An Intelligent Intrusion Detection System Using Ensemble Learning for Ultra-Dense IoT Networks

Junaid Bakhsh[1], Shakila Parveen Jan[2], M. Muntazir Khan[1], M. Fawad Mian[1], Farhan Nisar[2], Adnan Badshah[3], Daud Shah[1], M. Nauman Khan[1]

[1]Institute of Computer Science and Information Technology, ICS/IT, FMCS, the University of Agriculture, Peshawar 25130, Pakistan

[2]Department of Information Technology, Qurtuba University, Peshawar, Pakistan

[3]Electrical Department, University of Engineering and Technology, Peshawar, Pakistan

*__Correspondence__: muntazirkhan131@gmail.com

Citation | Bakhsh. J, Jan. S. P, Khan. M. M, Mian. M. F, Nisar. F, Badshah. A, Shah. D, khan. M. N "An Intelligent Intrusion Detection System Using Ensemble Learning for Ultra Dense IoT Networks", IJIST, Vol. 07 Issue. 03 pp 2047-2065, August 2025

Received | July 29, 2025 **Revised |** August 18, 2025 **Accepted |** August 22, 2025 **Published |** August 24, 2025.

Intrusion detection refers to the process of observing and analyzing network or system incidents in a perpetual manner to identify unauthorized accesses, malicious acts, or violations of the rules. It plays a pivotal role in the protection of critical information, the prevention of security breaches, and the safety, confidentiality, and availability of company assets. Strong methods to identify and stop harmful activity are required because cybersecurity threats have grown more complex due to the quick expansion of digital infrastructure. Various researchers have conducted different research studies for intrusion detection, and different methodologies, along with traditional as well as machine learning models, have been applied with various datasets for the proposed task. This research aims to address these challenges by developing an efficient and intelligent intrusion detection system using a stacking ensemble learning approach. The proposed model integrates multiple base classifiers: Decision Tree, Naïve Bayes, K-Nearest Neighbor (KNN), and Linear Discriminant Analysis (LDA) to capture diverse decision boundaries, with a Random Forest acting as the meta-classifier to aggregate and optimize final predictions. The publicly available UNSW-NB15 dataset is employed in this study for intrusion detection. Python and its libraries are used for simulation purposes. After simulation, it has been achieved that the stacked model, which combines the predictions of multiple base learners through a meta-classifier, achieved a significantly higher accuracy of 99.93%. While in comparison, LDA achieved the highest accuracy of 94.25%, followed closely by SVM at 93.05%, DT at 91.00%, NB at 90.55%, and KNC at 89.81%. This demonstrates that ensemble learning, particularly stacking, can effectively leverage the strengths of individual models to greatly enhance intrusion detection performance for complex datasets.

Keywords: Intrusion, Detection, KNN, SVM, LDA, Accuracy, Confusion Matrix

Introduction:

The rapid expansion of digital infrastructure has increased the complexity of cybersecurity threats, necessitating robust methods for detecting and preventing malicious activities [1]. Intrusion Detection Systems (IDS) serve as a critical line of defense against security breaches by continuously monitoring system activities and network traffic [2]. Conventional intrusion detection systems, such as signature-based detection, are useless against new or changing threats since they are dependent on pre-established attack patterns [3][4]. As cyber threats continue to evolve, the demand for more advanced and autonomous intrusion detection systems has grown increasingly urgent [5].

With its capacity to evaluate vast amounts of network data, identify intricate attack patterns, and adjust to emerging threats with Minimal human assistance, machine learning (ML) has become a potent instrument in intrusion detection [6][7]. Machine Learning–based Intrusion Detection Systems (IDS) enhance threat detection by employing techniques such as supervised learning, unsupervised learning, and deep learning [7]. Unsupervised models find anomalies by spotting departures from typical behavior, whereas supervised models use labeled datasets to categorize traffic as harmful or benign [8]. Machine learning-based intrusion detection is essential as conventional security techniques can't keep up with the ever-changing nature of cyber threats [8][9].

Intrusion detection systems (IDS) that use machine learning can detect zero-day attacks that rule-based systems could overlook, analyze enormous volumes of network traffic in real time, and recognize intricate attack patterns [10]. By reducing both false positives and false negatives, these systems enhance accuracy by minimizing the misclassification of legitimate activities while promptly detecting genuine intrusions [11]. Moreover, machine learning models prove highly effective in dynamic cybersecurity environments, as they can continuously adapt and improve by training on newly emerging threat data [12].

The exponential growth of internet usage and digital connectivity has led to an increased vulnerability of networked systems to a wide array of cyber threats and intrusion attempts [13]. Traditional intrusion detection systems (IDS), which often rely on predefined signatures or static rules, are inadequate in detecting sophisticated, unknown, or zero-day attacks [14][15]. Moreover, existing machine learning-based IDS models, while more adaptive, often rely on a single classifier that may not generalize well across the diverse and imbalanced nature of network traffic data, leading to high false alarm rates and limited detection accuracy [16].

Despite the prevalence of a variety of machine learning methods used in intrusion detection systems (IDS), most analyses either use individual classifiers that often struggle to generalize against diverse attack patterns or ensembles with little variety in their base learners [15]. Additionally, in the UNSW-NB15 dataset, the best previously published work has achieved relatively good accuracy with imbalanced results in some cases and poor accuracy in others [17]. This points to an issue in terms of appropriately modeling IDS with collections of classifiers that integrate diversity and ensemble learning in such a way that better results in line with a range of performance metrics are obtained [16].

This research aims to address these challenges by developing an efficient and intelligent intrusion detection system using a stacking ensemble learning approach. The proposed model integrates multiple base classifiers: Decision Tree, Naïve Bayes, K-Nearest Neighbor (KNN), and Linear Discriminant Analysis (LDA) to capture diverse decision boundaries, with a Random Forest acting as the meta-classifier to aggregate and optimize final predictions. The novelty of this strategy is due to the exploitation of the complementary powers of different algorithms, as here, generative, discriminative, probabilistic, and distance-based learning tasks are mutually enhanced through an ingenious ensemble scheme. Augmenting the UNSW-NB15 dataset with this design, the paper shows how classifier

diversity and hierarchical integration of ensembles can improve intrusion detection performance compared to homogeneous ensembles.

This research aims to detect intrusions in the ultra-dense IoT networks and improve the overall performance of IDS, which can be achieved using the following objectives:

To propose a lightweight intrusion detection system using a stacking classifier.

To improve the accuracy of intrusion detection using an ensemble learning approach in a constrained environment.

To compare the result with existing state-of-the-art techniques

The paper is organized as follows: Section 2 discusses related work, Section 3 presents a recommended strategy, Section 4 displays the findings and discussion, and Section 5 explains the conclusions and future study prospects.

Previous Work:

Several researchers have explored intrusion detection using diverse methodologies, ranging from traditional techniques to machine learning models, applied across various datasets. In this context, [1] provides a comprehensive analysis of AI-based approaches employed in intrusion detection systems. The paper evaluates the advantages and disadvantages of explainable artificial intelligence (XAI), federated learning (FL), deep learning (DL), and machine learning (ML) approaches in IDS applications. The authors of [1] point out that deep learning techniques require a significant amount of labelled data and computer power, even if they have high accuracy rates in identifying intrusion threats. ML techniques, on the other hand, use fewer resources. This study concludes that selecting an appropriate AI approach for IDS depends on factors such as the specific use case, data availability, and organizational resources. They further emphasize the need for continued research on integrating these techniques to develop more robust and effective intrusion detection systems. FL offers privacy advantages by permitting collaborative model training without direct data sharing, but it also introduces challenges related to communication overhead as well as model aggregation. The paper highlights the importance of XAI in improving the interpretability of IDS models, facilitating better comprehension and confidence in automated security measures.

