, rule-driven tools into dynamic, AI-enabled systems that play a central role in safeguarding both legacy networks and the new, interconnected infrastructures emerging today.

2. Related Work

This section surveys machine learning methods applied to intrusion detection. Broadly, IDS approaches lie into two sections: signature & anomaly-based. Signature-based systems detect attacks by comparing traffic against a database of known patterns; they are effective for previously observed threats but require continual updates and typically fail to catch novel or zero-day exploits.

Anomaly-based systems instead flag behavior that departs from an established baseline, making them better suited to uncovering previously unseen attacks. These methods frequently rely on machine learning and are more adaptable as adversary tactics evolve. Within machine learning, anomaly-detection strategies are commonly grouped into three paradigms:

- **Supervised learning:** Solutions are trained on labeled examples of normal & malicious traffic. This approach can achieve strong detection rates but depends on large, accurately labeled datasets, which are costly and time-consuming to produce.
- **Semi-supervised learning:** Systems leverage a small set of labeled samples together with the bigger pool of unlabeled data, reducing annotation effort while maintaining scalability in heterogeneous traffic environments.
- **Unsupervised learning:** Techniques identify outliers without relying on labels, which is advantageous when traffic patterns are highly variable or labels are unavailable.

A wide array of supervised algorithms has been evaluated for IDS, including K-NN, Support Vector Machines, and various neural network architectures. Researchers have explored ensemble and stacking strategies to boost performance—for instance, combining SVMs with other classifiers or using evolutionary methods for feature selection to improve true positive rates and reduce errors. Multi-stage pipelines that use a feature selector (such as Random Forest) followed by several classifiers stacked together have also shown improved detection over single-model baselines. Semi-supervised approaches that incorporate fuzzy reasoning have been proposed to enhance classifier robustness when labeled data are sparse.

Hybrid frameworks that blend supervised and unsupervised components are another active direction. These architectures aim to balance detection accuracy and computational efficiency by coupling different learners (decision trees, boosting methods, simple stumps, and probabilistic models) to capture complementary aspects of traffic behavior.

More recently, deep learning has gained prominence because its layered representations can automatically withdraw hierarchical features from raw network data, minimizing reliance on manual feature engineering. Studies comparing shallow and deep networks highlight how depth and feature learning affect anomaly-detection outcomes. Autoencoder-based and sparse-representation approaches combined with classifiers such as softmax regression have demonstrated promising flexibility and effectiveness for network intrusion detection.

