Transactions on Engineering and Computing Sciences - Vol. 13, No. 05
Publication Date: October 29, 2025
DOI:10.14738/tmlai.1305.19509.
Malekzadeh, M., & Al_Akabi, D. M. A. (2025). An AI-based Intrusion Detection System for SQL Injection Attacks in Smart Home IoT Networks. *Transactions on Engineering and Computing Sciences*, 13(05). 103-125.

An AI-based Intrusion Detection System for SQL Injection Attacks in Smart Home IoT Networks

Mina Malekzadeh
Electrical and Computer Engineering Faculty,
Hakim Sabzevari University, Sabzevar, Iran

Dheyaa Mohammed Ali Al_Akabi
Electrical and Computer Engineering Faculty,
Hakim Sabzevari University, Sabzevar, Iran

ABSTRACT

The widespread adoption of Internet of Things (IoT) technologies has transformed smart home environments by enhancing automation, connectivity, and user-centric functionality. However, the advancements also introduce different security vulnerabilities, among which SQL injection attack has become notably pervasive and damaging. The attack involves adversaries injecting or manipulating SQL queries through insecure IoT device interfaces, corrupting databases or exfiltrating data, which lead to unauthorized access, data leakage, device control compromise, and cascading effects across connected systems. In response to this attack, this study proposes an Intrusion Detection System (IDS) to detect and mitigate SQL injection attacks in IoT-based smart home networks. To ensure resilience and performance, the proposed IDS framework integrates three principal categories of artificial intelligence algorithms based on their distinct methodological advantages: first, traditional machine learning techniques, which provide foundational classification and clustering capabilities with interpretable decision boundaries and efficient performance on structured data; second, boosting-based ensemble methods, which contribute enhanced predictive accuracy and robustness through iterative refinement and sensitivity to complex feature interactions; and third, deep learning architectures, which further enrich the system by enabling hierarchical feature extraction and temporal pattern modeling, particularly suited to high-dimensional and sequential intrusion data. The strategic integration of these algorithmic classes allows the IDS to leverage complementary strengths, resulting in improved detection accuracy and adaptability across diverse threat environments.

Keywords: Deep Learning, Ensemble Methods, Machine Learning, Smart Home IoT, SQL Injection Attacks.

INTRODUCTION

The widespread adoption of Internet of Things (IoT) technologies has resulted in the development of energy-saving, safe, and convenient smart homes. A smart thermostat, security cameras, and health monitors all connect to each other, and one can easily control and automate home processes remotely. This interdependent relationship comes with great advantages that include better energy management, security, and remote home conditions monitoring and control. The residents will be able to manage energy consumption maximally, minimize the expenditure of the operating costs, and achieve a more comfortable everyday experience with the help of IoT appliances [1]. As an example, smart thermostats can now regulate the temperature depending on whether or not there is occupancy, and security cameras offer real-time footage to maximize security. With the proliferation of IoT technology in everyday life, it can be utilized to transform the model of controlling and managing a living environment to benefit not just the quality of life but the overall efficiency as well [2].

Nevertheless, there are critical security issues involved in the mass adoption of IoT technology as well. Most IoT devices are produced with low computing capacity, which implies that they might not have strong security or might not undergo rigorous security testing. Such weaknesses render the IoT networks highly prone to cyber-attacks. The increasing proliferation of more of these devices and networks translates to the fact that such devices and networks are one of the most targeted by attackers as a way of seeking to leverage the vulnerabilities in how they are designed or implemented. The absence of proper security measures in the majority of IoT frameworks may provide attackers with an opportunity to crack the system and access sensitive information or control the devices, thus breaking the integrity and privacy of the users [3].

Among the current threats to IoT networks is the SQL injection attack that targets vulnerabilities in database-driven applications. The problem of the injection attacks refers to the situation when malicious input, such as SQL commands or shell scripts, is injected into a system by exploiting vulnerabilities in the system's ability for safe and valid input validation and sanitization in an attempt to gain access to sensitive data, disrupt system functionality, or compromise device integrity. In the context of smart homes, where IoT devices often rely on lightweight, embedded databases and lack robust security protocols, the impact of such attacks can lead to severe privacy intrusions, financial loss, or even bodily harm in the event that core systems like smart locks or surveillance cameras are compromised [4]. As an example, hacked smart locks may enable intruders to enter homes, and hacking surveillance cameras may result in security breaches. The more the IoT networks rise and become complex, the harder detection and prevention become [5].

Legacy Intrusion Detection Systems (IDS) have been utilized for decades as part of efforts to help in securing networks against attacks. Most of these systems, however, are signature-based or rule-based detection and are normally ineffective against emergent or sophisticated threats. Signature-based IDSs are only capable of identifying a known pattern of attacks, and as such, they prove to have been inadequate in identifying new and emerging attacks. Second, there is the challenge of rule-based systems, which, as helpful as they are in the detection of pre-specified anomalies, do not play well with the highly dynamic and highly variable nature of IoT environments. Since IoT devices are continuously introduced and removed to and out of the network, and the form of data passed is different, the traditional IDS approaches are insufficient when it comes to detecting emergent assaults in real-time [6].

With these constraints, there exists an increasing requirement for smarter and responsive solutions to protect IoT networks. A solution to this problem is a smart IDS that involves artificial intelligence algorithms (AI) to efficiently differentiate malicious data injection. Such models can handle large amounts of network traffic, learn regular behavior patterns, determine anomalies, and recognize complicated patterns of the injection attacks [7]. Accordingly, the main contribution of this work is to propose an adaptive AI-based IDS framework to protect smart home IoT networks against SQL injection attacks. Recognizing the limitations of single-method approaches, the proposed IDS framework strategically integrates three distinct categories of artificial intelligence algorithms, each selected for its methodological strengths and domain-specific advantages. The first category comprises conventional machine learning techniques, including K-Nearest Neighbors (KNN), K-means, Support Vector Machines (SVM), Random Forest (RF), Decision Trees (DT), Logistic regression (LogR), and Linear Regression (LinearR). These models offer interpretable decision boundaries and efficient baseline performance across structured datasets. The second category encompasses boosting-based ensemble methods such as Gradient Boosting (GB), Adaptive Boosting (AdaBoost), Hist Gradient Boosting (HistGB), Extra Trees (ET), Extreme Gradient Boosting (XGBoost), and Light Gradient Boosting Machine (LightGBM), which are known for their capacity to reduce bias and variance through iterative refinement and feature sensitivity. The third category integrates deep learning architectures, namely Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), Gated Recurrent Units (GRU), and Deep Neural Networks (DNN), which enable hierarchical feature extraction and temporal pattern recognition, particularly effective in modeling complex, high-dimensional intrusion data. By leveraging the complementary capabilities of these algorithmic classes, the IDS aims to achieve robust detection accuracy, scalability across heterogeneous device ecosystems, and adaptability to evolving threat landscapes. This integration not only advances the technical resilience of smart home networks but also contributes to the broader discourse on AI-driven cybersecurity solutions for IoT environments.

RELATED WORKS

Network security has enjoyed a special degree of enhancement in the intrusion detection field, which has been necessitated by the need to protect critical systems against dynamic cyber-attacks. Intrusion Detection Systems (IDS) have been under research disturbance as there is a penetration through machine learning and deep learning to enhance accuracy, efficiency, and scalability of intrusion detection systems. More specifically, the study has explored new methods of identifying anomalies in network traffic and avoiding attacks on both traditional networks and the emerging IoT networks.

Singh et al. in 2015 [8] suggested an online sequential extreme learning machine (OS-ELM) approach for network traffic classification using the NSL-KDD 2009 and Kyoto University benchmark datasets. Their method achieved greater accuracy, reduced false positive rates, and better detection times than current techniques. However, one of the limitations is that applying OS-ELM on actual networks with high traffic volumes may be problematic in terms of scalability and performance. In 2016, Cai et al. [9] combined particle swarm optimization (PSO) and multi-criteria and time-varying chaos, and support vector machines (SVM) and linear programming to enhance intrusion detection. Their approach achieved high rates of detection and low false alarm rates, which were greater than both traditional PSO and Chaos PSO (CPSO). The main shortcoming of this approach is reliance on the particle swarm optimization algorithm, which might not be the best under all circumstances in dynamic networks.