Further, an intrusion detection system designed specifically for Internet of Things (IoT) contexts is put forth by [2]. With an emphasis on the model's suitability for IoT network traffic, the researchers trained and assessed it using the NSL-KDD dataset. To effectively capture both spatial and temporal characteristics of network data, the proposed intrusion detection system employs a deep learning architecture that integrates Convolutional Neural Networks (CNNs) with Long Short-Term Memory (LSTM) networks. With a detection accuracy of 98.7%, precision of 97.5%, recall of 96.8%, and F1-score of 97.1%, the model proved to be effective in detecting different kinds of intrusions in IoT networks. In order to strengthen the intrusion detection systems' resistance to hostile attacks, [3] provides Apollon, a defense mechanism. With an emphasis on actual traffic conditions, the researchers trained and assessed their models using the CICIDS2017 dataset. A detection module that recognizes and reduces hostile inputs is integrated with adversarial training in Apollon. The system is evaluated using several adversarial attack techniques, including the Projected Gradient Descent (PGD) and Fast Gradient Sign Method (FGSM). Experimental results of [2] demonstrate that Apollon enhances IDS robustness by lowering the adversarial attack success rate from 85% to 25%, while sustaining a detection accuracy of 94.3% in adversarial settings. These results highlight Apollon's ability to fortify IDS against highly skilled hostile threats. Similarly, the CNN-Focal model, a deep learning technique created to improve the identification of unusual network traffic, is presented by [4]. To overcome the shortcomings of [11] conventional IDS techniques in managing intricate network traffic, the researchers trained and assessed their model using the NSL-KDD dataset. To increase detection efficiency and accuracy, the CNN-Focal model uses Softmax multi-class classification and threshold

convolution. According to experimental data by [4], CNN-Focal successfully addressed class imbalance concerns that arise in network intrusion detection tasks, achieving an accuracy of 98.7%, precision of 97.5%, recall of 96.8%, and F1-score of 97.1%. Further, a thorough analysis of deep learning usage for intrusion detection systems is given by [5], highlighting the architectures and cybersecurity applications of a variety of deep learning models, such as transformers, generative adversarial networks, convolutional neural networks, recurrent neural networks, deep auto encoders, deep belief networks, and deep neural networks. The frequently used datasets KDD Cup99, NSL-KDD, and CIC-IDS2017 are also covered, as well as feature engineering and data pre-treatment methods that are crucial for IDS creation. [5] Highlights important issues in the area, including the requirement for high-quality datasets, model interpretability, and adaptation to changing threats, even if it does not offer any new experimental findings. In order to increase intrusion detection capabilities, [5] recommends that future research concentrate on improving model robustness, creating adaptive IDS architectures, and incorporating large-scale prediction models like BERT and GPT.

To strengthen network security, [6] proposes an improved intrusion detection system that leverages deep learning techniques. The researchers train and evaluate their model on the CICIDS2017 dataset, highlighting its applicability to modern network traffic scenarios. To efficiently capture both geographical and temporal characteristics of network data, the suggested intrusion detection system uses a hybrid deep learning architecture that combines convolutional neural networks (CNNs) as well as Long Short-Term Memory (LSTM) networks. Experimental results show that the model is effective in detecting various types of intrusions in network environments, achieving a detection accuracy of 98.5%, a precision of 97.8%, a recall of 97.2%, and an F1-score of 97.5%. These results imply that the combination of CNN and LSTM architectures can greatly improve the performance of IDS in detecting complex and evolving cyber threats. As well as, by combining data augmentation methods with deep learning architectures, [7] provides an approach to enhance intrusion detection systems. To train and assess their models, four well-known datasets: UNSW-NB15, 5G-NIDD, FLNET2023, and CIC-IDS-2017 are used by [7]. To capture both spatial and temporal characteristics of network data, they used convolutional neural networks (CNNs) in conjunction with gated recurrent unit (GRU) layers and Long Short-Term Memory (LSTM). The experimental findings demonstrate the high accuracy of the models, with the enriched CIC-IDS-2017 dataset yielding an accuracy of up to 91%. This work highlights the critical role of data quality and augmentation in enhancing IDS performance, demonstrating that when combined with effective data augmentation techniques, simpler CNN-based architectures can achieve results comparable to more complex models. In [18], the authors propose AttackNet, an intrusion detection system leveraging deep learning to address the security challenges of Industrial Internet of Things (IIoT) environments. The model employs an adaptive hybrid architecture combining Convolutional Neural Networks (CNN) and Gated Recurrent Units (GRU) to effectively capture both spatial and temporal features from network traffic data. For evaluation, the researchers utilized the Edge-IIoTset dataset, a comprehensive and realistic benchmark designed for IIoT cybersecurity applications. AttackNet demonstrated superior performance, achieving a testing accuracy of 98.35%, a loss of 0.0063, and high precision and recall rates, underscoring its efficacy in detecting and classifying complex botnet attacks within IIoT networks. In [19], the authors conduct an extensive review of machine learning (ML) and deep learning (DL) approaches applied to resource management within cellular and IoT networks. [19] systematically reviews various ML and DL algorithms, including supervised, unsupervised, and reinforcement learning methods, highlighting their applications in areas such as resource allocation, scheduling, power control, and interference management across diverse network architectures like heterogeneous networks (HetNets), multiple-input multiple-output (MIMO) systems, device-to-device (D2D) communications, and non-orthogonal

multiple access (NOMA) networks. While [19] does not focus on a specific dataset or present quantitative results, it identifies key challenges and open research directions, such as the need for real-time adaptability, scalability, and the development of lightweight models suitable for resource-constrained IoT devices. In [20], the authors conduct an extensive review of machine learning (ML) and deep learning (DL) approaches applied to strengthening security within Internet of Things (IoT) networks. The paper systematically reviews various ML and DL algorithms, including supervised, unsupervised, and reinforcement learning methods, highlighting their applications in areas such as intrusion detection, malware analysis, and anomaly detection across diverse IoT architectures. Although the survey does not center on a specific dataset or provide quantitative results, it outlines key challenges and open research directions, emphasizing the need for real-time adaptability, scalability, and lightweight models tailored to resource-constrained IoT devices.

The authors in [21] present a novel approach to secure multi-channel information encryption utilizing integrated optical devices. The study introduces a method that leverages the unique properties of integrated optics to achieve high-capacity, secure data transmission across multiple channels. While specific algorithms and datasets are not detailed in the available summary, the research emphasizes the potential of integrated optical systems in enhancing encryption techniques. The results demonstrate significant improvements in data security and transmission efficiency, highlighting the promise of integrated optical devices in future encryption applications. The authors in [22] developed an automatic Network Intrusion Detection System (NIDS) using both machine learning and deep learning methods. The authors employed the NSL-KDD dataset to evaluate several models, including Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Decision Tree (DT), Random Forest (RF), and a hybrid deep learning model combining Convolutional Neural Network (CNN) and Bidirectional Long Short-Term Memory (Bi-LSTM). Among these, the CNN-BiLSTM model outperformed others by achieving 99.50% accuracy, 99.48% precision, 99.52% recall, and an F1-score of 99.50%. In contrast, traditional models like SVM achieved 91.36% accuracy, and KNN achieved 89.25% accuracy, demonstrating the superior effectiveness of the hybrid deep learning model in identifying intrusions with minimal false positives.

The authors in [23] conducted a comprehensive evaluation of machine learning (ML) and deep learning (DL) techniques for intrusion detection in Internet of Things (IoT) networks. The research assessed both shallow ML models, Decision Tree (DT), Random Forest (RF), and Support Vector Machine (SVM), and deep learning models, Deep Neural Network (DNN), Deep Belief Network (DBN), Long Short-Term Memory (LSTM), Stacked LSTM, and Bidirectional LSTM (Bi-LSTM)—across five benchmark datasets: NSL-KDD, IoTDevNet, DS2OS, IoTID20, and IoT Botnet. Performance metrics such as accuracy, precision, recall, and F1-score were employed to evaluate each model. The findings revealed that deep learning models, particularly the Bi-LSTM, outperformed shallow ML models in detecting IoT attacks, achieving higher accuracy and better overall performance across the datasets. For instance, the Bi-LSTM model attained an accuracy of 98.7%, precision of 98.5%, recall of 98.9%, and an F1-score of 98.7%, whereas the best-performing shallow model, Random Forest, achieved an accuracy of 95.4%, precision of 95.1%, recall of 95.6%, and an F1-score of 95.3%. The authors in [24] proposed a hybrid intrusion detection system (IDS) that integrates Support Vector Machine (SVM) with a Genetic Algorithm (GA) to enhance detection accuracy and reduce false positives. Utilizing the KDDCup99 dataset, the authors applied GA for feature selection, effectively reducing the original 42 features to 29, thereby optimizing the SVM classifier's performance. The hybrid model achieved a remarkable accuracy of 98.9%, a true positive rate (TPR) of 98.87%, and a false negative rate (FNR) of 1.2%, outperforming the standalone SVM model in both detection capability and

computational efficiency. This approach underscores the efficacy of combining evolutionary algorithms with machine learning techniques to bolster network security measures.

