# Hybrid ML-DL Based Network Intrusion Detection System in Data Science Framework

*Gowtham Sri Vamsi* Angarapu[1,*], *Aashish* Jayakanthan[1,**], and *Gokul Pran* S[1,***]

[1]Vel Tech Rangarajan Dr.Sagunthala R&D Institute of Science and Technology, Computer Science Department, Avadi, Chennai

**Abstract.** Network intrusion detection systems (NIDS) are important in securing the current network infrastructures against the emerging cyber attackers. This paper introduces NIDS, which is one of the hybrid frameworks of machine learning and deep learning ensemble models to classify network traffic as normal or malicious traffic. The proposed system is composed of six classification models, namely Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), K-Nearest Neighbors (KNN), Support Vector machines (SVM), Random Forest, and Naive Bayes. Both models examine the features of the network separately and generate an intrusion probability that is combined to come up with the final prediction in a weighted ensemble strategy. The system offers the two popular benchmark datasets, such as NSL-KDD and CICIDS2017, having different feature representation and traffic properties. Each dataset has its own separate preprocessing pipelines and trained model sets that allow one to select the dataset dynamically using a web-based interface. Experimental analysis has shown that the ensemble method has an accuracy of 76.59 on NSL-KDD and 98.61 on CICIDS2017 and it is also able to give consistent results and performance in terms of precision, recall, and F1-score. The suggested framework is deployed as a full-stack software package that entails Python analytics engine, Node.js backend, MongoDB database and frontend made in React. The findings also demonstrate the efficiency of hybrid ensemble learning in refining the operation of intrusion detection in the heterogeneous network setup.

## 1 Introduction

The mesh network is becoming susceptible to cyber attacks due to the swift growth of digital communication and other cloud-based services. Network integrity may be compromised by malicious actions like denial-of-service attacks, port scanning, brute-force attempts, infiltration attacks among others which distort important services. To overcome these threats, Intrusion Detection Systems (IDS) are common in order to track network traffic in order to detect suspicious behaviour. Specifically, Network Intrusion Detection Systems (NIDS) are used to analyze network packets and network traffic streams so that possible security violations can be identified in real-time. Nonetheless, conventional signature-based detection

*e-mail: Vtu21533@veltech.edu.in
**e-mail: Vtu24190@veltech.edu.in
***e-mail: gokulprans@veltech.edu.in

© The Authors, published by EDP Sciences. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

algorithms use some known patterns of attacks and they are not categorical to detect new or dynamic attacks.

The effectiveness of intrusion detection system has increased considerably due to current improvements in machine learning and deep learning. Support Vector machine or K- Nearest Neighbor or random forest machine learning algorithms have been extensively used to classify network traffic using statistical trends. On the same note, deep learning systems like Convolutional Neural Networks (CNN) and Long Short-Term Memory networks (LSTM) can automatically learn and infer complex patterns and time dependencies on high-dimensional network data. Although these have been done, most available solutions are based on one model only, and it could be limiting in terms of detection accuracy and robustness to be implemented in varied network settings.

This paper offers NIDS as its solution to such shortcomings, which is a hybrid machine learning and deep learning ensemble architecture when it comes to network intrusion detection. The presented system amalgamates six models of classification such as CNN, LSTM, KNN, SVM, Random Forest, and Naive Bayes. These models are combined together using a weighted ensemble approach to provide a final verdict on the intrusion detention. Moreover, the system helps in two benchmark sets, NSL-KDD and CICIDS2017, which are diverse network traffic factors and feature designs. The experimental analysis shows that the suggested framework can obtain optimal detection results in all datasets.

## 2 Related Work and Research Gap

Intrusion detection has taken the form of digital communication with the development of digital communication infrastructures. a field of high research on risk reduction due to emerging and adaptive network attacks.There are several research works which have been carried out over the past two decades have already touched upon the application of machine learning and network intrusion through use of deep learning techniques.

Abdulganiyu et al. [1] gave a detailed systematic review of the evolution of IDS technologies and focusing on the move towards ML driven signature-based anomaly detection. Ahmad et al.[2] also categorized the IDS techniques as traditional.Deep learning methods and provided a comparative study that identified the respective. data-strengths, data-weaknesses, and applicability.