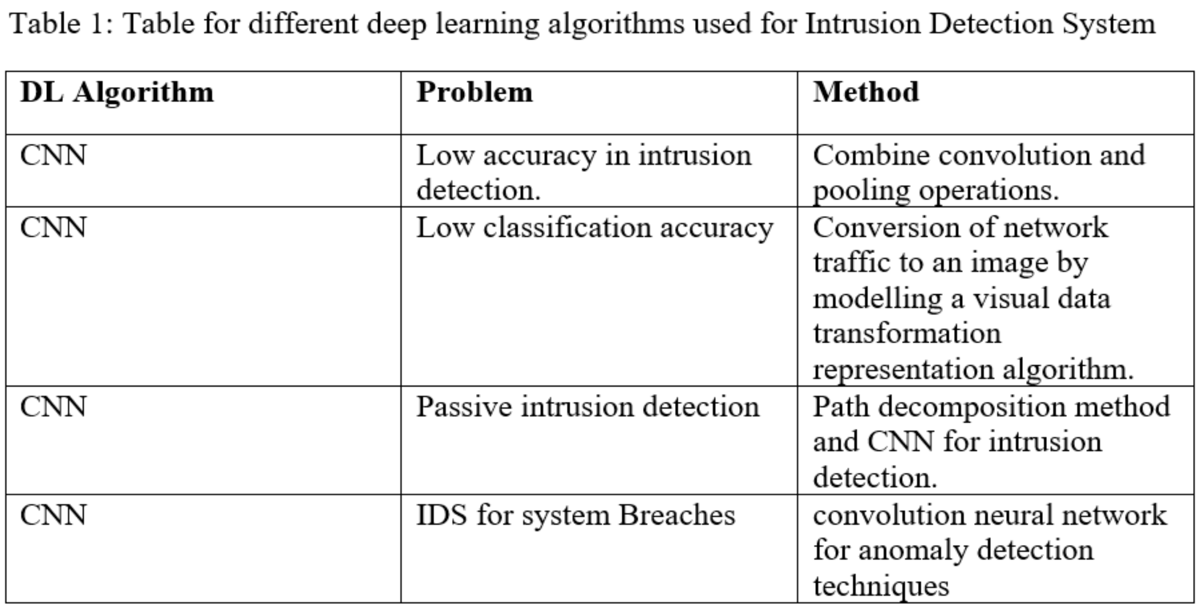
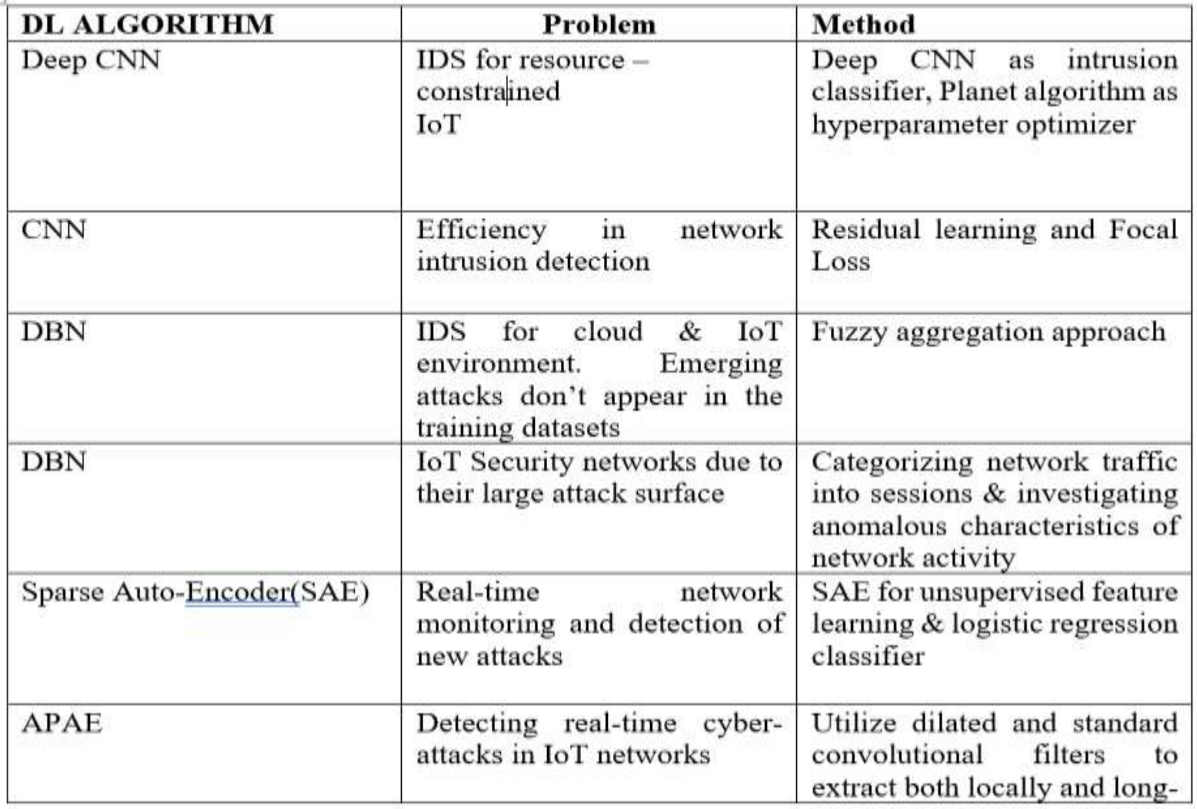


Table 1: Table for different deep learning algorithms used for Intrusion Detection System

DL Algorithm	**Problem**	**Method**
CNN	Low accuracy in intrusion detection.	Combine convolution and pooling operations.
CNN	Low classification accuracy	Conversion of network traffic to an image by modelling a visual data transformation representation algorithm.
CNN	Passive intrusion detection	Path decomposition method and CNN for intrusion detection.
CNN	IDS for system Breaches	convolution neural network for anomaly detection techniques

Figure 2: Table for different Deep Learning (DL) algorithms used for Intrusion Detection System



DL ALGORITHM	Problem	Method
Deep CNN	IDS for resource – constrained IoT	Deep CNN as intrusion classifier, Planet algorithm as hyperparameter optimizer
CNN	Efficiency in network intrusion detection	Residual learning and Focal Loss
DBN	IDS for cloud & IoT environment. Emerging attacks don't appear in the training datasets	Fuzzy aggregation approach
DBN	IoT Security networks due to their large attack surface	Categorizing network traffic into sessions & investigating anomalous characteristics of network activity
Sparse Auto-Encoder(SAE)	Real-time network monitoring and detection of new attacks	SAE for unsupervised feature learning & logistic regression classifier
APAE	Detecting real-time cyber-attacks in IoT networks	Utilize dilated and standard convolutional filters to extract both locally and long-

3. PROPOSED FRAMEWORK

The framework introduced in this study employs a two-tier deep learning architecture to enhance the detection of network intrusions. To ensure both high performance and rapid classification in real-time environments, the initial stage utilizes an Autoencoder-based approach. The process begins with loading and preprocessing the CICIDS-2017 dataset, which includes cleaning, normalization, and feature preparation.

In the first stage, all available features are input into the Auto- encoder to learn customized representations of the data. This stage is responsible for distinguishing between benign and malicious traffic, assigning a probability score to each classification outcome to reflect prediction confidence.

The second stage incorporates the NetBERT transformer, which leverages attention mechanisms and contextual embeddings to refine the classification results. By analyzing the latent features extracted from the Autoencoder, NetBERT improves the system's capability to find subtle anomalies and emerging threats with greater precision.