In 2017, Kabir et al [10], an optimal allocation-based least square support vector machine (OA-LS-SVM) intrusion detection that was run on the KDD dataset. The method was efficient in accuracy and real-world applicability and can be used practically in real-world environments. However, its use in large-scale networks is questionable, and the efficiency of the method is largely dependent on the quality of the feature set it is trained with, which may not be applicable across all network landscapes. In 2020, Wu et al. [11], semantic re-encoding and deep learning-based methodology, i.e., via ResNet, is proposed for intrusion detection. The result demonstrates that SRDLM algorithm outperforms traditional machine learning techniques by more than 8% and detects web character injection network attacks with more than 99% accuracy through the use of the NSL-KDD dataset. However, the deep learning model-based dependence of the approach may result in computation expenses, so its application in resource-constrained environments could be limited. In 2021, Thakur et al. [12], a domain-specific and generic autoencoder-based model for network intrusion detection is proposed. High precision, recall, specificity, and F1 scores are the outcomes. While these are promising results, the disadvantage is that training the autoencoders might be tricky in some cases where the data is grossly unbalanced, with some implications on the performance of the model.

In 2022, Khanday et al. [13], the authors propose a lightweight IDS using novel data pre-processing and machine learning and deep learning classifiers like Linear SVC, Naïve Bayes, Logistic Regression, ANN, and LSTM. The experiments identify that LSTM and ANN models were optimal for both binary and multi-class classification with 99% and 95% accuracy, respectively. Despite these promising results, the weakness is in the computational requirement in training deep learning models, particularly LSTM, in limited environments such as IoT devices. In 2023, Xu et al. [14] proposed a machine learning-based automated approach for intrusion and anomaly detection in the Internet of Things (IoT). Their approach achieved an incredible accuracy of 99.7%, outdoing existing methods available for multi-class classification problems. While such accuracy is promising towards securing IoT, its limitation is that it was trained on the KDDcup99 dataset, which may not accurately represent the more sophisticated or dynamic patterns of attacks encountered in real-world IoT contexts. Viegas et al. [15] also aimed at designing precise evaluation models for intrusion detection based on networks, comparing models such as Decision Trees (DT), Random Forest (RF), SVM, Naïve Bayes (NB), Artificial Neural Networks (ANN), and Deep Neural Networks (DNN). Based on their analysis, they found DNN to create the most precise model with 99% accuracy, followed by the NB model with the lowest false positive rate. The limitation of this study is that the DNN model is overfit, which brings down its usefulness when dealing with noisy or unbalanced data.

In 2023, Xu et al. [16], a machine learning strategy is employed to train an Intrusion Detection System (IDS) for Internet of Things (IoT) devices by utilizing models such as CNN-BiLSTM, CANET, FNN-Focal, RFS-1, XGBoost, and XGBoost-HPO. The results indicate that the proposed models achieved perfect accuracy, precision, recall, and F1 score with an excellent performance of 1.0 in all metrics. But the limitations are the utilization of various datasets (BoT-IoT, NSL-KDD, WUSTL-IIOT2021, N-BaIot, WUSTL-EHMS-2020), which may not be able to reflect the heterogeneity of real-world IoT environments. In 2023, Vishwakarma et al. [17], a two-stage IDS using Naïve Bayes classification and an elliptic envelope technique for anomaly detection is proposed. The result achieves good accuracy, 97.5% for the NSL-KDD dataset, 86.9% for the UNSW_NB15 dataset, and 98.59% for the CIC-IDS2017 dataset. However, the performance is still dependent on the dataset and the method can be ineffective in applying to new attack channels or heterogeneous network conditions so that its usefulness will be limited in dynamic environments. In 2023, Rangelov et al [18], the authors attempt to develop an integrated toolchain and methodology for IoT platforms and networks in urban environments. The study focuses on the real-world deployment and optimization of IoT security solutions on a continuous basis in real-world urban settings. While the methodology provides promising concepts for real-world applications, the lack of concrete performance metrics or evaluations in the absence of a provided dataset makes it difficult to measure the performance and scalability of the method.

In 2023, Maddu & Rao [19], feature extraction from CenterNet, ResNet152v2 classification, and DCGAN data augmentation are combined to optimize intrusion detection performance. The model is seen to achieve 99.65% accuracy when tested with the InSDN dataset and 99.31% with the Edge IIoT dataset. While excellent performance is attained, the approach may struggle in real-time scenarios with the computational cost in both deep learning training and data augmentation. In 2022, Escandón et al. [20], a hybrid DoS attack intrusion detection system (HDA-IDS) based on CL-GAN, CNN-LSTM, and GAN for attack detection is proposed. The performance is enhanced by 5% in accuracy, precision, recall, and F1-Score compared to earlier studies. Despite the positive outcomes, the limitations are that it is challenging to train GAN models for hybrid DoS attacks, which are computationally expensive and difficult to generalize across attacks. In 2023, Wadate et al. [21], the authors propose an edge-based intrusion detection system (IDS) with machine learning on IoT networks, such as Backpropagation (BP) neural networks, Basis Function Radial (BFR) neural networks, and the Simulated Annealing algorithm. The results show that the proposed IDS gives a tremendous 93% accuracy, in comparison to the Naive Bayes detection model. However, its limitations are its relatively lower performance by Naive Bayes over the BP neural network, which might not be optimal in processing the complexity of IoT network traffic.

In 2023, Talukder et al. [22], a novel hybrid system is proposed using machine learning (ML) and deep learning (DL) in order to maximize detection rates without sacrificing reliability, utilizing SMOTE for data balancing and XGBoost for feature extraction. The precision is extremely high, at 99.99% on the KDDCUP'99 and a perfect 100% on the CIC-MalMem-2022 datasets. High as these are, though, the limitations are the danger of the model fitting the provided datasets too well and not working as well under more diverse or real-world scenarios. In 2023, Zakariah et al. [23], explored the use of machine learning-based adaptive synthetic sampling for intrusion detection in the form of LSTM and CNN. Their results showed that the MLP classifier achieved 87% binarization classification accuracy, 89% F1 score in attack detection, and 83% for all classes, demonstrating the applicability of the method. The main drawback is that training LSTM and CNN models involves complex processes, which can result in higher processing time and computational complexity. In 2023, Gu & Lu [24], proposed the combination of Support Vector Machines (SVM) and Naïve Bayes feature embedding for improved data quality and detection rates. While tested using UNSW-NB15 and CICIDS2017 datasets, the method achieved high accuracy and detection rates with comparably low false alarm ratios. The most significant constraint is its reliance on a predefined feature set that can possibly refuse to evolve with time-changing attack patterns. In 2024, Ngo et al. [25], analyzed a machine learning-based intrusion detection technique integrating feature selection and extraction techniques with classifiers like Decision Trees (DT), Random Forest (RF), K-Neighbors, Multi-layer Perceptron (MLP), and Naïve Bayes (NB). The experiments highlighted the advantages of feature extraction over feature selection, especially when the parameter K is low (e.g., K=4). MLP was determined to be the optimal-performing classifier for feature extraction. The disadvantage of their approach is that it depends on the classifier used, as performance can be susceptible to change in more heterogeneous network environments.

Building on the limitations identified in prior work, it becomes evident that existing IDS solutions, particularly those relying solely on signature-based or rule-based mechanisms, struggle to keep pace with the dynamic and heterogeneous nature of smart home IoT environments. These systems often fail to detect novel or obfuscated SQL injection attacks, especially when deployed across resource-constrained devices with inconsistent data flows and evolving threat patterns. Moreover, approaches that rely on a single category of AI algorithms tend to suffer from trade-offs between interpretability, detection accuracy, and adaptability. To address these shortcomings, our proposed framework introduces a multi-tiered AI-based IDS that synergistically combines conventional machine learning, boosting-based ensembles, and deep learning architectures to enhance detection robustness.

MATERIALS AND METHODS

To effectively detect SQL injection attacks while maintaining computational efficiency on resource-constrained smart home devices, the proposed IDS architecture follows a multi-step process, as illustrated in Fig. 1.