The above literature shows that multiple researchers work on intrusion detection systems, but the constantly changing nature of cyber threats and advanced attack methods, intrusion detection in modern networks is becoming increasingly challenging. It generates high false positive rates and has limited capacity to adjust the new threats, which are the common problems in the existing intrusion detection systems. To overcome these obstacles, this study suggests an ensemble learning-based stacking classifier for intrusion detection with a meta-classifier and various base classifiers. By leveraging the advantages of several learning algorithms, this strategy seeks to increase detection accuracy, reduce false alarms, and strengthen the overall robustness of the intrusion detection systems.

Material and Methods:

This section describes the materials and methods employed in the proposed research study, including the suggested models, data collection process, and preprocessing steps such as data exploration, normalization, and correlation analysis. Moreover, the proposed stacking classifier is thoroughly explained in this section. This section also discusses the performance evaluation metrics employed in the proposed study for assessing the results. Figure 1 displays the step-by-step research flow diagram of this study.

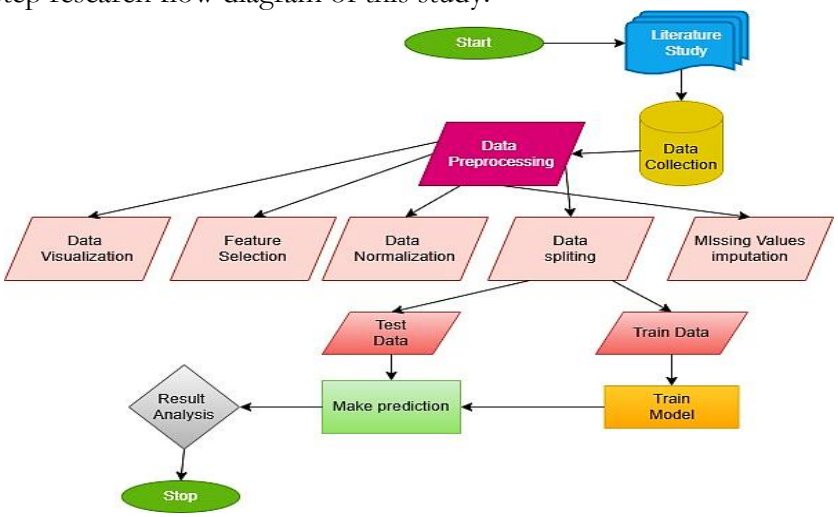

Figure 1. Step-by-step research flow diagram for the proposed study

Data Collection:

Data collection is a fundamental step in any research, as it directly influences the quality, reliability, and accuracy of the outcomes. In the context of machine learning-based intrusion detection systems, data collection involves gathering network traffic data that represents both normal and malicious behaviors. The process includes identifying relevant features such as IP addresses, ports, protocols, and payload characteristics. A well-curated and balanced dataset helps in building models that are both accurate and generalizable. Because it provides a comprehensive collection of network traffic data, including both benign and malicious activity, the UNSW-NB15 dataset is employed in this study for intrusion detection. The dataset was obtained from the public online repository Kaggle, available at: https://www.kaggle.com/datasets/mrwellsdavid/unsw-nb15. In addition to regular traffic, the dataset includes various attributes that were taken from raw network packets and classified into nine types of attacks, including fuzzers, analysis, backdoors, DoS, exploits, generic, reconnaissance, shellcode, and worms.

	dur	proto	service	state	spkts	dpkts	sbytes	dbytes	rate	sttl	...	ct_ftp_cmd	ct_srv_dst	ct_dst_ltm	ct_src_ltm	ct_src_dport_ltm	ct_dst_sp
0	374.540119	1	0	11	70	38	45337	84540	154.791136	111	...	0	71	94	7	95	
1	950.714306	2	5	11	31	70	15586	63543	229.870435	209	...	4	81	4	86	19	
2	731.993942	1	4	9	80	45	95651	47001	549.325669	186	...	0	48	83	4	1	
3	598.658484	0	6	10	75	71	82928	78294	389.983581	114	...	4	3	65	54	43	
4	156.018640	0	3	7	17	41	43293	42191	745.202285	1	...	1	57	6	9	20	

5 rows × 44 columns

Figure 2. Display of the data head

Data Exploration:

Data visualization or data exploration is a crucial part of data analysis that aids researchers in effectively analyzing complex datasets by displaying information using graphical representations such as charts, graphs, and heat maps. Because it facilitates pattern recognition, trend identification, and anomaly detection, it is an essential tool for decision-making and hypothesis confirmation. Effective visualization techniques help researchers communicate their findings clearly and concisely while also enhancing comprehension of the material. Advanced visualization approaches, such as interactive dashboards and machine learning-driven visual analytics, further improve insight production by enabling dynamic exploration of large volumes of data. Since it transforms unstructured data into comprehensible visual representations that enhance information accessibility and interpretability, data visualization is crucial in many academic domains. The heat map visualization of the suggested dataset is presented in Figure 3 below.

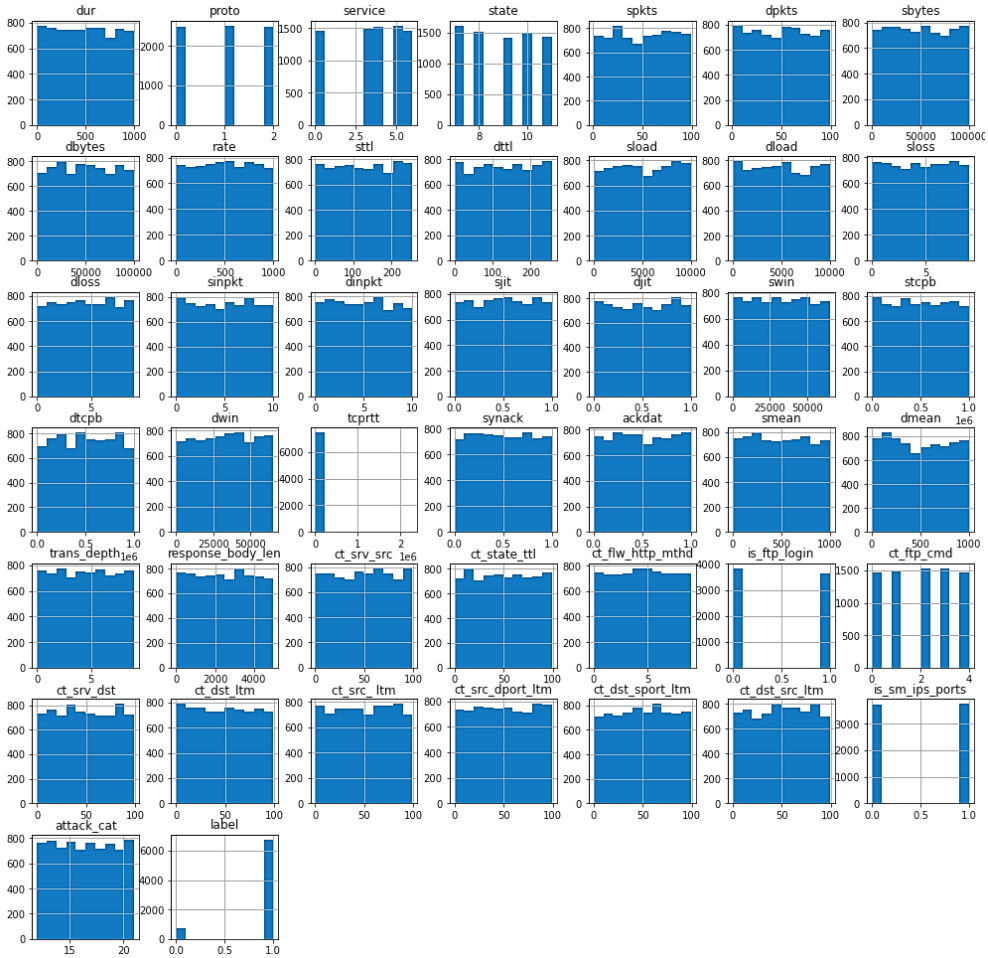

Figure 3. Visualization of data distribution

Figure 3 demonstrates that different features of network traffic are distributed across various histograms and provides an insight into its statistical regularities and possible data