This dual-stage design combines unsupervised feature learning with advanced sequence modeling, resulting in a more adaptive and accurate intrusion detection system.

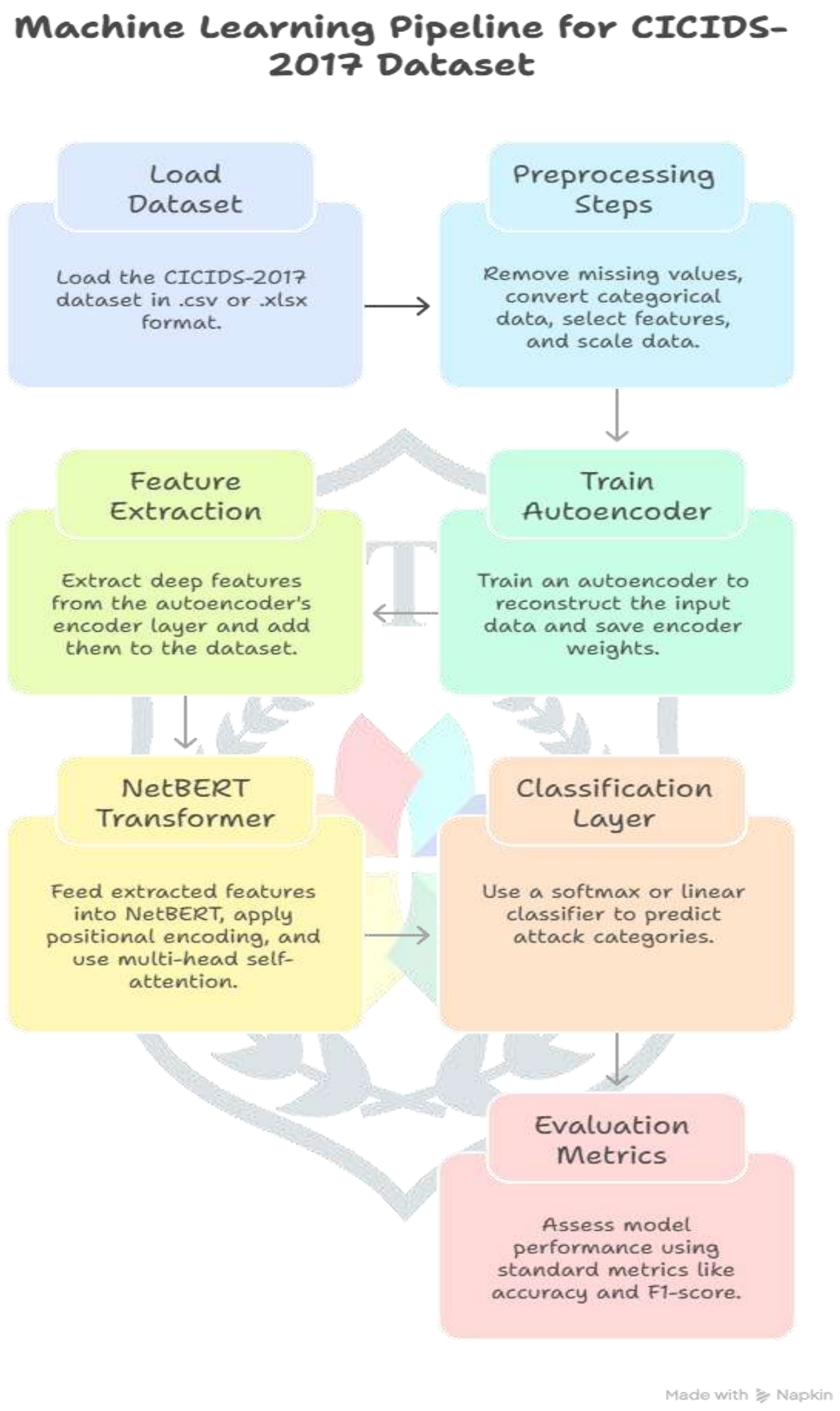

Machine Learning Pipeline for CICIDS-2017 Dataset
Load Dataset
Load the CICIDS-2017 dataset in .csv or .xlsx format.
Preprocessing Steps
Remove missing values, convert categorical data, select features, and scale data.
Train Autoencoder
Train an autoencoder to reconstruct the input data and save encoder weights.
Feature Extraction
Extract deep features from the autoencoder's encoder layer and add them to the dataset.
NetBERT Transformer
Feed extracted features into NetBERT, apply positional encoding, and use multi-head self-attention.
Classification Layer
Use a softmax or linear classifier to predict attack categories.
Evaluation Metrics
Assess model performance using standard metrics like accuracy and F1-score.
Made with Napkin

Figure 1: ML Pipeline for CICIDS 2017 Dataset

4. METHODOLOGY

The adoption of AI/ML in network-based intrusion detection marks a significant shift in cybersecurity practices. Traditional NIDS relied on fixed rules and signature databases, whereas AI-enabled systems can learn evolving attack behaviors—such as protocol tunneling and polymorphic malware—and adapt their detection strategies accordingly. The construction of an intelligent NIDS typically follows a staged pipeline described below.