The study on the classical ML models in the context of intrusion detection has been widely carried out. Almutairi et al.[7] demonstrated the effectiveness of SVM, KNN, and Random Forest algorithms in the detection of different types of attacks. A comparative analysis of various ML methods was introduced by Siddharth et al.[12], which revealed that the quality of dataset and features choice determines the accuracy of intrusion detection. The problem of high-dimensional network traffic characteristics and part of the dimensionality reduction tools including PCA and autoencoders were suggested by Abdul hammed et al.[11], which contribute to the classification efficiency with a considerable margin. Lastly, Thaseen et al.[10] explored the effectiveness of ML algorithms in practice. They came to the conclusion that preprocessing, normalization, and tuning are the factors that influence the performance of an IDS.

Deep learning based architectures began to emerge soon, in large-scale datasets and complex and continuously changing attacks. One of the earliest deep learning systems of NIDS was proposed by Ashiku and Dagli [3], which demonstrated better nonlinear pattern recognition than the previous ones. Mohammadpour et al. [4] and Wang et al.[6] were the first to introduce the further advancements of CNN-based systems, and both authors have found that convolutional models greatly exceeded traditional ML methods because they can automati-

cally extract spatial features. Dong et al. [15] also proposed a real-time DL-based IDS that could handle high-velocity network traffic, for which the deep models were well suited.

SDN continued to be used, researchers paid attention to the flexibility of the IDSs in programmable networks. A survey of ML based IDS solutions, in particular SDN, architectural challenges, and opportunities in centralized detection were offered by Sultana et al. [5].Alzahrani and Alenazi[14] created a special ML based SDN IDS which demonstrated better accuracy in flowlevel anomaly detection and controller assisted monitoring.

More recent works are concerned with reliability, robustness, and generalization on unseen datasets.Magan Carron et al.[8] mentioned that different evaluation methods do not cohere. There has been a call by authors to use standardized benchmarking to enable natural comparison of the IDS models. Apruzzese et al.[13]reported the cross-evaluation studies that revealed that most ML-based IDS model fails to generalize when tested not on the same dataset as it was trained on. In contrast, Jmila and Khedher [9] had adversarial threats in machine learning. It was depicted therein that ML and DL are susceptible to evasion attacks hence, the IDS set up must be strong and capable of withstanding adversarial attacks. All these works show that the IDS research has evolved over the years since the conventional ML classifiers to deep neural formations and explain features that remain in correlation to scalability, generalization, security, and reliable evaluation.

## 3 Methodology

### 3.1 System Architecture

The suggested system NIDS is implemented in the form of a modular full stack network intrusion detector, that combines machine learning models and deep learning models into a hybrid ensemble architecture. The architecture has four significant components, which include a web-based front end interface, a backend application server, an analytics engine that uses machine learning, and a database layer. These elements are connected via RESTful APIs in processing network traffic characteristics to produce predictions of intrusion detection. Figure 1 depicts the architecture of the proposed system in general.

The frontend interface has been created on the ReactJS framework and delivers the interface to interact with the intrusion detection system. This interface allows the users to authenticate with the help of secure login method, provide network traffic feature inputs, and demonstrate prediction results produced by the system. The frontend dashboard also enables the users to alternate between the datasets supported and visualise their detect results in terms of normal or intrusion classes, type of attack and level of confidence. The interface is carried out to give a user and interactive image of the detecting capabilities of the system.

The backend application server is built on Node. js and Express framework. This layer is the one that will bridge between the frontend interface and the analytics engine. The backend server A is in charge of API routing, user inputs validation, authentication request and redirecting prediction queries to the machine learning engine. The security features such as the JSON Web Token (JWT) authentication, role-based access control and password hashing are adopted to provide safe communication channels and safeguard sensitive data belonging to the users.