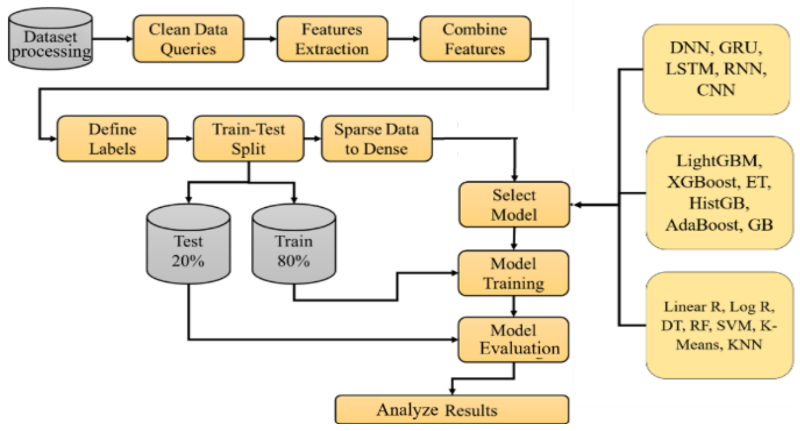

Fig 1: IDS architecture

Dataset Processing

The dataset used to detect SQL injection attacks [26] has 30,919 instances and two columns with names Query and Label. Raw SQL queries are stored in the Query column, which is either malicious or non-malicious. The goal is the Label column whose values are 0 for non-malicious and 1 for malicious queries. This data set has no missing values and a class imbalance, with 36.81% of the queries being malicious and 63.19% being non-malicious. The data is of moderate size, so it is simple to train and test on, while no imputation is needed due to a lack of missing values. The dataset processing includes 4 steps: clean data, feature extraction, feature combination, and label definition. The clean query function is utilized to convert all the query strings to lowercase for consistency, since SQL is case-insensitive. Also, unwanted characters that are not included in the SQL characters, such as letters, numbers, spaces, single quotes, semicolons, and commas, are removed from the query strings via regular expressions. This eliminates noise in the query strings, allowing the model to learn the meaningful parts of the query strings, thus improving the accuracy of the classification. Normalization is also done on the SQL queries for enhanced performance.

Lexical and semantic analysis of SQL query strings serves as a feature extraction technique that identifies patterns typically associated with SQL injection attacks. This process helps uncover anomalous syntax structures, keyword misuse, and semantic inconsistencies that signal malicious intent. Some of the lexical features extracted include query length, count of quotes, count of dashes, count of semicolons, and occurrence of SQL keywords. These can identify structural patterns in SQL queries indicative of malicious use. Secondly, the semantic features are also obtained by applying the Term Frequency-Inverse Document Frequency (TF-IDF) method, converting the query strings into vector values. A combination of features from the lexical and semantic features includes a comprehensive representation of the query, supporting the model in classifying the SQL queries accurately as malicious or non-malicious.

After the extraction of lexical and TF-IDF features, they are merged together to form a full set of features. The TF-IDF vectorization-generated sparse matrix is horizontally concatenated with lexical features such as query length and quote count. The resultant integrated feature matrix incorporates structural and semantic information into the model, such that the model can encode a wide range of features of the SQL queries. By combining these two types of features, the model acquires a clearer picture of the queries, thus distinguishing between the malicious and non-malicious queries more effectively. The target variable of the model is the Label column of the dataset, indicating whether a query is malicious (1) or not (0). The label is employed as ground truth while training, which guides machine learning models to classify SQL queries correctly. The correct identification of labels is crucial as it directly impacts the learning process of the model, which leads to the models being trained to recognize malicious query patterns.

Train-Test Split

The data is separated into a training and a test set to estimate the performance of models. The data is split by using the train-test split function in a way that 20% of the data is reserved for testing, and 80% is used for training models. The `stratify=y` parameter maintains class distribution (malicious and non-malicious queries) in both the test and training set. This stratification becomes even more significant with regard to the class imbalance of the dataset to ensure that both the training set and test set possess an adequate representation of malicious as well as non-malicious queries.

Sparse to Dense Conversion

The majority of machine learning models do not accept sparse matrices, though a few of them can accept them. Therefore, sparse matrices of training and test sets are converted into dense arrays using the provision of the array method. This mapping makes the machine learning models interoperable and enables the models to process the feature data in a usable format. Dense representations are more appropriate for most machine learning algorithms for proper training and testing.

Model Selection, Training, Evaluation

Several machine learning and deep learning models are selected for SQL query classification. Classical machine learning algorithms used include KNN, K-means, SVM, RF, DT, LogR, LinearR, which are favored because of their interpretable decision boundaries and efficient baseline performance across structured datasets. In addition to traditional ML, the boosting ensemble ML algorithms are also selected for this purpose. The selected boosting algorithms include GB, AdaBoost, HistGB, ET, XGBoost, and LightGB. The models are chosen because of their stability and ability to fit complex patterns of data. Additionally, deep learning algorithms, including CNN, RNN, LSTM, GRU, and DNN, are selected. These models are ideal for finding complex relationships within the data, especially in the event of sequential dependencies.

The training process begins with the selection of optimal hyperparameters for each model using established optimization techniques. Hyperparameter tuning is a critical pre-training step that enhances a model’s generalization capability and reduces the risk of overfitting, thereby improving performance on unseen data. In the context of resource-constrained smart home devices within IoT networks, hyperparameters are carefully configured to balance predictive accuracy with computational efficiency. An effective combination can substantially improve both the accuracy and generalization performance of the model. In this work, hyperparameter tuning is conducted using GridSearchCV within the IDS framework, which systematically evaluates a comprehensive grid of hyperparameter configurations. At each iteration, cross-validation is employed to assess performance, ensuring the selection of the most effective parameter set. Once the models and their optimized hyperparameters are finalized, training proceeds to classify SQL queries based on the extracted features. After training the models, the final step is the evaluation of their performance for the detection of SQL injection attacks. Their performance is then evaluated using key metrics, including confusion matrix, accuracy, precision, recall, F1-score, ROC AUC, AUC, log loss, specificity, and Cohen's Kappa to ensure robust detection of SQL injection attacks. The training parameters for conventional ML, boosting ensemble ML, and DL algorithms are detailed in Table 1, Table 2, and Table 3, respectively, along with structural parameters of DL algorithms in Table 4.

Table 1: Conventional ML training parameters

Model	Parameter	Value
LogR	solver	'liblinear'
	C (regularization)	1
SVM	C (regularization)	1
	kernel	'rbf' (default)
	gamma	'scale' (default)
K-Means	n_clusters	3-10 (depends on data)
	init	'k-means++'
	max_iter	300
	n_init	10
KNN	n_neighbors	5
	algorithm	'auto'

Table 2: Boosting ensemble ML training parameters

Model	Hyperparameter	Values
GB	n_estimators	50, 100, 150
	learning_rate	0.01, 0.1, 0.2
	max_depth	3, 5, 7
AdaBoost	n_estimators	50, 100, 150
	learning_rate	0.01, 0.1, 1.0
HGB	max_iter	100, 200
	learning_rate	0.01, 0.1, 0.2
	max_depth	3, 5, 7
ET	n_estimators	50, 100, 150
	max_depth	3, 5, 7
	min_samples_split	2, 5
XGBoost	n_estimators	50, 100, 150
	learning_rate	0.01, 0.1, 0.2
	max_depth	3, 5, 7
LightGBM	n_estimators	50, 100, 150
	learning_rate	0.01, 0.1, 0.2
	max_depth	3, 5, 7

Table 3: DL training parameters

Model	Layer	Details
CNN	Input Layer	Shape = (X_train_dense.shape[1], 1)
	Conv1D Layer	64 filters, kernel size = 3, activation = 'relu'
	MaxPooling1D Layer	Pool size = 2
	Flatten Layer	Flatten the output of the previous layer
	Dense Layer	64 units, activation = 'relu'
	Dense Layer	1 unit, activation = 'sigmoid'
RNN	Input Layer	Shape = (X_train_dense.shape[1], 1)
	SimpleRNN Layer	64 units, activation = 'relu'
	Dense Layer	1 unit, activation = 'sigmoid'
LSTM	Input Layer	Shape = (X_train_dense.shape[1], 1)

	LSTM Layer	64 units, activation = 'relu'
	Dense Layer	1 unit, activation = 'sigmoid'
GRU	Input Layer	Shape = (X_train_dense.shape[1], 1)
	GRU Layer	64 units, activation = 'relu'
	Dense Layer	1 unit, activation = 'sigmoid'
DNN	Input Layer	Shape = (X_train_dense.shape[1],)
	Dense Layer	128 units, activation = 'relu'
	Dropout Layer	Dropout rate = 0.5
	Dense Layer	64 units, activation = 'relu'
	Dense Layer	32 units, activation = 'relu'
	Dense Layer	1 unit, activation = 'sigmoid'

Table 4: DL structural parameters

Model	Hyperparameter	Possible Values
RNN, LSTM, DNN, GRU, CNN	Units	32, 64, 128
	Dropout Rate	0.5 (fixed)
	Epochs	10
	Batch Size	32, 64

RESULTS AND DISCUSSIONS

This section outlines the experimental results obtained from the implementation of the proposed IDS framework in Python. It presents a comparative analysis of each model's performance within the IDS framework, evaluating their effectiveness in detecting malicious SQL injection attacks and enhancing the security of smart homes in IoT environments.