4.1 Data collection

A reliable AI/ML-driven NIDS begins with comprehensive traffic capture. Researchers gather data using packet-capture tools (for example, Wireshark or tcpdump) or purpose-built sensors placed at routers, gateways, and other critical network points. Captured features commonly include IP addresses, port numbers, protocol types, packet sizes, and payload-derived statistics. Public benchmarks such as KDD Cup 99, NSL-KDD, UNSW-NB15, and CICIDS2017 are frequently used for training and evaluation.

4.2 Data processing

Raw network traces are often noisy, voluminous, and heterogeneous, so preprocessing is essential before model training. Typical steps include:

- Feature extraction: converting raw packets and flows into structured attributes (e.g., connection counts, mean packet size, protocol frequencies).
- Normalization: scaling numeric features to a common range (for instance, 0 to 1) to stabilize and speed up learning.
- Labelling: assigning classes such as “normal” or specific attack types (DoS, Probe, R2L, U2R).
- Dimensionality reduction (optional): applying methods like PCA to shrink feature space while retaining discriminative information.

4.3 Model selection

The choice of learning method depends on the data characteristics and detection goals:

- Supervised methods—including Decision Trees, Random Forests, SVMs, Naive Bayes, and feedforward neural networks—depend on labeled examples and are trained to distinguish benign traffic from malicious activity.
- Unsupervised learning: clustering methods (K-Means, DBSCAN) and reconstruction techniques (autoencoders) are useful when labels are scarce and anomalies are identified as deviations from learned norms.
- Deep learning models such as CNNs, RNNs, and LSTMs are effective at modeling intricate spatial and temporal relationships present in high-dimensional network data.

4.4 Deployment strategies

IDS solutions are usually deployed as:

- Host-based IDS (HIDS): agents installed on endpoints to monitor local events and enforce device-level security; these can be difficult to scale across many machines without incurring performance costs.
- Network-based IDS (NIDS): sensors located at central network choke points to observe traffic across segments in real time; this centralized view is the focus of this work due to its broad coverage and suitability for coordinated incident response.

4.5 Evaluation metrics

Model performance is assessed using confusion-matrix-derived measures:

- True Positive (TP): correctly detected attacks.
- False Negative (FN): attacks mistakenly labeled normal.
- False Positive (FP): benign traffic incorrectly flagged as attacks.
- True Negative (TN): normal traffic correctly classified.
From these quantities, precision, recall, F1-score, and overall accuracy are computed to provide a balanced evaluation of detection capabilities.

4.6 Autoencoder

Autoencoders are unsupervised neural networks that learn compact representations of data for tasks like dimensionality reduction, automatic feature extraction, and spotting anomalies. They are valuable when labeled examples are limited because they learn compact encodings of input patterns without supervision—useful in IDS scenarios where labeling is costly. A standard autoencoder has an encoder that compresses input data into a low-dimensional latent vector and a decoder that reconstructs the input from that code. Training minimizes reconstruction error on normal data so that anomalous inputs yield higher reconstruction loss. Autoencoders can also serve as feature extractors for downstream tasks and support transfer learning across related datasets.

4.6.1 Encoder

The encoder maps input vectors to progressively smaller hidden representations and typically includes:

- An input layer for raw features.
- Multiple hidden layers that reduce dimensionality.
- Nonlinear activation functions (e.g., ReLU, sigmoid) to capture complex relationships.

4.6.2 Latent space

The bottleneck (latent) layer holds a compact representation of the input; its dimensionality is chosen based on the trade-off between compression and information preservation and underpins both reconstruction and anomaly detection.

4.6.3 Decoder

The decoder reconstructs the input from the latent code. Activation choices depend on data type (sigmoid for binary-like features, linear for continuous values). Reconstruction loss is the primary signal for anomaly scoring: larger losses indicate deviations from learned normal patterns.

Recent work demonstrates that deep autoencoders can scale to large traffic datasets and effectively support threat detection. For instance, models trained on modern IDS benchmarks have reported high accuracy after addressing class imbalance using techniques such as SMOTE. Variants like asymmetric deep autoencoders have been proposed to improve feature extraction for growing and evolving network traffic. Semi-supervised frameworks based on stacked encoder–decoder networks (for example, systems that use a primary detector for bulk processing and a secondary model for ambiguous cases) have shown competitive performance on benchmarks such as NSL-KDD, indicating practical readiness for deployment.