The machine learning analytics engine is written in Python and currently uses Flask and REST API as a service. This element carries out the basic intrusion detection functions such as data preprocessing, data feature encoding, and model inference. The engine combines several classification algorithms such as Convolutional neural networks (CNN), long short-term memory (LSTM), the k-nearest neighbor (KNN), support vectors machine (SVM) and the random forest, the naive Bayes. Both of the models examine the processed feature at

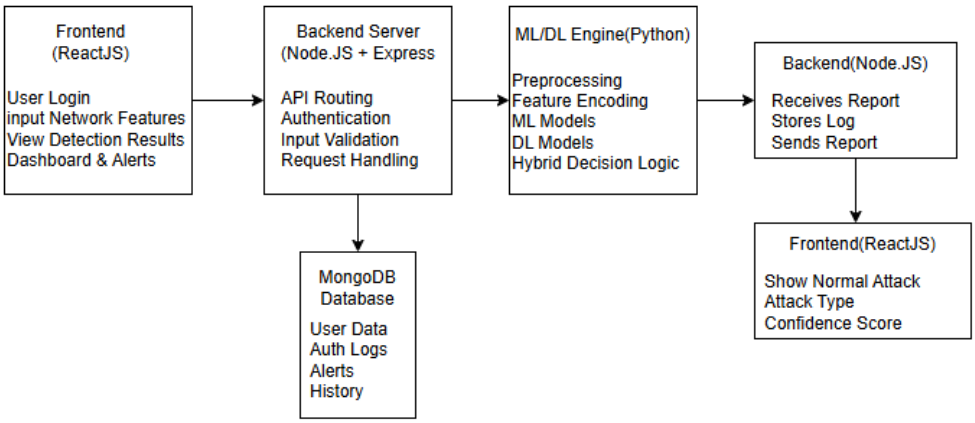


**Figure 1.** System Architecture

once and give out a prediction score. The hybrid ensemble decision mechanism is then used to combine these outputs to produce the resultant classification.

The MongoDB database layer implements the database layer, which is in charge of storing data in systems and operational logs. The database is used to store user authentication data, history of prediction, records of system audit, which are used to track user actions or prediction events. This logging system is traceable and aids in the monitoring of the usage and security events of the systems.

Network traffic features are provided by the user during the prediction process using the frontend interface. The backend server checks the feedback and sends the request to the analytics engine by calling a REST API. The analytics engine will invoke the relevant preprocessing pipeline, and input the data down the ensemble models to produce a prediction result. The end result of the detection is the result in the form of the predicted class and the confidence score, which is sent back to the backend server and shown to the user via the frontend dashboard.

### 3.2 Hybrid ML-DL Ensemble Model

To improve the strength and accuracy of intrusion detection, the presented NIDS framework will utilize a hybrid ensemble method that combines ML and DL systems. The rationale of the approach is to integrate the power of various algorithms to be able to detect complex patterns in the network traffic data. Whereas deep learning models are good at extracting high-level feature representation, classical machine learning algorithms offer good statistical classification. The combination of these methods can enable the system to have higher detection performance and enhance generalization in new network settings.

The hybrid ensemble model uses six classification algorithms, which include CNN, LSTM networks, KNN, SVM, RF, and NB. Each of the models analyses the input feature vector of network traffic characteristics independently and generates a probability score that the network traffic instance is normal or malicious.

CNN model captures local feature relationship in the input data with convolutional filters. CNN can detect space patterns and relationships among the network features that can depict malicious behaviour by performing convolution operations on the feature vector. The

LSTM model is a recurrent neural network that is used to capture the sequential dependencies and temporal patterns of features of network traffic. This enables the model to identify behavioural patterns that could happen over time as part of intrusion events.