Detection Successes: True Positives and Negatives

True Positive (TP) reflects the IDS's ability to detect genuine threats. When a malicious input is submitted to the IoT network and the IDS correctly identifies and flags it as a SQL injection attack, it contributes to the TP count. A high TP rate indicates robust detection capabilities of malicious queries. In contrast, True Negative (TN) denotes the correctly classified benign input as non-malicious and allows it to proceed without intervention by the IDS. When a legitimate entry is accurately classified as safe, it contributes to the TN count. A high TN rate suggests that the system maintains usability by minimizing unnecessary alerts or disruptions to legitimate users. Together, from TP and TN values, we have an insight into the ability of the proposed IDS framework to differentiate between malicious and benign SQL queries, especially identifying non-target cases. The results of detection successes in terms of TP and TN are illustrated in Fig. 2.

The comparative evaluation of SQL injection detection models based on TP and TN results reveals distinct performance characteristics across different models in the IDS framework. Conventional ML models exhibited varied detection capabilities. SVM and LogR achieved high TP scores (8463 and 8190, respectively), indicating strong sensitivity to attack patterns. However, their TN values were notably low (137 and 10), suggesting limited ability to correctly identify benign inputs. KNN demonstrated more balanced results, with 5861 TP and a substantially higher TN of 3415, reflecting effective discrimination between malicious and legitimate traffic.

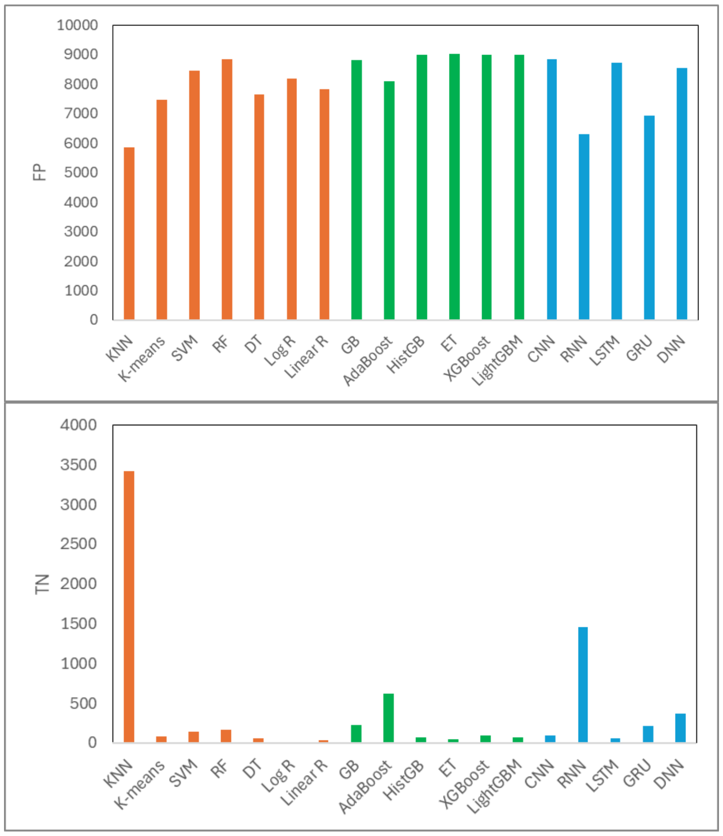

Fig 2: True Detection Analysis: TP and TN

In contrast, KMeans, DT, and LinearR recorded moderate TP values (7482, 7656, and 7832) but suffered from low TN scores (82, 56, and 32), indicating a tendency toward over-detection and reduced generalization. Comparatively, boosting-based ML models consistently achieved high TP rates. ET recorded the highest TP overall (9032), followed closely by XGB (8999), HistGB (8999), and LGBM (9011). However, TN values varied significantly. ET and HistGB had relatively low TN scores (50 and 75), while GB and AdaBoost offered more balanced results (TP: 8822 and 8109; TN: 223 and 622). These findings suggest that while boosting models are highly effective at identifying SQL injection attacks, their ability to correctly classify benign traffic depends on the specific ensemble strategy employed. DL models also demonstrated competitive TP performance, with CNN (8846), LSTM (8736), and DNN (8553) performing strongly. RNN achieved the highest TN among all models (1459), though its TP score (6319) was comparatively lower, indicating a conservative detection approach that prioritizes minimizing false positives. GRU offered intermediate results (TP: 6945; TN: 214), balancing detection sensitivity with benign input recognition. Accordingly, the results suggest that the boosting models such as ET, XGB, and LGBM excelled in TP detection, affirming their effectiveness in identifying SQL injection attacks. However, models like KNN and RNN demonstrated superior TN performance, highlighting their strength in preserving legitimate traffic flow. The observed trade-offs between TP and TN across models underscore the importance of aligning algorithm selection with operational priorities, whether emphasizing aggressive threat detection or minimizing disruption to benign users.

Detection Failures: False Positives and Negatives

In evaluating the performance of SQL injection detection systems, False Positive (FP) and False Negative (FN) metrics are critical for understanding the limitations and operational risks associated with model predictions. FP refers to when the IDS mistakenly flags a valid, benign input as a SQL injection attack, which contributes to the FP count. This type of error can lead to unnecessary blocking of legitimate user activity, reduced system usability, and increased administrative overhead due to false alarms. High FP values indicate that the IDS has the habit of over-predicting positive instances, while low FP values demonstrate that it is more precise in its positive predictions. In contrast, FN denotes cases where the system fails to identify an actual SQL injection attempt, allowing the malicious input to pass undetected. This error poses a direct threat to system integrity, as it enables attackers to exploit vulnerabilities without triggering defensive mechanisms. An effective SQL injection detection system seeks to minimize both FP and FN error types to ensure robust security while maintaining operational efficiency. Thereby, we implement the IDS to determine FP and FN and thereby provide essential insight into the trade-offs between detection sensitivity and specificity. The results of detection failures are in terms of FP and FN are illustrated in Fig. 3.

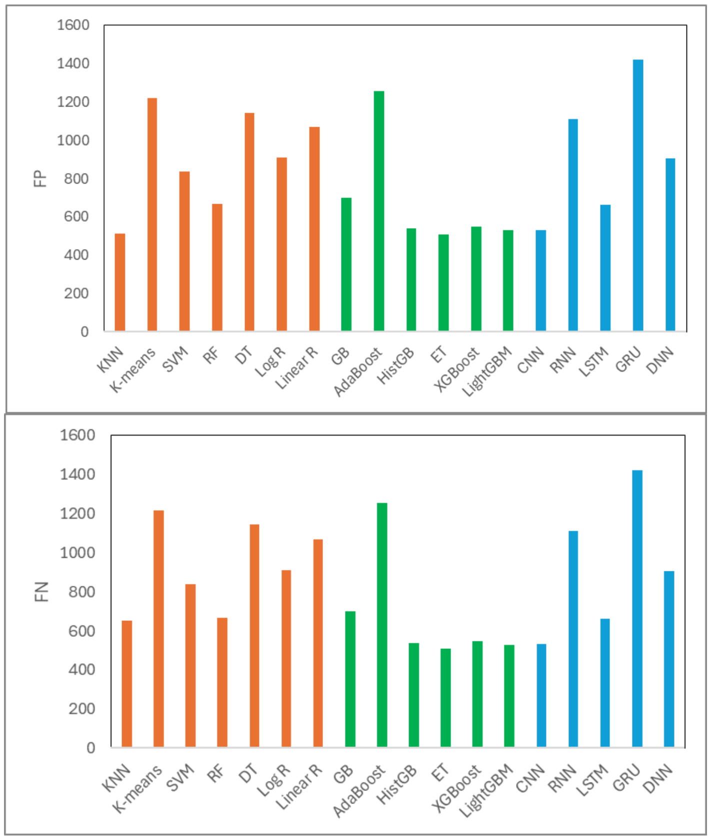

Fig 3: True Detection Analysis: FP and FN

Based on the results, conventional ML models demonstrate varied error rates. KNN records relatively low FP and FN values (510 and 651), indicating a balanced detection strategy with moderate misclassification. In contrast, KMeans, SVM, DT, LogR, and LinearR produce higher FP and FN counts, ranging from 837 to 1218, which suggests reduced precision and recall. Notably, DT and LinearR show elevated error values (1144 and 1068), potentially reflecting overfitting or limited generalization to unseen data. Furthermore, boosting-based ML models show a tendency to better control errors. ET and HistGB yield the lowest FP and FN values among all models (509 and 538, respectively), indicating strong classification reliability. XGB and LGBM also perform well, with both FP and FN scores below 550, validating their capacity to minimize false alarms and missed detections. GB and AdaBoost present slightly higher error rates (700 and 1257), with AdaBoost exhibiting the highest FP and FN among boosting models, which may indicate sensitivity to noise or class imbalance.