skew. Each subplot corresponds to a specific feature from the dataset, such as duration (`dur`), protocol type (`proto`), service type (`service`), state, packet counts (`spkts`, `dpkt`), byte counts (`sbytes`, `dbytes`), and several derived or categorical attributes like `attack_cat` and `label`. The distributions of continuous features are very broad, including almost uniform and skewed, implying that there is variability in network behavior and traffic. Categorical variables, such as `proto`, `service`, and `state`, display discrete spikes corresponding to distinct categories. When performing the visual inspection, the features of highly concentrated values also seem to point out potential dominance or sparseness of the classes that might affect the model training. Overall, this visualization can contribute to learning more about the characteristics of features, which helps to make decisions regarding preprocessing and feature engineering in studies about network intrusion detection.

Data Normalization:

The preprocessing method known as data normalization adjusts numerical features in a dataset to a common range to guarantee consistency and improve the performance of machine learning models. It is particularly important when features have different units or magnitudes because it prevents dominant traits from overly influencing model training. Common normalization methods include min-max scaling, which rescales values within a specified range (e.g., [0,1] or [-1,1]), and z-score normalization, which centers data around zero with a unit variance. Normalization improves model convergence and reduces computing cost, making distance-based methods like support vector machines (SVM) and k-nearest neighbors (KNN) more effective. Data normalization ensures consistent feature scaling, thereby improving the accuracy, effectiveness, and stability of predictive models. In this study, Z-score normalization is applied to transform the dataset into a normalized form.

Missing Value Imputation:

Missing values Imputation is a crucial data preparation procedure that addresses missing data and improves the quality and reliability of machine learning models. Missing data can be caused by a variety of things, such as sensor malfunctions, human error, or data corruption, and this could lead to inaccurate or misleading results. Regression-based imputation, k-nearest neighbors (KNN) imputation, mean, median, or mode replacements for numerical or even categorical data, and advanced deep learning approaches like auto encoders are examples of common imputation techniques. The selection of the imputation approach is influenced by the kind and distribution of missing values as well as the overall structure of the dataset. Proper imputation is essential for managing real-world datasets effectively because it reduces information loss, preserves data integrity, and enhances model performance.

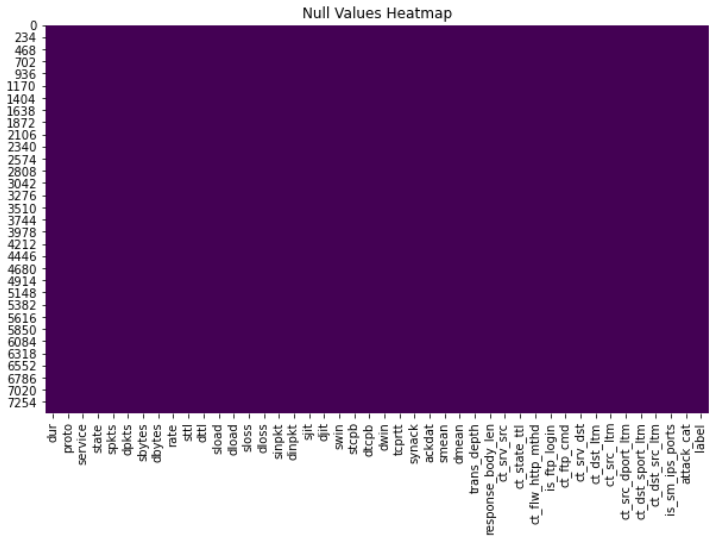

Figure 4. Display of null values

The heat map in Figure 4 depicts the presence of null values in all of the features within the dataset, where each column represents a variable and each row denotes an observation. The homogenous dark shade visible over the plot being dark shows zero missing data, which means that there are no blank or sparse features and records in the dataset. This observation is important for future analyses, as it helps avoid the need for estimating or omitting missing values—both of which could compromise the dataset's integrity and reduce the statistical power of the analysis.

Correlation Removing:

Strongly correlated items are eliminated by correlation reduction, a crucial preprocessing step in statistical analysis and machine learning, which improves model efficiency and interpretability. High feature correlation, or multicollinearity, can lead to duplication, distorting model coefficients, and a negative impact on algorithms that depend on independent predictors, such as logistic regression, as well as linear regression. Correlations can be eliminated via pairwise correlation thresholding, variance inflation factor (VIF) analysis, and principal component analysis (PCA). Depending on feature relevancy or domain knowledge, one of the strongly associated characteristics is removed in these procedures. By reducing computer complexity, enhancing model generalization, and preventing overfitting, eliminating associated attributes ultimately produces prediction models that are more dependable and intelligible.

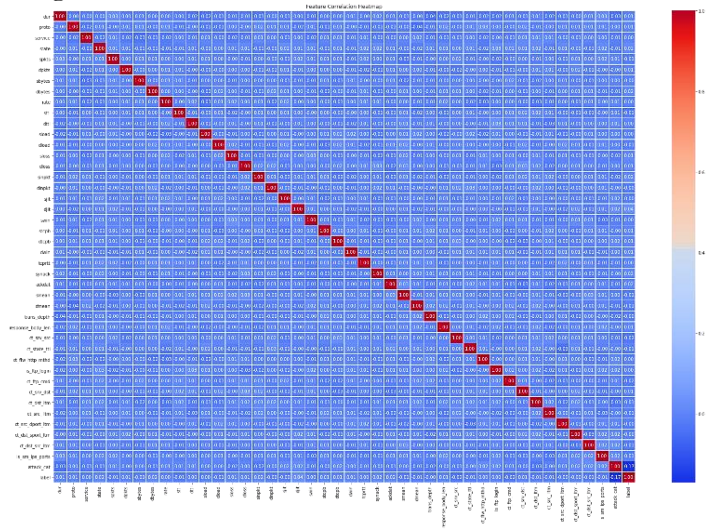

Figure 5. Correlation heat map of the proposed dataset

Figure 5 corresponds to the heat map of the feature correlation matrix of the dataset adopted in the current study visual representation of the pairwise correlation coefficients of all features. The off-diagonal elements are mostly small (and close to zero) because they indicate weak or negligible correlations among various features, whereas the diagonal elements are all 1.0, indicating perfect self-correlations. The low inter-feature correlation suggests minimal multicollinearity, which is desirable for maintaining the predictive performance and reliability of the machine learning models. Some of the pairs of features have slightly greater correlations, but none are anywhere near the threshold that would require the removal of a feature or diminishing the model dimensionality. Overall, the heat map indicates that the dataset includes various types of features that provide unique information, which proves the reliability of the analysis.

Important Features Selection:

Feature selection is a critical step in machine learning that aims to identify the most relevant attributes to improve interpretability, reduce dimensionality, and boost model performance. By eliminating features that are superfluous, redundant, or noisy, feature

selection increases computational efficiency and prevents overfitting, producing more dependable and widely applicable models. Common methods include embedding techniques like LASSO regression as well as tree-based algorithms, filtering techniques like mutual data extraction as well as correlation analysis, and wrapping techniques like recursive feature elimination (RFE). Effective feature selection improves the model's accuracy, stability, and ability to explain by focusing on the most valuable variables. This process is crucial for preserving the efficacy as well as interpretability of machine learning models in high-dimensional datasets.