Along with the deep learning models, there are four classical machine learning algorithms incorporated into the system. KNN classifier takes a sample and compares it with its closest neighbors in the feature space to identify the class of the sample. Euclidean distance This is usually computed as the distance between samples with the help of a metric called the Euclidean distance:

$$d(x_a, x_b) = \sqrt{\sum_{k=1}^{n} (x_{ak} - x_{bk})^2} \tag{1}$$

where $x_a$ and $x_b$ represent feature vectors and $n$ denotes the number of features.
The SVM classifier builds an optimum decision boundary to separate the normal and malicious samples of traffic. Decision capability could be determined as:

$$f(x) = w^T x + b \tag{2}$$

where $w$ represents the weight vector and $b$ is the bias term

Random Forest model is a series of decision trees which are trained based on random samples of the data. An independent prediction is made by each tree and the ultimate classification is made by summing up all the outputs of the trees. Naive Bayes classifier is a probabilistic model which uses the Bayes theorem, and assumes that there is conditional independence amongst features. The likelihood of a given class under the feature vector is given as:

$$P(y \mid X) = \frac{P(X \mid y)P(y)}{P(X)} \tag{3}$$

where X represents the feature vector and $y$ represents the class label

A weighted ensemble fusion strategy is implemented in order to combine the predictions of each and every model. Every classifier generates a probability value which indicates the probability of an intrusion. These probabilities are put together with predetermined weights to each model. Deep learning models have higher weights in the proposed system because they can learn complicated patterns of features. The probability of final ensemble is calculated as:

$$P_{final} = 0.25P_{CNN} + 0.25P_{LSTM} + 0.15P_{KNN} + 0.15P_{SVM} + 0.10P_{RF} + 0.10P_{NB} \tag{4}$$

The sum total of the probabilities will be then matched to some threshold set to show the final classification result. When the ensemble probability is above the threshold value, the network traffic instance is declared to be intrusion whereas, when it is below the threshold value the network traffic instance is declared as normal traffic.

This method of ensemble strategy offers better reliability in the detection because many classifiers are involved in the end decision. Consequently, the suggested system makes the contribution of the individual model bias less significant and delivers more consistent results in the various datasets and network traffic conditions.

### 3.3 Dataset Description

The effectiveness of the suggested NIDS framework was tested on two well-known benchmark datasets of network intrusion detection: NSL-KDD and CICIDS2017. The datasets

have been chosen due to the fact that they represent the various generations of network traffic data and offer various attack scenarios on which to test intrusion detection models. The two datasets will enable the proposed system to be tested on both the traditional and modern network traffic conditions.

NSL-KDD dataset is a better version of the KDD Cup 1999 dataset and it was developed to overcome the limitations of the original dataset e.g. redundant records and biased learning. It has 41 network connection features which characterize various attributes of a network session such as protocol type, service type, packet size, and connection statistics. These characteristics are classified as basic connection features, content features, time-based traffic features and host-based traffic features. The data set consists of normal traffic, and various kinds of attacks in the form of Denial of Service (DoS), Remote-to-Local (R2L), User-to-Root (U2R), and Probe attacks. NSL-KDD dataset has a total of about 125,973 training samples and 22,544 testing samples and is widely used as a benchmark to test intrusion detection algorithms.

The dataset created by the Canadian Institute of Cybersecurity (CICIDS2017) is more recent and reflects more realistic network traffic situations. As opposed to NSL-KDD, which relies on connection-level features, CICIDS2017 has flow-based network traffic features based on packet capture (PCAP) files. The dataset contains 77 numerical features that characterize the network flow attributes like the number of packets, flow time, bytes counts, and protocol behaviour. CICIDS2017 is full of contemporary cyber attacks such as Distributed Denial of Service (DDoS), brute force attacks, intrusion, botnet activity, web attacks, and port scanning. The data set consists of more than three million records of network traffic, which is a huge and diverse dataset to train and evaluate intrusion detection systems.

Given that the two datasets have varying feature structure and traffic characteristics, the proposed NIDS framework employs different preprocessing pipelines and trained model sets in each dataset. The NSL-KDD dataset needs the encoding of categorical variables and normalization of numerical features, whereas the CICIDS2017 dataset is mostly associated with removing missing values and normalizing numerical variables. The system supports the separation of the data processing pipelines, where each dataset is processed in a proper manner without causing any feature inconsistencies.

In order to enable the flexibility of evaluation, the proposed system provides users with the ability to dynamically choose between the NSL-KDD and CICIDS2017 datasets by means of the system interface. The model set and the preprocessing pipeline are loaded and predict on the basis of the chosen dataset. This design will allow the system to exhibit its efficiency in various forms of network traffic settings.