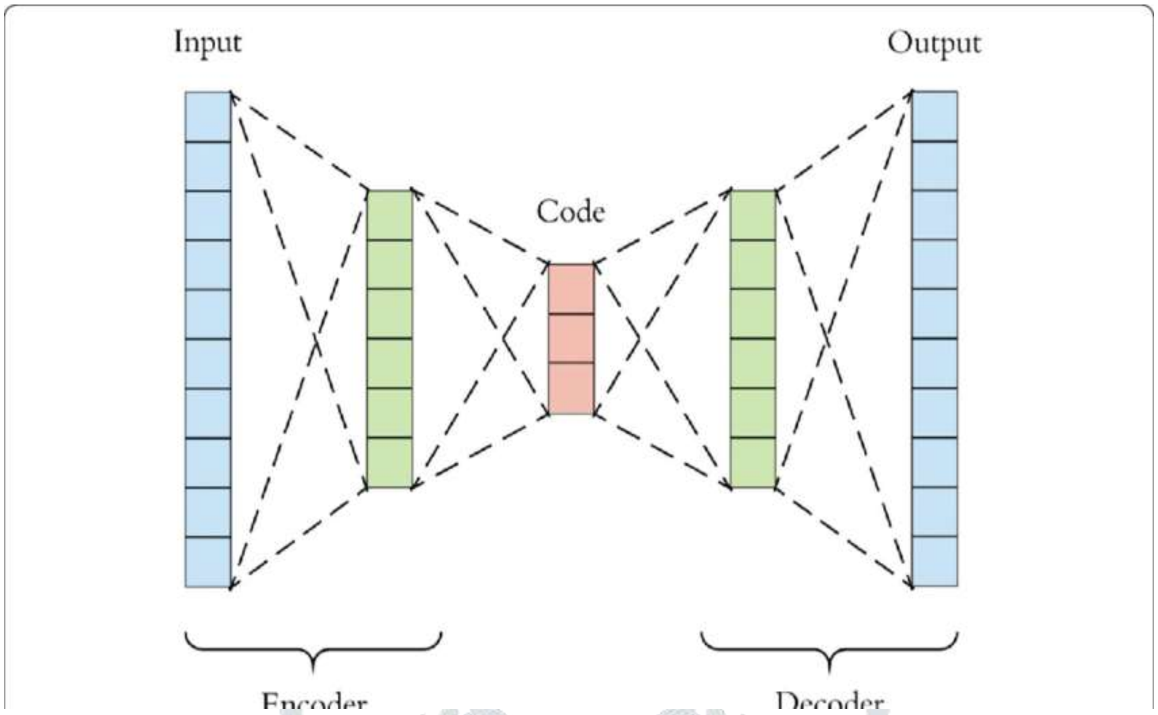


Figure 2. Autoencoder Architecture

4.7 NetBERT Transformer

NetBERT is a transformer-derived deep learning model tailored to improve both the adaptability and detection accuracy of network intrusion systems by leveraging bidirectional contextual encoding. Drawing on the design principles of BERT from NLP, NetBERT treats sequences of network flows or packet-level features as tokens, enabling it to model complex dependencies and temporal patterns among session attributes such as source/destination IPs, ports, packet-size profiles, and protocol metadata. Unlike conventional approaches that depend heavily on manual feature engineering and often struggle to generalize, NetBERT learns task-relevant representations end-to-end. Its multi-head self-attention layers assign varying importance to tokens in a sequence, producing richer contextual embeddings that enhance the system's ability to identify subtle anomalies.

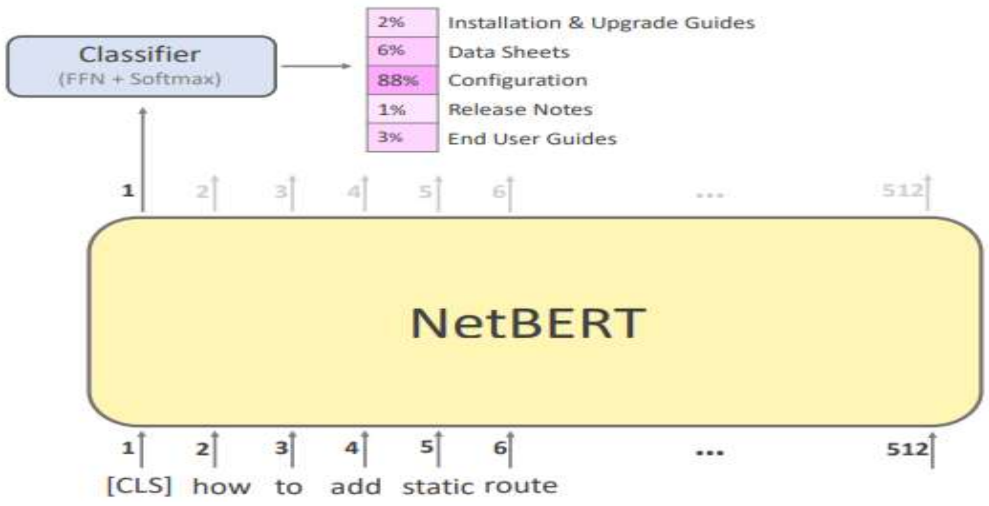


Figure 3. NetBERT Architecture

The model is typically pretrained on large volumes of unlabeled network traffic using masked token prediction, and fine-tuned on labeled attack datasets such as CICIDS 2017 or NSL-KDD for supervised classification tasks. NetBERT demonstrates strong performance in identifying both known and zero-day attacks due to its ability to generalize learned patterns, making it highly suitable for real-time intrusion detection in modern, dynamic network environments. Moreover, its modularity allows it to be integrated with downstream systems like autoencoders or graph-based anomaly detectors for improved hybrid detection pipelines. The use of transformers like NetBERT signifies a shift towards more intelligent, context-aware intrusion detection models capable of adapting to evolving cyber threats. The important components of NetBERT are as follows:

4.7.1 Input Embedding Layer

The input embedding layer is the first critical component in NetBERT. In traditional NLP-based BERT models, this layer converts tokens into vector representations. In a similar fashion, in NetBERT, preprocessed structured network features are encoded into fixed-size numerical vectors, including IP addresses, port numbers, types of protocols, and packet-level statistics.

These features may include both categorical and continuous values, either embedded directly in the case of categorical fields or normalized and projected in the case of numerical ones. This step ensures that all incoming data, no matter what form it may have initially taken, is transformed into a dense, learnable representation capable of passing through the transformer architecture. The purpose of an embedding is to capture semantic meaning and statistical behavior of network features across sessions.

4.7.2. Positional Encoding

Since transformers are inherently permutation-invariant and do not process input data sequentially by default, NetBERT uses positional encoding to inject information about the position or order of features in a given network sequence. In this context, the sequence could refer to ordered packets, flows, or feature tokens arranged over time. These positional encodings are summed together with the input embeddings to allow the model to differentiate context and relevance based on position for each feature. This is particularly important when trying to detect patterns evolving temporally, such as slow data exfiltration or distributed attacks.

4.7.3 Multi-Head Self-Attention

The multi-head self-attention mechanism lies at the center of NetBERT's ability to contextualize relationships in network data. Each attention head enables the model to concentrate on multiple aspects of the input sequence simultaneously, learning dependencies among features from multiple perspectives. For example, one attention head may learn the interaction between source and destination IPs, while another focuses on temporal patterns related to packet size or protocol behavior. These attention scores calculate the significance of every token in relation to others, helping NetBERT understand higher-order local and global structures in the data. This property is especially useful for identifying very subtle anomalies or unseen attack patterns that only manifest through inter-feature correlations.

4.7.4 Transformer Encoder Blocks

In particular, NetBERT consists of stacked transformer encoder blocks, composed of layer normalization, multi-head self-attention, and position-wise feed-forward neural networks with residual connections. These encoder layers are used to progressively refine the internal representations of the input features. With each additional layer, the model captures deeper and more abstract patterns of the network data. The residual connections help reduce problems like vanishing gradients and improve the stability of the training process such that the model can scale to much deeper architectures. By doing so, NetBERT can learn complex relationships and generalize well to new, unseen traffic scenarios.

4.7.5 Classification Head

A task-specific classification head tops the NetBERT architecture. This normally consists of a fully connected (dense) layer followed by a softmax or sigmoid activation function, depending on the nature of the output: multi-class or binary. During fine-tuning, this layer maps the contextualized embeddings produced by the encoder stack into class labels corresponding to normal or malicious traffic types. The classification head is trained on labeled datasets like CICIDS 2017, NSL-KDD, and UNSW-NB15. It explicitly enables the model to perform intrusion detection by giving high probabilities to potential attack vectors with regard to learned contextual representations.

4.7.6 Modularity and Hybrid Integration

By design, NetBERT is quite modular; thus, it can easily be combined with other detection mechanisms to enhance performance. For example, its embeddings may be fed into an autoencoder for anomaly detection or combined with recurrent layers like LSTMs to model long-range temporal dependencies. This makes NetBERT suitable for a wide range of deployment settings and hybrid detection pipelines. It can also function both as a feature extractor within a larger system and as a standalone classifier, depending on the specifics of the intrusion detection task.