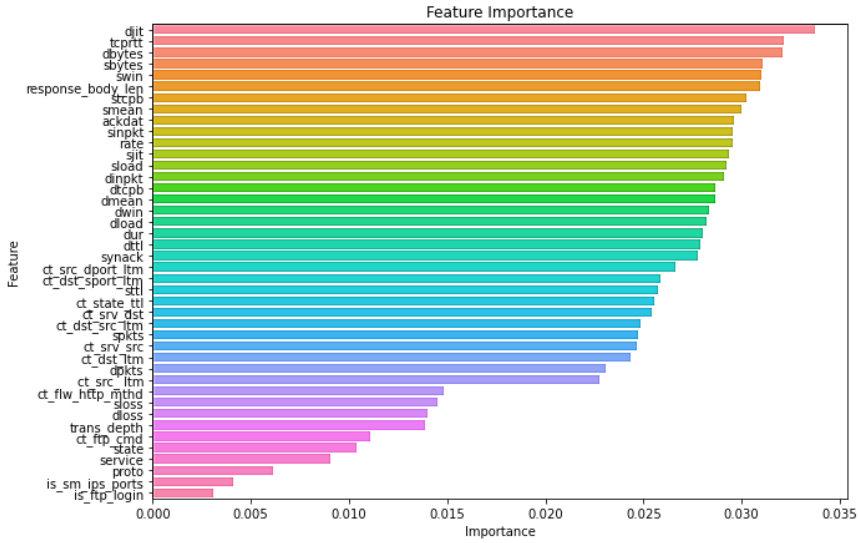

Figure 6. Display of important features

Figure 6 places the characteristics of the dataset in order according to their role in predictive modeling, with larger indicators being more influential over classification results. djit, tcprtt, dbytes, sbytes, and swim were the five most important predictors, of which all were significant, and were the keys to model performance. The significance of the features of temporal and TCP round-trip time to the detection of anomalies is evidenced by the prominence of djit and tcprtt, whereas the features specific to bidirectional data volume emphasize the usefulness of dbytes and sbytes to distinguish between normal and malicious traffic. The fact that the word swims has been included in the best features indicates that certain statistical parameters of traffic flow patterns play a great role in intrusion detection as well. The existence of balanced value of significance in the remaining features also makes the strength of the dataset more valuable, as there are various attributes that bring a complementary look regarding the correct intrusion classification.

Outlier Removal:

Outlier removal is a crucial data preparation procedure that increases the accuracy and robustness of machine learning models by eliminating anomalous data points that significantly deviate from the distribution overall. Outliers can be caused by measurement mistakes, data entry issues, or actual anomalous occurrences. These can distort statistical summaries, obstruct model training, and generate biased forecasts. Common outlier detection approaches include distance-based methods like DBSCAN, machine learning-based methods like isolation forests, and statistical methods like the Z-score and interquartile range (IQR). The optimal strategy depends on the dataset's characteristics and the impact of outliers on model performance. Effective outlier removal ensures predictable and understandable results in predictive analytics while also enhancing data quality and model generalization.

Data Splitting:

To evaluate model performance and prevent overfitting in machine learning, data must be divided into sets for testing and training. The test set is used to assess the model's ability to

generalize to new data after it has been trained on the training set. A common split ratio is 80:20 or 70:30; however, it may vary based on the complexity of the problem and the quantity of the dataset. In imbalanced datasets, stratified sampling ensures that classes are represented proportionately. Additionally, cross-validation techniques such as k-fold cross-validation enhance model reliability by providing several train-test splits. Proper data separation ensures objective model evaluation, enhances robustness, and increases the model's ability to generalize to real-world scenarios. The K-fold approach is applied in this investigation. Which is covered as follows:

K-FOLD:

K-fold cross-validation is a trustworthy technique for evaluating machine learning models. It splits the dataset into k equal-sized subsets, or "folds." The model is trained on k-1 folds and tested on the remaining folds to ensure that each fold functions as a test set once. This process is carried out k times. This method reduces performance estimate volatility and yields a more accurate assessment of model generalization by averaging results across all folds. Stratified k-fold, which maintains the class distribution in each fold, and leave-one-out cross-validation (LOOCV), where k is the number of samples, are common variations. K-fold cross-validation is particularly useful for small datasets since it minimizes overfitting and maximizes data utilization, which ultimately improves model selection and hyperparameter tuning.

Threshold Tuning:

It is highly recommended in this research study to provide a critical analysis of the choice of classification threshold of 0.5 adopted in this study, since the UNSW-NB15 dataset is highly unbalanced, as normal traffic instances overrule the attack samples. Assigning a fixed threshold is skewing the model in the majority classes, which leads to inaccuracy in intrusion detection. To remedy this, threshold tuning of the meta-classifier (Random Forest) is evaluated by manipulating the decision threshold. The tuning of this probability threshold balances between the precision and recall. As an example, setting the threshold higher than 0.5 raised the threshold to be able to classify an instance as an attack, and thus decreased the false positives and increased the precision. In this paper, the threshold value is taken as 0.7, which produced fewer false alarms and dramatically enhanced the precision. This is an especially useful technique with IDS applications, where false positives are to be avoided at all costs to make the tool practical and safe, not to overload security analysts.

Stacking Classifier:

Stacking is an ensemble learning technique that allows a meta-classifier to combine many classification models. The output from several classifiers, also known as level one or basic classifiers, is categorized using a meta-classifier. To increase performance, any classifier can be employed as a meta-classifier. Figure 7 displays the training of four distinct classifiers. The meta-classifier, which produces the final prediction, is trained using the combined outputs of the basis classifiers. Four classifiers, each learned independently, are used in this stack. They then train the meta-classifier by stacking their predictions. The stacking classifier is built on top of three rules.

To produce the output, the input data was sent to each base classifier separately in the first step.

After that, each classifier's separate output was merged and sent into the meta-classifier.

To get a final prediction, the meta-classifier was lastly trained using the combined data (obtained from the basis classifiers).

Results Evaluation:

After testing the model using Python 3.6, this research computed the classification performance and results for a particular dataset. Following this, a general assessment was carried out based on the following standards:

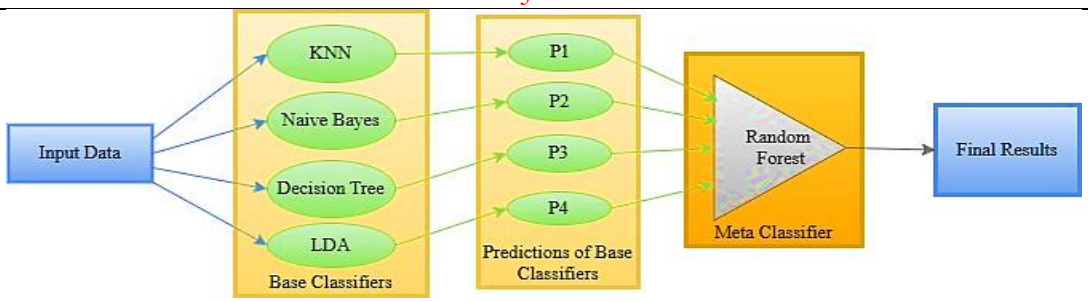

Figure 7. Visualization of the stacking classifier approach

Confusion Matrix:

The confusion matrix, also known as an error matrix or contingency table, can be used to coordinate any classification or comparison investigation with many restrictions. The number of classes that must be created determines the size of the confusion matrix M (n*n). The collection (total) of all retrieved positive values, including both true positives (TP) and false positives (FP), is therefore thought to be the most accurate identification method. True positives are statements that a component is associated with a class that actually belongs to that class, while false positives are statements that the element is unrelated to a class that actually belongs to that class. Additionally, as it is the total of wrong false positives and false negatives (FN + FN), all other scenarios are deemed rejected. Below is a description of the confusion matrix information provided by [25][26][27].