One of the important steps in preparing raw network traffic data to be used in machine learning and deep learning models is data preprocessing. NSL-KDD and CICIDS2017 datasets have various feature structures and data formats, and, thus, different preprocessing pipelines were applied in each of the datasets to guarantee uniform feature representation when training models and making predictions.

In the case of the NSL-KDD data, the raw data has both numeric and categorical variables that describe network connections. Label encoding was used to transform categorical variables like protocol type, service and flag into numerical forms. Once the categorical variables had been coded, all the numerical variables were scaled by Min-Max scaling to cut the feature values within a common range of 0 to 1. This process of normalization enhances convergence of the model to the training process and ensures that the high numeric range features do not dominate the learning process.

In the CICIDS2017 dataset, the dataset is mainly composed of numerical flow-based features that are derived in the form of packet capture (PCAP) files. The missing values and the infinite values were first determined during preprocessing to ensure the integrity of

the records. Min-Max scaling was then applied to the cleaned data to normalize the data to guarantee that the distribution of the features in the training data were similar. This is especially necessary since the CICIDS2017 data has many features that have different scales.

The data sets were evaluated by means of splitting them into training and testing after preprocessing. Each dataset was stored separately with separate scalers and preprocessing artifacts to make sure that the same feature transformations are used in the prediction phase. This will ensure that there is consistency between the training and inference pipelines.

The machine learning and deep learning models in the hybrid ensemble framework are then fed with the processed feature vectors as input. The proposed NIDS system provides the reliability of the features representation by performing the relevant preprocessing methods on each dataset and enhances the performance of the intrusion detection models.

## 4 Experimental Result

### 4.1 Experimental Setup

The effectiveness of the suggested NIDS framework was measured through experiments on two common benchmark datasets, including NSL-KDD and CICIDS2017. These datasets are the various generations of network intrusion detection data and offer various traffic patterns and attack scenarios.

In the case of NSL-KDD data, the models were trained on the file KDDTrain+.txt and tested on the file KDDTest+.txt that contains 22,544 samples of network traffic. This data set contains normal traffic and various categories of attacks which are denial-of-service (DoS), probing, remote-to-local (R2L), and user-to-root (U2R) attacks.

In the case of the CICIDS2017 dataset, the evaluation was done on network flow records derived out of the packet capture files. The sample size selected to test the models was 50,000 since the dataset is large. The data set comprises of contemporary network attacks such as distributed denial-of-service (DDoS), brute force attacks, web attacks, infiltration and port scanning.

The models were measured based on standard classification measures such as accuracy, precision, recall, and F1-score. Accuracy is used to measure the overall percentage of the correctly classified cases and precision and recall are used to measure the capability of the model to identify the intrusion events correctly. The F1-score is a balanced score that takes into account both the precision and the recall.

### 4.2 Model Performance Comparison

The classification accuracy of the evaluated machine learning and deep learning models on both datasets is summarized in Table 1.

The findings suggest that the models perform differently with different datasets. In the case of NSL-KDD dataset, the Support Vector Machine (SVM) classifier had the best accuracy of 78.67 and the LSTM model had the second best accuracy of 78.06. The hybrid ensemble model attained 76.50% which illustrates good performance in terms of prediction on various classifiers.

In the case of CICIDS2017, the most accurate model using the CICIDS2017 was the Random Forest classifier with the highest accuracy of 99.91, and the CNN model with the highest accuracy of 99.36. The hybrid ensemble model proposed has a 99.42% accuracy and it has been proven to work well in the contemporary network traffic situations.

Model accuracy comparison between the two datasets is shown in Fig. 2 which gives a graphical description of the performance variations between the considered models.