As for DL models, they achieve competitive error reduction. CNN, LSTM, and DNN maintain FP and FN values between 531 and 906, reflecting consistent performance in both detection and generalization. RNN and GRU, however, record higher error values (1111 and 1421), which show that there is a trade-off between temporal modeling and classification precision. GRU's high FP and FN indicate challenges in distinguishing subtle patterns between benign and malicious inputs. In summary, boosting models such as ET, HistGB, XGB, and LGBM demonstrate superior error minimization, while DL models like CNN and LSTM offer stable detection with moderate misclassification. Conventional ML models show greater variability, with KNN performing most effectively among them. These findings emphasize the importance of balancing sensitivity and specificity in model selection, particularly in security-critical applications where both FP and FN carry significant operational consequences.

Accuracy Analysis

Accuracy measures the proportion of correct predictions out of all predictions made by the IDS. It provides a rough notion of how the IDS is performing on all classes. In this work, accuracy is used for evaluating the overall effectiveness of SQL injection detection of the IDS models in correctly classifying malicious and non-malicious queries. The comparative analysis across conventional machine learning (ML), boosting-based ML, and deep learning (DL) models is provided in Fig. 4.

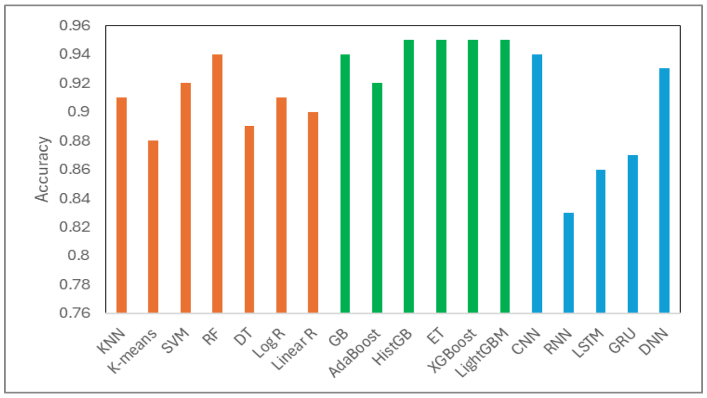

Fig 4: IDS Accuracy Analysis

Conventional ML models demonstrate moderate to high accuracy. SVM achieves the highest accuracy within this group (0.92), followed closely by KNN and LogR (0.91 each). LinearR and DT record slightly lower scores (0.90 and 0.89), while KMeans yields the lowest accuracy among conventional models (0.88), indicating limited generalization capacity in unsupervised settings. By contrast, boosting-based ML models consistently outperform their conventional counterparts. HistGB, ET, XGB, and LGBM each reach an accuracy of 0.95, representing the highest scores across all evaluated models. GB and RF follow closely with 0.94, while AdaBoost attains 0.92. These results affirm the robustness of ensemble learning techniques in capturing complex patterns and reducing classification errors. Additionally, DL models exhibit competitive accuracy. CNN and DNN achieve scores of 0.94 and 0.93, respectively. LSTM reaches 0.86, while GRU and RNN record lower scores (0.87 and 0.83), suggesting that although recurrent architectures offer temporal modeling advantages, they may be more sensitive to training dynamics and data variability. Taken together, boosting models such as HistGB, ET, XGB, and LGBM demonstrate superior accuracy, highlighting their effectiveness in SQL injection detection tasks. DL models like CNN and DNN also perform strongly, while conventional ML models show greater variability. These findings underscore the importance of informed model selection in optimizing detection performance, particularly in security-critical environments where classification precision directly influences system integrity and operational resilience.

Precision Analysis

Precision serves as a critical metric in the evaluation of SQL injection attack detection, as it quantifies the proportion of correctly identified attacks among all instances classified as malicious SQL queries. A high precision value reflects the model's capacity to minimize false positives, thereby reducing unnecessary disruptions to legitimate users and preserving overall system usability. Moreover, precision is particularly important, where excessive false alarms can degrade operational efficiency and user trust. Fig. 5 presents the precision scores of the conventional ML, boosting-based ML, and DL models implemented within the IDS framework, highlighting their comparative effectiveness in accurately identifying malicious inputs.

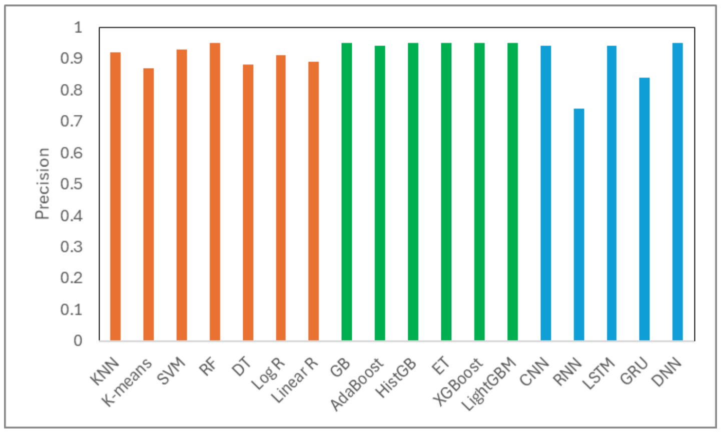

Fig 5: IDS Precision Analysis

Among conventional ML models, SVM achieves the highest precision (0.93), followed closely by KNN (0.92) and LogR (0.91), indicating strong reliability in distinguishing actual threats from benign inputs. In comparison, LinearR and DT yield slightly lower precision values (0.89 and 0.88), while KMeans records the lowest precision within this group (0.87), suggesting a greater tendency to misclassify benign traffic as malicious. In contrast, boosting-based ML models consistently demonstrate superior precision. RF, GB, HistGB, ET, XGB, and LGBM each attain a precision of 0.95, reflecting exceptional accuracy in attack identification with minimal false alarms. AdaBoost follows closely with a precision of 0.94, further reinforcing the effectiveness of ensemble learning techniques in enhancing classification confidence and reducing misclassification. Additionally, DL models exhibit competitive precision performance. CNN, LSTM, and DNN each achieve precision scores of 0.94 or higher, aligning closely with the top-performing boosting models. However, RNN and GRU record lower precision values (0.74 and 0.84), indicating increased susceptibility to false positives. These results may reflect the challenges recurrent architectures encounter in distinguishing subtle patterns within SQL injection payloads from benign input sequences. Taken together, boosting models such as RF, GB, HistGB, ET, XGB, and LGBM lead in precision, affirming their robustness in minimizing false alarms and maintaining classification integrity. Moreover, DL models like CNN, LSTM, and DNN perform strongly, offering reliable alternatives in high-stakes detection environments. Conventional ML models demonstrate moderate precision, with SVM and KNN emerging as the most effective within their category. These findings underscore the importance of precision in operational contexts where false positives can erode user trust and compromise system efficiency.

Recall Analysis

Recall, also referred to as sensitivity, quantifies the model's ability to correctly identify positive instances and is expressed as the proportion of true positives among all actual positive cases. This metric is particularly important where comprehensive detection is prioritized, even at the expense of a higher false positive rate. In the context of SQL injection attack detection, recall serves as a key measure of the effectiveness of the IDS framework in identifying malicious queries and preventing threats from going undetected. Fig. 6 presents the recall scores of the conventional ML, boosting-based ML, and DL models within the IDS framework, enabling a comparative assessment of their detection capabilities.

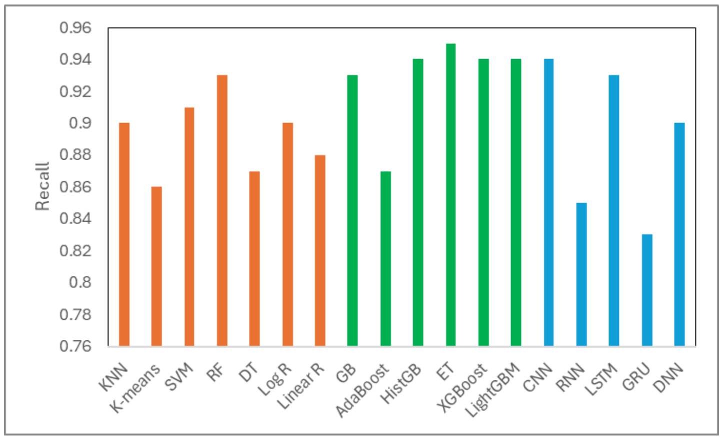

Fig 6: IDS Recall Analysis

Among conventional ML models, SVM achieves the highest recall (0.91), followed closely by KNN and LogR (0.90 each), indicating reliable detection of SQL injection attempts. In comparison, LinearR and DT yield slightly lower recall scores (0.88 and 0.87), while KMeans records the lowest recall within this group (0.86), suggesting a higher likelihood of false negatives and reduced threat coverage. In parallel, boosting-based ML models demonstrate consistently high recall performance. ET leads with the highest recall overall (0.95), followed by HistGB, XGB, and LGBM (0.94 each), and GB and RF (0.93 each). These results affirm the strength of ensemble learning techniques in capturing complex attack patterns and maintaining high detection sensitivity. AdaBoost, however, records a lower recall (0.87), indicating comparatively weaker performance in identifying all attack instances.