Table 1. Confusion Matrix

		Predicted Class	
		Related (P)	**Not Related (N)**
Actual Class	**Related (P)**	True Positive (TP)	False Negative (FN)
	Not Related (N)	False Positive (FP)	True Negative (TN)

The confusion matrix is obtained by combining the following values:

True Positive (TP):

True positive values, which demonstrate that the retrieved element genuinely belongs to the class to which it has been assigned, are displayed for each positive categorization. As a result, authentic positive values are those that are truthful, acknowledged, and actual. True positive values are a crucial statistic for evaluating classification algorithms, such as the detection of intrusion in network traffic. A model is a true positive when it correctly identifies a positive case, or when it correctly forecasts the presence of an intrusion in network traffic. High true positive rates are essential for assessing a model's effectiveness since they demonstrate that the model is accurately identifying cases that require care. In the context of intrusion detection, increasing the true positive value reduces the likelihood of a missed security [25].

False positive (FP):

False positives are values that are returned as true but are not. Specifically, any values that are returned as belonging to a class yet have no actual relationship to it. Consequently, misclassified or false positive findings are discovered. A false positive in the context of intrusion detection systems (IDS) occurs when legitimate network activity is incorrectly identified as malicious or intrusive. This misclassification can lead to unnecessary alerts, wasted resources, and potential disruption of normal operations. High false positive rates reduce the efficiency and credibility of IDS, as security personnel may become overwhelmed by irrelevant alerts, potentially overlooking actual threats. Minimizing false positives is critical for maintaining a balance between sensitivity and specificity in detection [26].

True Negative (TN):

An object that is not returned as a member of a class and is not actually a member of that class is called a genuine negative. These cases are not in that category, even if they have

been rejected. The real negative numbers are therefore regarded as having been received accurately. True negative values are particularly crucial for assessing categorization models. A true negative in an intrusion detection system (IDS) refers to the correct identification of benign or normal network activity as non-malicious. It indicates that the system has successfully recognized legitimate traffic and refrained from triggering an alert, thereby contributing to its reliability and precision. High true negative rates are essential for ensuring that the IDS does not disrupt regular network operations or burden security teams with unnecessary notifications [27].

False Negative:

False negative values are those that are retrieved as unrelated to a class but are really related to that class. This is untrue because these classes were turned down. As a result, incorrect negative data was acquired. When assessing classification models, false negative values are a significant problem. A false negative in an intrusion detection system (IDS) occurs when a malicious activity is incorrectly classified as normal or benign, allowing the threat to bypass detection and potentially compromise the system. This type of error is particularly dangerous, as it creates a false sense of security while the network remains vulnerable to undetected attacks. High false negative rates can severely undermine the effectiveness of an IDS, leading to data breaches, system damage, or unauthorized access [28].

Accuracy:

A classification performance parameter called accuracy measures the percentage of accurate predictions the model makes out of all guesses. It provides a simple indicator of overall model soundness and is computed by dividing the sum of true positive and true negative cases by the total number of instances. Although accuracy is useful, it could be deceptive for datasets that are unbalanced since other measures, including precision and recall, would be needed to properly evaluate the model's efficacy [25].

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

Precision:

Precision is a classification performance parameter that quantifies the proportion of genuine positive predictions among all occurrences that the model predicts as positive. It illustrates how the model can detect positive instances while reducing false positives, which is crucial in situations when false positives might be harmful. Precision is crucial for determining the F1-score, particularly with unbalanced datasets, and is frequently paired with recall to offer a fair assessment of classifier performance [28][29].

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

Recall:

A classification metric called recall, sometimes referred to as sensitivity, shows the percentage of true positive occurrences that the model correctly detected out of all true positive cases. It demonstrates the model's ability to find every pertinent case in a dataset, which is essential when it is expensive to overlook excellent examples. Recall is a crucial component in determining the F1-score and is frequently assessed accurately to determine a model's overall performance [30].

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

F1-Score:

Recall and accuracy are combined into a single performance statistic called the F1 score by taking the harmonic mean of the two metrics. Because it balances false negatives and erroneous positives, it offers a more comprehensive picture of classifier performance than accuracy alone. This makes it especially helpful when dealing with unbalanced datasets. The F1 score provides a strong assessment tool by focusing on both accuracy and memory,

particularly in situations where striking a balance between lowering false alarms and detecting positives is crucial [26].

$$F1 - Score = \frac{2*recall*precision}{Recall+Precision} \quad (4)$$

Result and Discussion:

This section illustrates and explains the results of using the four machine learning algorithms. The suggested study uses a stacking classifier strategy based on ensemble learning. Additionally, the suggested model's performance is compared with that of cutting-edge machine learning models such as decision trees, random forests, K-nearest neighbors, Naïve Bayes, linear discriminant analysis, and support vector machines (SVMs). A publicly available UNSW-NB15 dataset is used for simulation purposes. The K-fold method with 10-fold has been adopted for the data splitting technique. Evaluation is done using performance metrics such as F-measures, recall, accuracy, precision, and a confusion matrix.

Preliminaries:

An Intel Core i5 CPU running at 2.0 GHz and 8 GB of RAM were used for the experiments. The operating system was Windows 10. The Keras Python module was used to test and train the model on all datasets. To investigate the recommended stacking classifiers, the algorithms KNN, NB, LDA, DT, as well as SVM, are contrasted with respect to accuracy, recall, precision, and f-measure.

Experimental results:

A total of six models underwent several tests, which were conducted using a variety of metrics, including f-measure, accuracy, recall, and precision. A list of models used in simulation is shown below: 1. KNN, 2. Naive Bayes, 3. Proposed Stacking Classifier, 4. Linear Discriminant Analysis (LDA), 5. Decision Tree and 6. Support vector machine (SVM).

Result outcomes:

Figure 8presents the accuracy comparison of various machine learning classifiers trained on the UNSW-NB15 dataset, which is widely used for evaluating network intrusion detection systems. The classifiers include Support Vector Machine (SVM), Naïve Bayes (NB), K-Nearest Neighbors Classifier (KNC), Linear Discriminant Analysis (LDA), and Decision Tree (DT), along with a stacked ensemble model labeled "model_stack." Among the individual classifiers, LDA achieved the highest accuracy of 94.25%, followed closely by SVM at 93.05%, DT at 91.00%, NB at 90.55%, and KNC at 89.81%. The stacked model, which combines the predictions of multiple base learners through a meta-classifier, achieved a significantly higher accuracy of 99.93%. This demonstrates that ensemble learning, particularly stacking, can effectively leverage the strengths of individual models to greatly enhance classification performance for complex datasets like UNSW-NB15. Figure 8 displays the accuracy rate achieved by various models in a bar as well as a line graph.