**Table 1.** Performance comparison of different models on NSL-KDD and CICIDS2017 datasets

| Model | NSL-KDD Accuracy (%) | CICIDS2017 Accuracy (%) |
|---|---|---|
| CNN | 74.20 | 99.36 |
| LSTM | 78.06 | 98.30 |
| KNN | 77.63 | 98.85 |
| SVM | 78.67 | 94.40 |
| Random Forest | 77.31 | 99.91 |
| Naïve Bayes | 77.08 | 53.32 |
| Hybrid Ensemble | 76.50 | 99.42 |

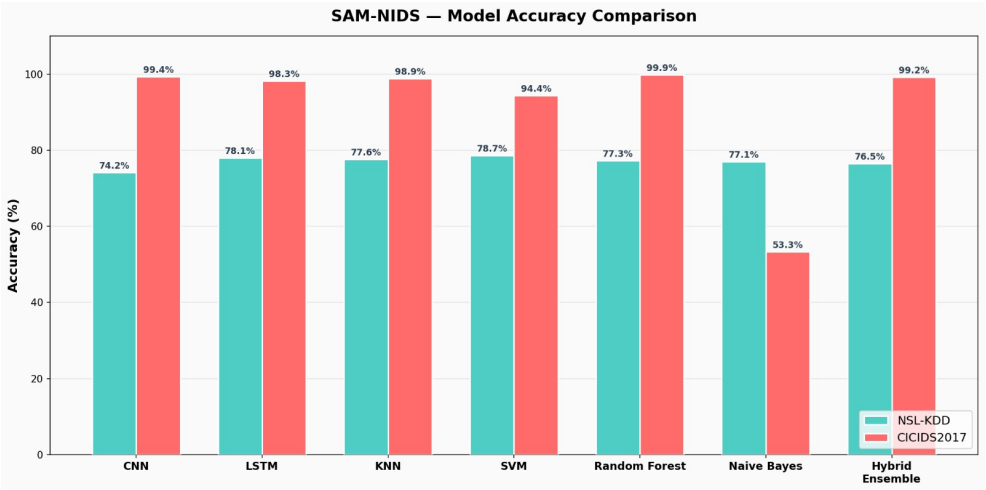


**Figure 2.** Accuracy comparison of machine learning and deep learning models on the NSL-KDD and CICIDS2017 datasets.

### 4.3 Performance Analysis

Besides accuracy, several evaluation metrics were used to better understand the detection capability of each model. A comparison of accuracy, precision, recall, and F1-score for the evaluated classifiers is shown in Fig.3.

According to the results, most models achieved better performance on the CICIDS2017 dataset than on the NSL-KDD dataset. This difference can be attributed to the richer feature representation and well structured flow based traffic data available in the CICIDS2017 dataset.

In order to further assess the performance of the classification, confusion matrices have been evaluated on each of the models. The confusion matrices show the count of the correctly classified normal and intrusion samples plus the misclassifications as shown in Fig.4. As demonstrated by the confusion matrices, certain models have a higher rate of misclassifying intrusion samples in the NSL-KDD dataset and that is why the overall accuracy is relatively lower.

The findings with the CICIDS2017 data illustrate much better classification results. Specifically, the hybrid ensemble models and the Random Forest have almost perfect classification scores, and the false positive and false negative are minimal.