Alternatively, DL models perform competitively in terms of recall. CNN, LSTM, and DNN achieve scores of 0.94, 0.93, and 0.90, respectively, reflecting strong detection capabilities. By contrast, RNN and GRU record lower recall values (0.85 and 0.83), indicating a higher rate of missed attacks and potential limitations in temporal feature extraction or training stability. Taken together, boosting models such as ET, HistGB, XGB, and LGBM demonstrate superior recall, highlighting their effectiveness in minimizing false negatives. Moreover, DL models like CNN and LSTM perform strongly, offering reliable alternatives in high-sensitivity detection environments. Conventional ML models exhibit moderate recall, with SVM, KNN, and LogR emerging as the most effective within their category. These findings underscore the importance of recall in security-critical applications, where undetected threats can compromise system integrity and erode user trust.

F1-Score Analysis

The F1-score provides a balanced measure of a model's classification performance by harmonizing precision and recall. F1-score analysis interprets how the models are trading off between precision and recall, which is extremely critical in applications where both are vital to success. In the context of SQL injection detection, a high F1-score indicates that the model effectively identifies malicious input queries while minimizing both false positives and false negatives. The F1-score results across conventional ML, boosting-based ML, and DL models are compared in Fig. 7.

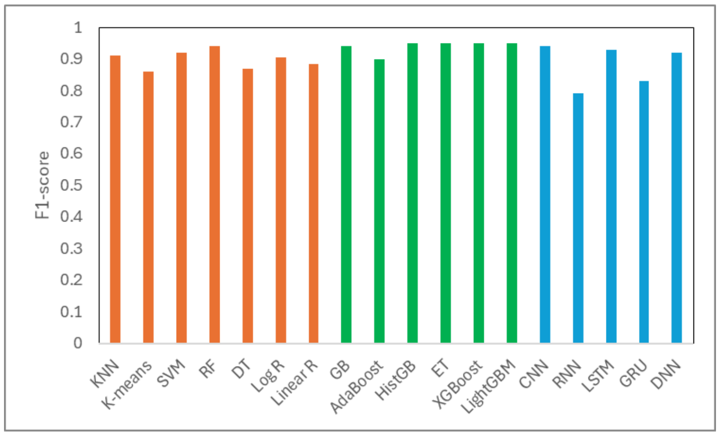

Fig 7: IDS F1-score Analysis

Among conventional ML models, SVM achieves the highest F1-score (0.92), followed by KNN (0.91) and LogR (0.905), indicating consistent performance in detecting malicious queries and correctly classifying benign inputs. LinearR and DT yield slightly lower scores (0.885 and 0.87), while KMeans records the lowest F1-score within this group (0.86), suggesting a diminished balance between sensitivity and specificity. Turning to boosting-based ML models, performance remains consistently strong. HistGB, ET, XGB, and LGBM each attain an F1-score of 0.95, demonstrating exceptional classification reliability and effective error minimization. GB and RF follow closely with scores of 0.94, while AdaBoost achieves 0.90. These results reinforce the capacity of ensemble learning techniques to capture complex attack patterns and maintain high detection fidelity.

DL models also exhibit competitive F1-score performance. CNN and LSTM reach scores of 0.94 and 0.93, respectively, while DNN achieves 0.92, placing them on par with the leading boosting models. Conversely, GRU and RNN record lower scores (0.83 and 0.79), indicating challenges in maintaining equilibrium between precision and recall, potentially due to instability during training or limitations in temporal feature extraction. Altogether, boosting models such as HistGB, ET, XGB, and LGBM lead in F1-score performance, confirming their effectiveness in SQL injection detection tasks. DL models like CNN, LSTM, and DNN also demonstrate a strong balance between precision and recall, offering reliable alternatives. Conventional ML models show greater variability, with SVM and KNN emerging as the most effective within their category. These findings highlight the value of the F1-score as a comprehensive metric for evaluating detection systems, particularly in security-critical environments where both false alarms and missed threats carry significant operational consequences.

ROC-AUC Analysis

Most classification metrics, such as accuracy, precision, or recall, depend on a specific decision threshold (e.g., classifying anything with a probability > 0.5 as positive). But that threshold can be arbitrary or suboptimal, especially in imbalanced datasets. The ROC-AUC metric, on the other hand, provides a threshold-independent measure of a model's ability to distinguish between positive and negative instances by evaluating a model's performance across all possible classification thresholds, rather than at just one fixed operating point. Higher ROC-AUC values indicate stronger classification performance and greater robustness to threshold selection. In the context of SQL injection detection, where class imbalance and varying decision boundaries are common, ROC-AUC captures how effectively models within the IDS framework differentiate between malicious and benign queries across varying decision thresholds. The corresponding ROC-AUC results obtained from the evaluated models are illustrated in Fig. 8.

Boosting-based ML models consistently demonstrate top-tier ROC-AUC performance. GB, HistGB, ET, XGBoost, and LightGBM each achieve a score of 0.95, indicating near-perfect class separation and exceptional robustness across decision thresholds. AdaBoost follows closely with a score of 0.94, reflecting strong discriminative capability, though with slightly more variability. Deep learning models also perform competitively. CNN, LSTM, DNN, and GRU each attain ROC-AUC scores of 0.95, 0.95, 0.95, and 0.92, respectively, aligning closely with the best-performing boosting models. RNN, while slightly lower at 0.90, still demonstrates solid classification reliability, though it may be more sensitive to sequential noise or training instability.

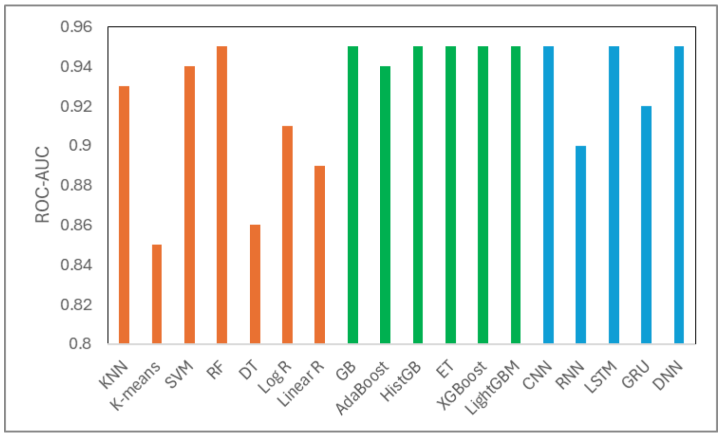

Fig 8: IDS ROC-AUC Analysis

Conventional ML models show more variability. RF and SVM lead this group with a score of 0.95 and 0.94, respectively, followed by KNN (0.93) and LogR (0.91), all reflecting strong discriminative power. LinearR and DT yield lower scores (0.89 and 0.86), while K-means records the weakest performance at 0.85, underscoring the limitations of unsupervised clustering in binary classification contexts. Overall, boosting models such as GB, HistGB, ET, XGBoost, and LightGBM demonstrate the highest ROC-AUC scores, confirming their reliability in SQL injection detection. DL models like CNN, LSTM, and DNN also perform strongly, offering well-calibrated alternatives. Conventional ML models exhibit moderate effectiveness, with SVM, KNN, and RF leading their category. These findings affirm ROC-AUC as a decisive metric for evaluating model performance in security-critical environments, where consistent class separation across thresholds is essential.

Log Loss Analysis

Log loss, also referred to as cross-entropy loss, quantifies the uncertainty in a model's predictions by penalizing incorrect classifications made with high confidence. This metric is particularly valuable when models are required to output well-calibrated probability estimates, such as in tasks where risk or uncertainty quantification is important. A lower log loss value indicates more reliable probabilistic outputs and reflects a model's ability to align its confidence levels with actual outcomes. Within the context of SQL injection detection, minimizing log loss is essential not only for accurate classification of input queries but also for ensuring that the system assigns appropriate confidence to its decisions. Fig. 9 illustrates the log loss values across conventional ML, boosting-based ML, and DL models, enabling comparative evaluation of their calibration performance in the IDS framework.