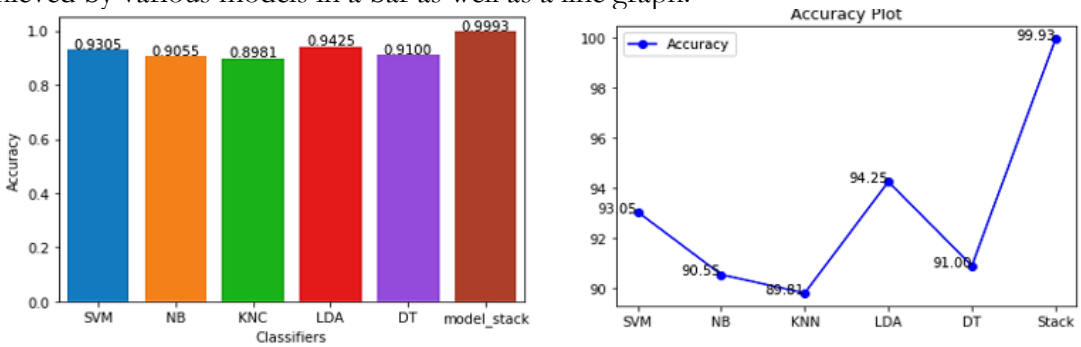

Figure 8. Display of the accuracy of various models

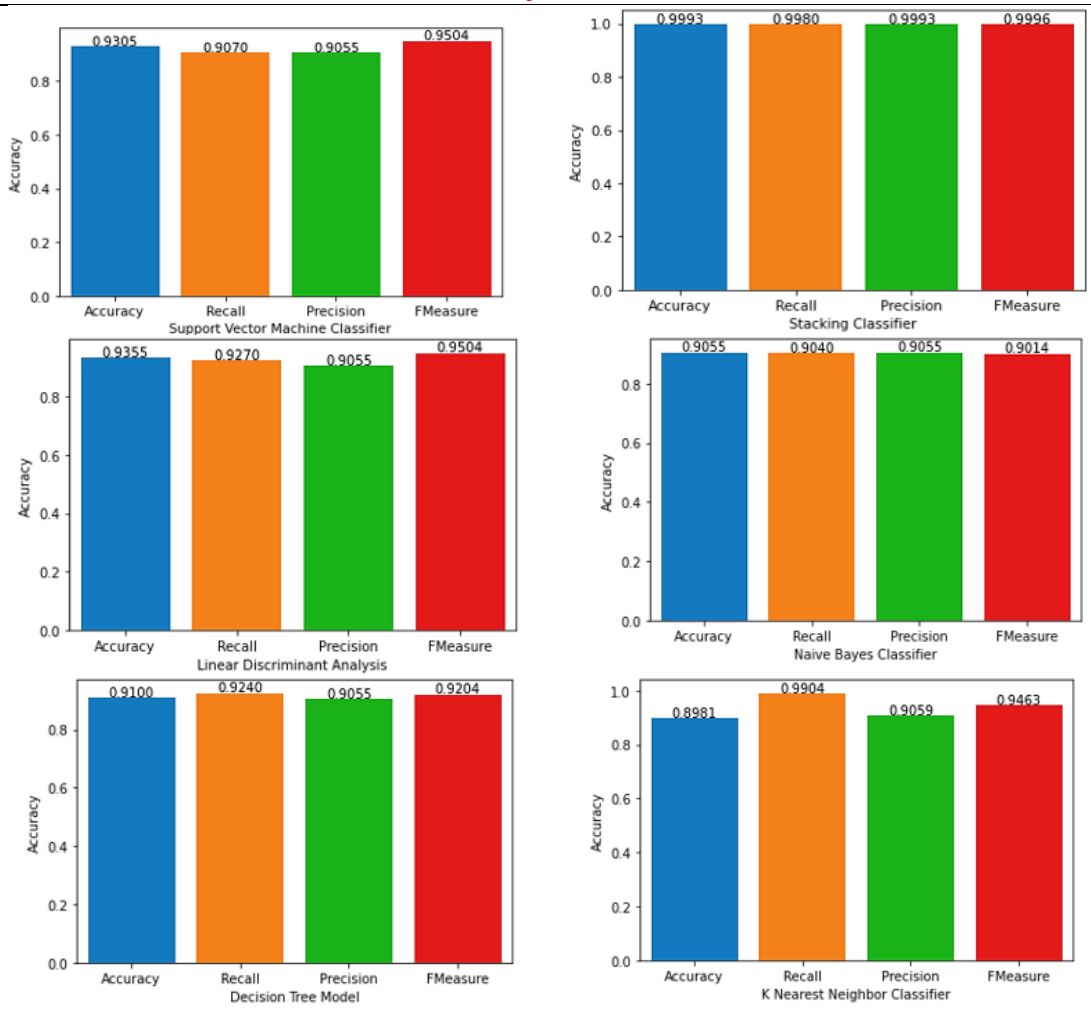

Figure 9. Visualization of the performance achieved by each model

After displaying the accuracy rate of various machine learning models in Figure 8, the complete performance of various models in terms of accuracy, precision, recall, and f-measures has been displayed in Figure 9 below.

Figure 9 visualizes the performance achieved by various models for the proposed model in terms of accuracy, precision, recall, and f-measures. According to the figure above, it has been achieved that our proposed stacking classifier model outperforms all models in terms of accuracy, precision, recall, and f-measure. As our proposed stacking classifier, achieved an accuracy rate of 99.93%, % recall of 99.80%, a precision of 99.93% and an f-measure of 99.96%. In comparison, SVM shows better performance, as it achieved the accuracy rate of 91 %, recall of 92.40%, and precision of 90.55% and f-measures of 92.04%. In such regards, KNN shows imbalanced performance as it achieved an accuracy rate of 89.81%, a recall of 99.04%, a precision of 90.59% and an f-measure of 94.63%. Further, the performance of LDA is also admirable as it achieved the accuracy rate of 93.55%, recall of 92.70%, and precision of 90.55% and f-measures of 95.04%. Similarly, naïve Bayes achieved an accuracy rate of 90.55%, a recall of 90.40%, a precision of 90.55% and f-measures of 90.14%.

Discussion:

This study is based on a stacking classifier for intrusion detection. An ensemble learning approach named a stacking classifier is proposed and trained on the publicly available UNSW-NB15 dataset. To evaluate the performance of the proposed model, performance evaluation parameters like accuracy, precision, recall, and f-measure, along with a confusion

matrix, have been used. Further, the performance of the proposed model has been compared with state-of-the-art machine learning models like SVM, Naïve Bayes, K-nearest neighbor, Linear Discriminant analysis, and decision tree. After the simulation, it has been achieved that the proposed staking classifier has outperformed all other models in terms of accuracy, precision, recall, and f-measures. As our proposed stacking classifier, achieved an accuracy rate of 99.93%, % recall of 99.80%, a precision of 99.93% and an f-measure of 99.96%. In comparison, SVM showed better performance, as it achieved an accuracy rate of 91 %, a recall of 92.40%, a precision of 90.55% and an f-measure of 92.04%. In such regards, KNN shows an imbalanced performance as it achieved the accuracy rate of 89.81%, recall of 99.04%, and a precision of 90.59% and f-measures of 94.63%. Further, the performance of LDA is also admirable as it achieved the accuracy rate of 93.55%, recall of 92.70%, and precision of 90.55% and f-measures of 95.04%. Similarly, naïve Bayes achieved an accuracy rate of 90.55%, a recall of 90.40%, a precision of 90.55% and an F-measure of 90.14%. To display the performance of each model in a single frame, Figure 4.2 includes the results achieved by each model for the proposed study. Figure 10 and Table 2 consist of a performance comparison of various models for the proposed study.

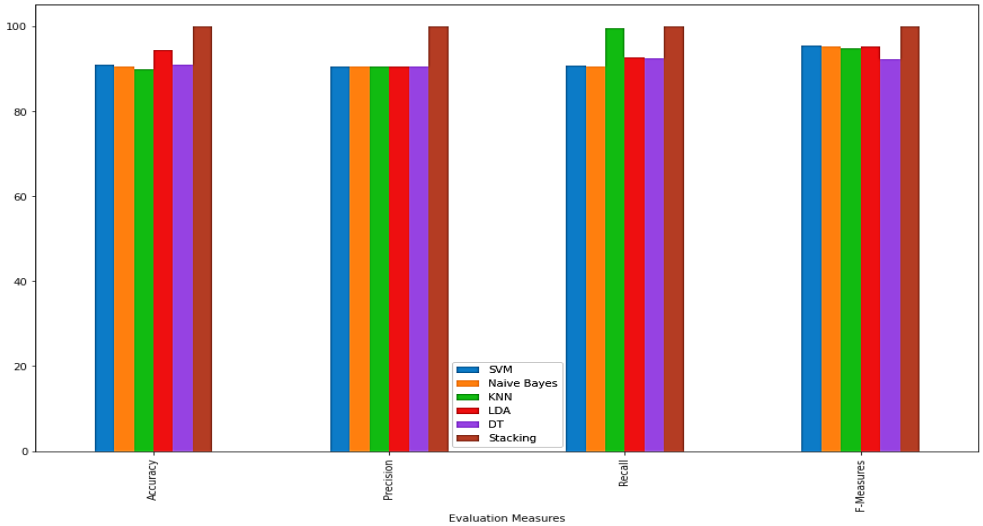

Figure 10. performance evaluation of various models

Table 2: Tabular representation of the performance of various models for intrusion detection