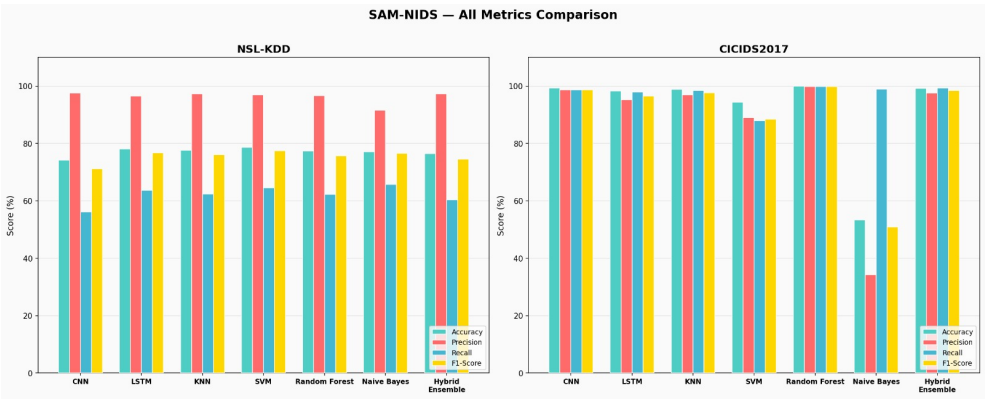


**Figure 3.** Comparison of classification metrics for ML and DL models on NSL-KDD and CICIDS2017 datasets.

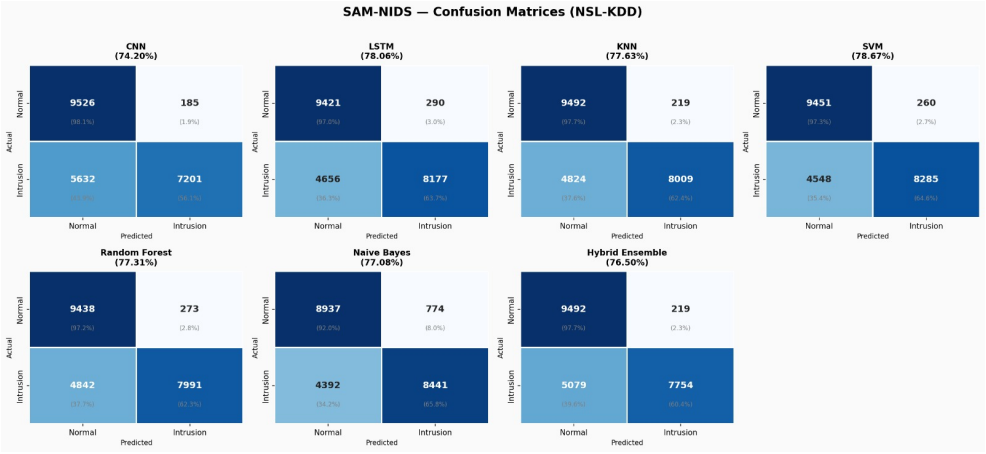


**Figure 4.** Confusion matrices of evaluated models on the NSL-KDD dataset.

In general, the experimental findings indicate that the proposed NIDS hybrid ensemble framework offers stable and reliable intrusion detection performance on both traditional and modern network traffic data. When machine learning and deep learning models are integrated, the system will have the advantages of having multiple classifiers in effect and a high performance with regard to detection.

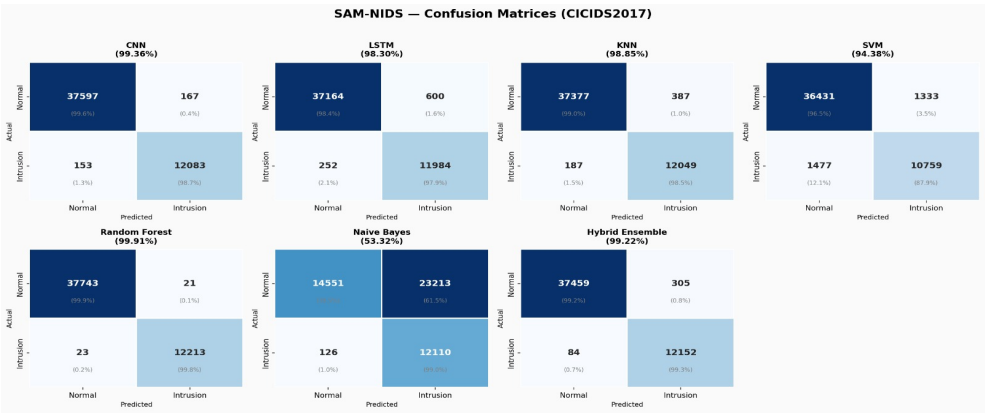


**Figure 5.** Confusion matrices of evaluated models on the CICIDS2017 dataset.

## 5 Conclusion

The current paper introduced NIDS, a machine learning and deep learning hybrid framework that is applicable in network intrusion detection. The proposed system incorporates a weighted ensemble structure that incorporates six classification models, such as CNN, LSTM, KNN, SVM, RF, and NB. The framework is able to use the strengths of both machine learning and deep learning approaches to enhance detection robustness and general system reliability.