Among conventional ML models, SVM achieves the lowest log loss (0.03), followed closely by KNN (0.04), LogR (0.05), and LinearR (0.06), indicating well-calibrated predictions and stable confidence estimation. By comparison, DT and KMeans record higher log loss values (0.07 and 0.09), suggesting reduced certainty in their outputs and a greater tendency toward overconfident misclassifications. Ensemble-based ML models exhibit markedly stronger performance in minimizing log loss. GB, RF, and DNN each attain a value of 0.02, while HistGB, ET, XGB, and LGBM report negative log loss values (−0.03 or −0.02). These negative scores may result from numerical artifacts introduced by calibration techniques or regularization strategies that enhance predictive stability. Conversely, AdaBoost records the highest log loss (0.48), indicating poor calibration and a pronounced tendency toward overconfident errors.

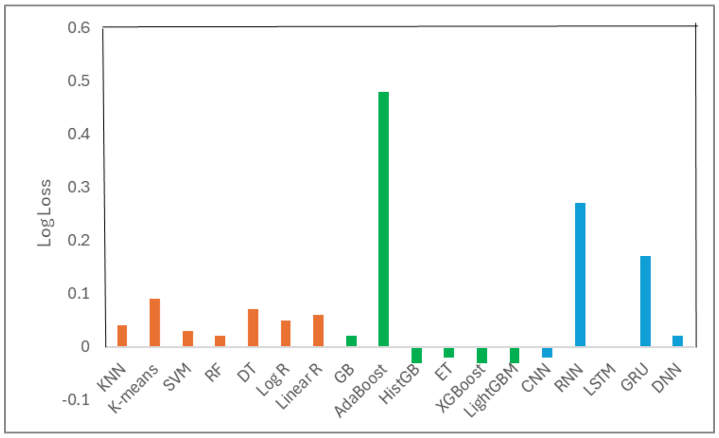

Fig 9: IDS Log Loss Analysis

Deep learning models also demonstrate competitive log loss performance. CNN and DNN match the best-performing boosting models with values of −0.02 and 0.02, respectively. LSTM achieves a log loss of 0, reflecting perfect calibration on the evaluated dataset. In contrast, RNN and GRU yield higher values (0.27 and 0.17), pointing to less stable confidence estimation and increased predictive uncertainty. When comparing all the models in the IDS framework, it is observed that the boosting models such as HistGB, ET, XGB, and LGBM deliver exceptional log loss performance, confirming their reliability in probabilistic classification. DL models like CNN, LSTM, and DNN also perform strongly, offering well-calibrated alternatives. Conventional ML models show moderate calibration quality, with SVM and KNN leading their group. These findings emphasize the value of log loss as a complementary metric to accuracy and F1-score, particularly in security-sensitive environments where prediction confidence plays a critical role in decision-making and risk mitigation.

Specificity Analysis

Specificity measures a model's ability to correctly identify benign inputs by quantifying the proportion of true negatives among all non-attack instances. This metric is especially important in applications where the cost of false positives is high, and the system must avoid misclassifying legitimate activity as malicious. Within the context of SQL injection detection, high specificity ensures that legitimate user queries are not erroneously flagged, thereby preserving system usability and user trust. Fig. 10 presents the specificity scores of conventional ML, boosting-based ML, and DL models, enabling comparative evaluation of their capacity to minimize false alarms.

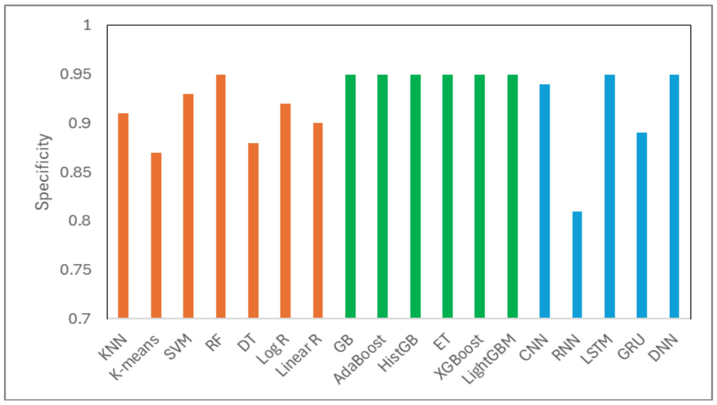

Fig 10: IDS Specificity Analysis

Within the conventional ML category, models demonstrate moderate to strong specificity. SVM leads with a score of 0.93, followed by LogR (0.92) and KNN (0.91), indicating reliable performance in preserving benign traffic. LinearR and DT yield slightly lower values (0.90 and 0.88), while KMeans records the lowest specificity (0.87), reflecting a greater tendency to misclassify non-malicious inputs. By comparison, boosting-based ML models consistently achieve top-tier specificity. RF, GB, AdaBoost, HistGB, ET, XGB, and LGBM each attain a score of 0.95, demonstrating exceptional precision in distinguishing benign from malicious traffic. These outcomes highlight the effectiveness of ensemble learning techniques in reducing false alarms and maintaining operational stability under real-world conditions. In terms of DL performance, models also exhibit competitive specificity. CNN achieves a score of 0.94, closely aligning with the best-performing boosting models. LSTM and DNN match the highest specificity (0.95), indicating robust generalization and low false positive rates. GRU records a slightly lower value (0.89), while RNN yields the lowest among DL models (0.81), suggesting increased sensitivity at the expense of benign traffic preservation. Taken as a whole, boosting models such as RF, GB, HistGB, ET, XGB, and LGBM demonstrate superior specificity, confirming their reliability in minimizing false positives. DL models like LSTM, DNN, and CNN also perform strongly, offering well-calibrated alternatives. Conventional ML models show greater variability, with SVM and KNN emerging as the most effective within their group. These findings emphasize the importance of specificity in maintaining user trust and system efficiency, particularly in high-volume environments where benign traffic predominates.

Cohen's Kappa Analysis

Cohen's Kappa quantifies the agreement between predicted and actual classifications, adjusting for the possibility of chance agreement. Unlike raw accuracy, which may be inflated in imbalanced datasets, Kappa offers a more nuanced measure of model reliability by accounting for the expected agreement that could occur randomly. A higher Kappa value indicates stronger consistency between the model's predictions and the true labels, thereby reflecting greater classification stability. In the context of SQL injection detection, this metric is particularly valuable for evaluating how reliably a model distinguishes between malicious and benign queries under varying class distributions.

Fig. 11 presents the Cohen's Kappa scores for conventional ML, boosting-based ML, and DL models, enabling a comparative assessment of their agreement strength and predictive consistency.

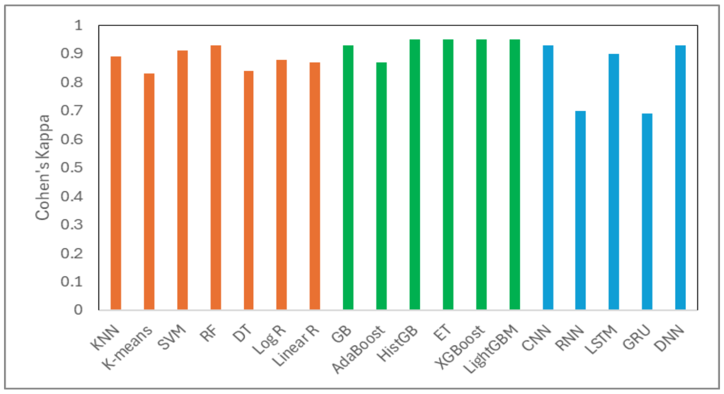

Fig 11: IDS Cohen's Kappa Analysis

Among conventional ML models, agreement levels range from moderate to strong. SVM achieves the highest Kappa score within this group (0.91), followed by KNN (0.89), LogR (0.88), and LinearR (0.87), indicating consistent classification performance beyond chance. DT and KMeans yield slightly lower scores (0.84 and 0.83), suggesting reduced reliability in distinguishing between attack and benign inputs. Boosting-based ML models, by comparison, demonstrate consistently high Kappa values. HistGB, ET, XGB, and LGBM each attain the top score (0.95), reflecting near-perfect agreement and exceptional classification stability. RF and GB follow closely (0.93), while AdaBoost records a slightly lower score (0.87), indicating solid but less consistent performance relative to other ensemble methods. These outcomes reinforce the strength of boosting techniques in producing highly dependable detection results. Likewise, DL models perform competitively. CNN, DNN, and LSTM achieve Kappa scores of 0.93, 0.93, and 0.90, respectively, aligning closely with the leading boosting models and demonstrating strong agreement in classification decisions. Conversely, RNN and GRU record lower scores (0.70 and 0.69), pointing to greater variability and diminished reliability, potentially due to instability during training or sensitivity to sequential input noise. Viewed collectively, boosting models such as HistGB, ET, XGB, and LGBM exhibit superior Cohen's Kappa performance, confirming their robustness in SQL injection detection tasks. DL models like CNN, DNN, and LSTM also show strong agreement, offering reliable alternatives. Conventional ML models display more variability, with SVM and KNN emerging as the most consistent within their category. These findings underscore the value of Cohen's Kappa as a diagnostic metric for evaluating model consistency, particularly in security-critical applications where sustained classification precision is essential across diverse input conditions.