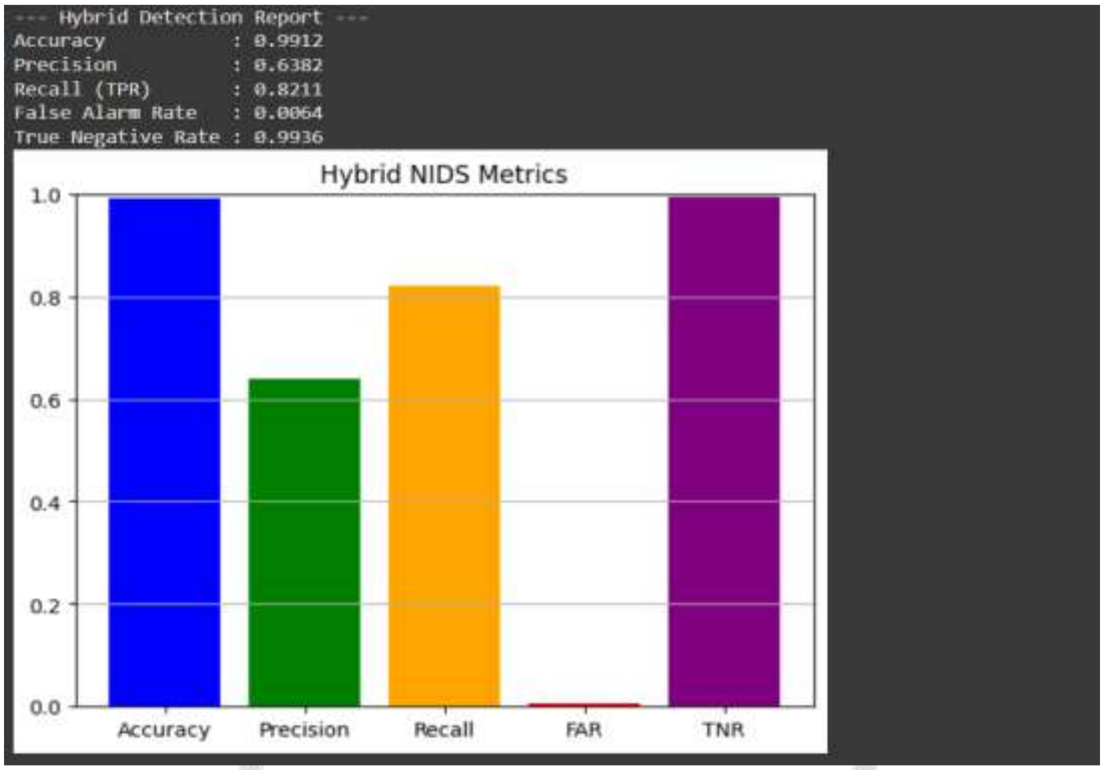
5. Results and Discussion

The experiments reported here use the CICIDS2017 benchmark from the Canadian Institute for Cybersecurity, a realistic and widely adopted dataset for evaluating network intrusion and anomaly-detection methods. CICIDS2017 simulates a modern enterprise environment by recording both benign and malicious traffic over a five-day period (3–7 July 2017), and it includes a variety of attack types embedded among normal network activities.

Flows in the dataset are produced by the CIC Flow Meter from raw PCAP captures and contain over eighty features per flow; each flow represents a bidirectional exchange between hosts and carries a label indicating normal behavior or a specific attack class. The captured environment was segmented and instrumented (clients, web/mail/file servers, routers, firewalls), and routine user actions such as web browsing, VoIP, e-mail, and SSH sessions were interleaved with scripted attacks to preserve realism.

Because of its rich feature space, realistic traffic mixes, and explicit attack labels, CICIDS2017 remains a preferred resource for NIDS research. Researchers must, however, contend with issues like severe class imbalance, the need for careful feature selection, and concerns about scalability when moving models from offline evaluation to real-time, large-scale deployment.

Using a hybrid pipeline that combines NetBERT with an autoencoder-based feature extractor, our system gain an accuracy of 99.12% and a true negative rate of 99.36% on the evaluated splits. The detailed metric breakdown is presented below.



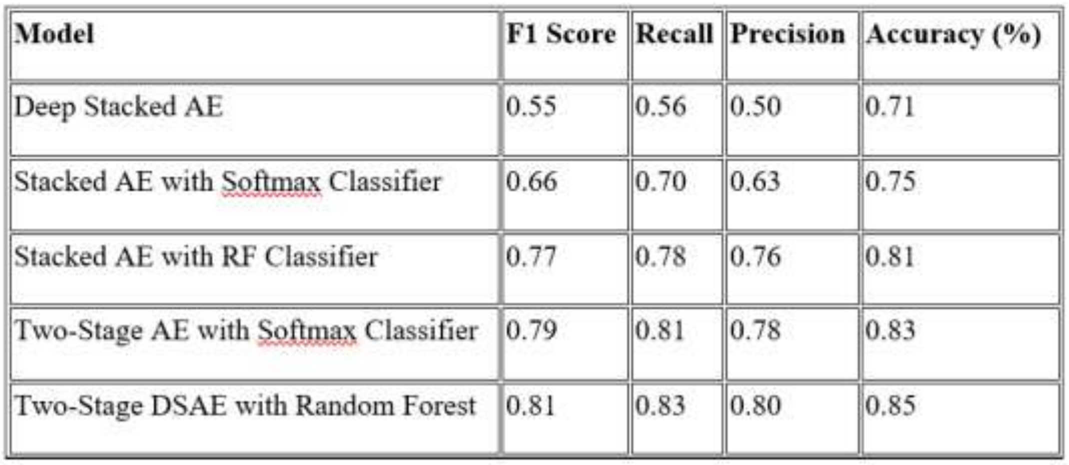
```
--- Hybrid Detection Report ---
Accuracy            : 0.9912
Precision           : 0.6382
Recall (TPR)        : 0.8211
False Alarm Rate    : 0.0064
True Negative Rate  : 0.9936
```

Figure 4. Results of Evaluation

These results are comparable when the algorithms are not used in tandem. As seen throughout the time many researchers used various algorithms for their studies on Intrusion Detection System, the summary for it is given below:

Table 3: Comparison Table of Results of Past Experiments

Table 1. Results Comparison Between Deep Stacked Autoencoder and Two-Stage Deep Learning with Shallow Learning (CUHKSZ 2017 Dataset)



Model	F1 Score	Recall	Precision	Accuracy (%)
Deep Stacked AE	0.55	0.56	0.50	0.71
Stacked AE with Softmax Classifier	0.66	0.70	0.63	0.75
Stacked AE with RF Classifier	0.77	0.78	0.76	0.81
Two-Stage AE with Softmax Classifier	0.79	0.81	0.78	0.83
Two-Stage DSAE with Random Forest	0.81	0.83	0.80	0.85

Over the past ten years, a broad range of techniques-additionally models, algorithms, and frameworks have been developed to tackle the challenges of network intrusion detection using machine learning and deep learning. Numerous studies have explored various strategies for identifying malware threats within network environments. For instance, Janarthan and Zargiri utilized a Random Forest classifier on the UNSW-NB15 dataset, selecting only five features for classification. Although their model achieved an accuracy of approximately 81.61%, it was acknowledged that further improvements were possible. Despite such efforts, conventional approaches often fall short when applied to real-time network traffic due to several limitations. These include difficulties in processing large-scale data, suboptimal detection accuracy, limited support for dynamic analysis, and poor adaptability to the specific characteristics of the systems they are intended to protect.

A detailed analysis of intrusion detection systems (IDS) presented in the referenced article underscores a growing trend: hybrid models are proving to be highly effective, offering improved accuracy and precision. The combination of Autoencoders with Transformer-based architectures like NetBERT represents a specific advancement in IDS development, particularly for handling complex and high-dimensional traffic data. Tokenization plays a Important role in preprocessing, significantly refining the input data. This hybrid methodology capitalizes on the strengths of both unsupervised feature extraction and contextual sequence understanding, thereby overcoming key drawbacks of traditional IDS approaches especially in terms of adaptability, generalization, and real-time responsiveness. Autoencoders, as unsupervised neural networks, are adept at learning compressed representations of network behavior. In IDS applications, they model the statistical patterns of normal traffic, and any significant deviation—reflected as reconstruction error—can be flagged as a potential anomaly.

6. CONCLUSION AND FUTURE SCOPE

This study presents a novel deep learning-based framework for intrusion detection in network traffic. The architecture operates in two distinct phases: the first involves an autoencoder with two hidden layers, while the second utilizes NetBERT for tokenization. The model is trained using a semi-supervised approach, beginning with unsupervised pre-training of each hidden layer on a large volume of unlabeled data. Subsequently, the model is fine-tuned using labeled network traffic features to enhance its detection capabilities.

Through this comprehensive analysis, we explored various deep learning techniques employed in the development of Intrusion Detection Systems (IDS) over the past five years (2019–2023). The objective was to assess the effectiveness of current methodologies in safeguarding digital environments. Looking ahead, future research should focus on advancing IDS models tailored for the rapidly expanding Internet of Things (IoT) ecosystem. Enhancing these systems will be crucial for mitigating sophisticated cyber threats and ensuring robust security across interconnected devices.

References:

1. Song, Y., Hyun, S. & Cheong, Y.-G. Analysis of autoencoders for network intrusion detection. Sensors 21, 4294 (2021).

2. Yang, L., Song, Y., Gao, S., Hu, A. & Xiao, B. Griffin: Real-time network intrusion detection system via ensemble of autoencoder in sdn. IEEE Transactions on Netw. Serv. Manag. 19, 2269–2281 (2022).

3. Bertoli, G. D. C. et al. An end-to-end framework for machine learning-based network intrusion detection system. IEEE access 9, 106790–106805 (2021).

4. Lamshöft, K., Neubert, T., Krätzer, C., Vielhauer, C. & Dittmann, J. Information hiding in cyber physical systems: Challenges for embedding, retrieval and detection using sensor data of the swat dataset. In Proceedings of the 2021 ACM Workshop on Information Hiding and Multimedia Security, 113–124 (2021).

5. Choi, H., Kim, M., Lee, G. & Kim, W. Unsupervised learning approach for network intrusion detection system using autoencoders. The J. Supercomput. 75, 5597–5621 (2019).

6. Moraboena, S., Ketepalli, G. & Ragam, P. A deep learning approach to network intrusion detection using deep autoencoder. Rev. d'Intelligence Artif. 34, 457–463 (2020).

7. Kimanzi, R., Kimanga, P., Cherori, D. & Gikunda, P. K. Deep learning algorithms used in intrusion detection systems–a review. arXiv preprint arXiv:2402.17020 (2024).

8. Yan, B. & Han, G. Effective feature extraction via stacked sparse autoencoder to improve intrusion detection system. IEEE Access 6, 41238–41248 (2018).

9. Mushtaq, E., Zameer, A., Umer, M. & Abbasi, A. A. A two-stage intrusion detection system with auto-encoder and lstms. Appl. Soft Comput. 121, 108768 (2022).