NSL-KDD and CICIDS2017 are two popular benchmark datasets which were used to evaluate the performance of the proposed framework in terms of traditional and modern network traffic environment. The experimental findings also showed a competitive detection performance in both datasets. Specifically, SVM classifier was the most accurate on the NSL-KDD dataset (78.67%), whereas the best result on the CICIDS2017 dataset was the Random Forest classifier (99.91%). The hybrid ensemble model proposed reached 76.50% accuracy on NSL-KDD and 99.42% accuracy on CICIDS2017, and the model is reliable in various network traffic conditions.

The findings show that an ensemble approach to machine learning and deep learning models can enhance the accuracy of intrusion detection systems. Also, the presented architecture illustrates that it is possible to combine AI based intrusion detection mechanisms into a complete stack web application.

The Future work will be based on the extension of the proposed framework to multi-class intrusion detection, the incorporation of real time network traffic monitoring, and the evaluation of the system with the help of other intrusion detection datasets. More studies can also be conducted on improved deep learning structures and dynamic ensemble methods to improve detection accuracy and scalability in dynamic network settings.

## References

[1] Abdulganiyu, Oluwadamilare Harazeem, Taha Ait Tchakoucht, and Yakub Kayode Saheed. "A systematic literature review for network intrusion detection system (IDS)." International journal of information security 22.5 (2023): 1125-1162.

[2] Ahmad, Zeeshan, et al. "Network intrusion detection system: A systematic study of machine learning and deep learning approaches." Transactions on Emerging Telecommunications Technologies 32.1 (2021): e4150.

[3] Ashiku, Lirim, and Cihan Dagli. "Network intrusion detection system using deep learning." Procedia Computer Science 185 (2021): 239-247.

[4] Mohammadpour, Leila, et al. "A convolutional neural network for network intrusion detection system." Proceedings of the Asia-Pacific Advanced Network 46.0 (2018): 50-55

[5] Sultana, Nasrin, et al. "Survey on SDN based network intrusion detection system using machine learning approaches." Peer-to-Peer Networking and Applications 12.2 (2019): 493-501..

[6] Wang, Hui, Zijian Cao, and Bo Hong. "A network intrusion detection system based on convolutional neural network." Journal of Intelligent Fuzzy Systems 38.6 (2020): 7623-7637.

[7] Almutairi, Yasmeen S., Bader Alhazmi, and Amr A. Munshi. "Network intrusion detection using machine learning techniques." Advances in Science and Technology Research Journal 16.3 (2022): 193-206.

[8] Magán-Carrión, Roberto, et al. "Towards a reliable comparison and evaluation of network intrusion detection systems based on machine learning approaches." Applied Sciences 10.5 (2020): 1775.

[9] Jmila, Houda, and Mohamed Ibn Khedher. "Adversarial machine learning for network intrusion detection: A comparative study." Computer Networks 214 (2022): 109073.

[10] Thaseen, I. Sumaiya, B. Poorva, and P. Sai Ushasree. "Network intrusion detection using machine learning techniques." 2020 International conference on emerging trends in information technology and engineering (IC-ETITE). IEEE, 2020.

[11] Abdulhammed, Razan, et al. "Features dimensionality reduction approaches for machine learning based network intrusion detection." Electronics 8.3 (2019): 322.

[12] Siddharth, K., et al. "A Comparative Analysis of Network Intrusion Detection Using Different Machine Learning Techniques." J Contemp Edu Theo Artific Intel: JCETAI-102 (2023).

[13] Apruzzese, Giovanni, Luca Pajola, and Mauro Conti. "The cross-evaluation of machine learning-based network intrusion detection systems." IEEE Transactions on Network and Service Management 19.4 (2022): 5152-5169.

[14] Alzahrani, Abdulsalam O., and Mohammed JF Alenazi. "Designing a network intrusion detection system based on machine learning for software defined networks." Future Internet 13.5 (2021): 111.

[15] Dong, Yuansheng, Rong Wang, and Juan He. "Real-time network intrusion detection system based on deep learning." 2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS). IEEE, 2019