CONCLUSIONS

This work introduces an AI-based IDS framework designed to identify SQL injection attacks in smart home IoT networks. The framework integrates a diverse set of learning models for SQL query classification, spanning three categories: conventional ML, boosting-based ML, and DL models. Comprehensive evaluation using a wide range of performance metrics reveals that ensemble models consistently outperform others. These models proved highly reliable in both detecting malicious traffic and minimizing false alarms, making them ideal candidates for deployment in smart home environments where trust and responsiveness are paramount. Deep learning models, particularly CNN, LSTM, and DNN, also exhibited strong and balanced performance. Their ability to learn complex patterns translated into high recall and F1-scores, with DNN and CNN matching ensemble models in precision and specificity. While RNN and GRU contributed valuable strengths in generalization and temporal modeling, their lower precision and variability in agreement metrics suggest a need for further refinement. Conventional ML models displayed more heterogeneous results. SVM and RF offered solid detection capabilities, especially in attack identification, but were less consistent in benign traffic classification. KNN stood out for its high specificity and low false positive rate, though its overall sensitivity was comparatively limited. Regression-based models and K-means, while useful for baseline comparisons, showed reduced effectiveness across most metrics. The findings further highlight the importance of selecting models not solely based on measurement but through diverse considerations that determine detection sensitivity, classification confidence, and operational reliability.

References

1. Magara T, Zhou Y. Internet of Things (IoT) of Smart Homes: Privacy and Security. J. Electr. Comput. Eng. 2024 Apr; 2024:1-17. https://doi.org/10.1155/2024/7716956.
2. Merino-Campos C. The Impact of Artificial Intelligence on Personalized Learning in Higher Education: A Systematic Review. Trends High. Educ. 2025 Mar; 4(2): 1-15. https://doi.org/10.3390/higheredu4020017.
3. Canavese D, Mannella L, Regano L, Basile C. Security at the Edge for Resource-Limited IoT Devices. Sens. 2024 Jan; 24(2):1-16. https://doi.org/10.3390/s24020590.
4. Noman HA, Abu-sharkh OMF. Code Injection Attacks in Wireless-Based Internet of Things (IoT): A Comprehensive Review and Practical Implementations. Sens. 2023 Jun; 23(13):1-53. https://doi.org/10.3390/s23136067.
5. Caballero-Gil C, Álvarez R, Hernández-Goya C, Molina-Gil J. Research on smart-locks cybersecurity and vulnerabilities. Wirel. Networks, 2023 Agu; 30:1-13. https://doi.org/10.1007/s11276-023-03376-8.
6. Diana L, Dini P, Paolini D. Overview on Intrusion Detection Systems for Computers Networking Security. Comput. 2025 Mar; 14(3):1-44. https://doi.org/10.3390/computers14030087.
7. Ghaffari A, Jelodari N, Pouralish S, Derakhshanfard N, Arasteh B. Securing internet of things using machine and deep learning methods: a survey. Cluster Comput. 2024 Oct; 27:9065-9089. https://doi.org/10.1007/s10586-024-04509-0.
8. Singh R, Kumar H, Singla RK. An intrusion detection system using network traffic profiling and online sequential extreme learning machine. Expert Syst. Appl. 2015 Dec; 42(22):8609-8624. https://doi.org/10.1016/j.eswa.2015.07.015.
9. Cai Z, Wu Q, Huang D, Ding L, Yu B, Huang J, Fu S. Cognitive state recognition using wavelet singular entropy and ARMA entropy with AFPA optimized GP classification. Neurocomput. 2016 Jul; 197:29-44. https://doi.org/10.1016/j.neucom.2016.01.054.
10. Kabir E, Hu J, Wang H, Zhuo G. A novel statistical technique for intrusion detection systems. Futur. Gener. Comput. Syst. 2017 Jan; 79. https://doi.org/10.1016/j.future.2017.01.029.

11. Wu Z, Wang J, Hu L, Zhang Z, and Wu H. A network intrusion detection method based on semantic Re-encoding and deep learning. J. Netw. Comput. App. 2020 Aug; 164:102688. https://doi.org/10.1016/j.jnca.2020.102688.

12. Thakur S, Chakraborty A, De R, Kumar N, Sarkar R. Intrusion detection in cyber-physical systems using a generic and domain specific deep autoencoder model. Comput. Electr. Eng. 2021 May; 91:107044. https://doi.org/10.1016/j.compeleceng.2021.107044.

13. Khanday S, Fatima H, Rakesh N. Implementation of intrusion detection model for DDoS attacks in Lightweight IoT Networks. Expert Syst. Appl. 2022 Nov; 215:119330. https://doi.org/10.1016/j.eswa.2022.119330.

14. Xu H, Sun Z, Cao C, Bilal H. A data-driven approach for intrusion and anomaly detection using automated machine learning for the Internet of Things. Soft Comput. 2023 Jul. https://doi.org/10.1007/s00500-023-09037-4.

15. Viegas EK, Santin AO, Tedeschi P. Toward a Reliable Evaluation of Machine Learning Schemes for Network-Based Intrusion Detection. IEEE Inter. Thin. Mag. 2023 Jun; 6(2):70–75. https://doi.org/10.1109/IOTM.001.2300106.

16. Xu B, Sun L, Mao X Ding R, Liu C. IoT Intrusion Detection System Based on Machine Learning. Electro. 2023 Oct; 12(20):1-21. https://doi.org/10.3390/electronics12204289.

17. Vishwakarma M, Kesswani N. A new two-phase intrusion detection system with Naïve Bayes machine learning for data classification and elliptic envelop method for anomaly detection. Decis. Anal. J. 2023 Jun; 7:100233. https://doi.org/10.1016/j.dajour.2023.100233.

18. Rangelov D, Lämmel P, Brunzel L, Borgert S, Darius P, Tcholtchev N, Boerger M. Towards an Integrated Methodology and Toolchain for Machine Learning-Based Intrusion Detection in Urban IoT Networks and Platforms. Fut. Internet. 2023 Fab; 15(3):1-20. https://doi.org/10.3390/fi15030098.

19. Maddu M, Rao PN. Network intrusion detection and mitigation in SDN using deep learning models. Int. J. Inf. Secur. 2023 Oct; 24:1-14. https://doi.org/10.1007/s10207-023-00771-2.

20. Barragán-Escandón A, Jara-Nieves D, Romero-Fajardo I, Zalamea-Leon EF, Serrano-Guerrero X. Barriers to renewable energy expansion: Ecuador as a case study. Energy Strateg. Rev. 2022 Sep; 43:100903. https://doi.org/10.1016/j.esr.2022.100903.

21. Wadate AJ, Deshpande SP. Edge-Based Intrusion Detection using Machine Learning Over the IoT Network. 11th International Conference on Emerging Trends in Engineering & Technology - Signal and Information Processing (ICETET - SIP). 2023 Jun; 1-6. https://doi.org/10.1109/ICETET-SIP58143.2023.10151535.

22. Talukder MA, Hasan KF, Islam MM, Uddin MA, Akhter A, Yousuf MA, Alharbi F, Moni MA. A dependable hybrid machine learning model for network intrusion detection. J. Info. Sec. Appl. 2023 Fab;72. https://doi.org/10.1016/j.jisa.2022.103405.

23. Zakariah M, AlQahtani SA, Al-Rakhami MS. Machine Learning-Based Adaptive Synthetic Sampling Technique for Intrusion Detection. Appl. Sci. 2023 May; 13(11)1-31. https://doi.org/10.3390/app13116504.

24. Hnamte V, Najar Nguyen AH, Hussain J, Naik MS. DDoS attack detection and mitigation using deep neural network in SDN environment. Comput. Secur. 2023 Dec; 138:10366. https://doi.org/10.1016/j.cose.2023.103661.

25. Ngo VD, Vuong TC, Luong TV, Tran H. Machine learning-based intrusion detection: feature selection versus feature extraction. Cluster Comput. 2024 Jun; 27(3):2365–2379. https://doi.org/10.1007/s10586-023-04089-5.

26. Sajid576. SQL Injection Dataset. 2021. Available online: https://www.kaggle.com/datasets/sajid576/sql-injection-dataset (Accessed on 02 September 2025).