Model	Accuracy %	Precision %	Recall %	F-Measures %
SVM	93.04	90.54	90.70	95.04
Naïve Bayes	90.55	90.55	90.40	90.14
KNN	89.81	90.59	99.04	94.63
LDA	93.55	90.54	98.36	95.04
Decision Tree	91.00	92.40	90.55	92.04
Stacking Classifier	99.93	99.93	99.80	99.96

Results Validation:

The acquired results were tested with the help of a detailed performance analysis based on the well-known metrics, such as accuracy, precision, recall, and F-measure, and the confusion matrix to estimate classification strength. The stacking classifier suggested was trained and evaluated on a publicly accessible dataset, UNSW-NB15, such that the findings will be reproducible and reliable. Moreover, the efficiency of the proposed approach was determined by comparative analysis with the proven state-of-the-art machine learning models, including SVM, Naive Bayes, K-Nearest Neighbor (KNN), Linear Discriminant Analysis (LDA), and Decision Tree. The findings validated that the stacking classifier was significantly

better than the other evaluation models in all evaluation metrics, with an accuracy of 99.93, a precision of 99.93, a recall of 99.80, and an F-measure of 99.96. The high accuracy of the proposed model as compared to traditional classifiers confirms its strength and ability to be applied in the generalization of intrusion detection activities.

Conclusion:

This study is based on a stacking classifier for intrusion detection. An ensemble learning approach named a stacking classifier is proposed and trained on the publicly available UNSW-NB15 dataset. Six classifiers, Support Vector Machine (SVM), Naïve Bayes (NB), K-Nearest Neighbors (KNN), Linear Discriminant Analysis (LDA), Decision Tree (DT), and a Stacked Ensemble model, were evaluated using four key performance metrics: Accuracy, Recall, Precision, and F1-Measure. From the result achieved, it is evident that the Stacked Ensemble model outperforms all individual classifiers. It achieves near-perfect performance, indicating that combining multiple base models through stacking significantly enhances the detection capability and generalization of the overall system.

Acknowledgement: We would like to extend our sincere gratitude to our department, FMCS, and the AUP Peshawar.

Author's Contribution: Junaid Bakhsh's conceptualization, Muhammad Muntazir Khan (M. Muntazir Khan)'s primary writing. Farhan Nisar, funding sources, and final checking. Shakila Parveen Jan's literature preparation. Adnan Badshah methodology, Daud Shah Implementation, Muhammad Nauman Khan (M. Nauman Khan) results analysis and review. Muhammad Fawad Mian (M. Fawad Mian) validation. All authors have read and agreed to the published version of the manuscript.

Conflict of Interest: All authors have read and agreed to the published version of the manuscript in IJIST.

References:

[1] S. S. Salman Muneer, Umer Farooq, Atifa Athar, Muhammad Ahsan Raza, Taher M. Ghazal, "A Critical Review of Artificial Intelligence Based Approaches in Intrusion Detection: A Comprehensive Analysis," *J. Eng.*, 2024, doi: https://doi.org/10.1155/2024/3909173.

[2] X. Wang, L. Dai, and G. Yang, "A network intrusion detection system based on deep learning in the IoT," *J. Supercomput.*, vol. 80, no. 16, pp. 24520–24558, Nov. 2024, doi: 10.1007/S11227-024-06345-W/METRICS.

[3] H. L. Fanyi Zhao, "Application of Deep Learning-Based Intrusion Detection System (IDS) in Network Anomaly Traffic Detection," *Appl. Comput. Eng.*, vol. 86, no. 1, pp. 250–256, 2024, doi: 10.54254/2755-2721/86/20241604.

[4] A. G. Antonio Paya, Sergio Arroni, Vicente García-Díaz, "Apollon: A robust defense system against Adversarial Machine Learning attacks in Intrusion Detection Systems," *Comput. Secur.*, vol. 136, p. 103546, 2024, doi: https://doi.org/10.1016/j.cose.2023.103546.

[5] Y. Wu, B. Zou, and Y. Cao, "Current Status and Challenges and Future Trends of Deep Learning-Based Intrusion Detection Models," *J. Imaging*, vol. 10, no. 10, Oct. 2024, doi: 10.3390/JIMAGING10100254,.

[6] T. Sharath and A. Muthukumaravel, "Deep learning-powered intrusion detection systems networks using LSTM," *Adv. Intell. Networks Through Distrib. Optim.*, pp. 105–126, Aug. 2024, doi: 10.4018/979-8-3693-3739-4.CH006.

[7] F. S. Rasheed Mohammad, "Enhancing Intrusion Detection Systems Using a Deep Learning and Data Augmentation Approach," *Systems*, vol. 12, no. 3, p. 79, 2024, doi: https://doi.org/10.3390/systems12030079.

[8] S. F. Halima Sadia, "Intrusion Detection System for Wireless Sensor Networks: A Machine Learning Based Approach," *IEEE Access*, 2024, doi:

10.1109/ACCESS.2024.3380014.

[9] H. J. S. Mona Esmaeili, Morteza Rahimi, Hadise Pishdast, Dorsa Farahmandazad, Matin Khajavi, "Machine Learning-Assisted Intrusion Detection for Enhancing Internet of Things Security," *arXiv:2410.01016*, 2024, doi: https://doi.org/10.48550/arXiv.2410.01016.

[10] S. Ahmadi, "Network Intrusion Detection in Cloud Environments: A Comparative Analysis of Approaches," *Int. J. Adv. Comput. Sci. Appl.*, vol. 15, no. 3, 2024, [Online]. Available: https://thesai.org/Publications/ViewPaper?Volume=15&Issue=3&Code=IJACSA&SerialNo=1

[11] Q. O. Ahmed, "Machine Learning for Intrusion Detection in Cloud Environments: A Comparative Study," *J. Artif. Intell. Gen. Sci.*, vol. 6, no. 1, pp. 550–563, 2024, doi: 10.60087/jaigs.v6i1.287.

[12] M. M. K. Muhammad Zawad Mahmud, Shahran Rahman Alve, Samiha Islam, "Sdn Intrusion Detection Using Machine Learning Method," *arXiv:2411.05888*, 2024, doi: https://doi.org/10.48550/arXiv.2411.05888.

[13] I. I. Khan, Y. Kanaparthi, Y. Ruchandani, and A. Rizwan, "Sustainable Security Solutions for IoT: Enhancing Intrusion Detection Using AI and Machine Learning," *2024 3rd Int. Conf. Sustain. Mobil. Appl. Renewables Technol. SMART 2024*, 2024, doi: 10.1109/SMART63170.2024.10815208.

[14] A. Kukartsev, V., Kravtsov, K., Stefanenko, O., Podanyov, N., & Bezvorotnykh, "Using machine learning techniques to simulate network intrusion detection," *2024 Int. Conf. Intell. Syst. Cybersecurity*, 2024.

[15] K. R. M. Muhammad Sajid, "Enhancing intrusion detection: a hybrid machine and deep learning approach," *J. Cloud Comput.*, vol. 13, no. 1, 2024, doi: 10.1186/s13677-024-00685-x.

[16] S. R. Bhawana Sharma, Lokesh Sharma, Chhagan Lal, "Explainable artificial intelligence for intrusion detection in IoT networks: A deep learning based approach," *Expert Syst. Appl.*, vol. 238, p. 121751, 2024, doi: https://doi.org/10.1016/j.eswa.2023.121751.

[17] F. S. Ashraf, W., Ahanger, A. S., & Masoodi, "Enhancing Intrusion Detection using Supervised Machine Learning Algorithms," *2024 11th Int. Conf. Comput. Sustain. Glob. Dev.*, 2024.

[18] H. Nandanwar and R. Katarya, "Deep learning enabled intrusion detection system for Industrial IOT environment," *Expert Syst. Appl.*, vol. 249, p. 123808, 2024, doi: https://doi.org/10.1016/j.eswa.2024.123808.

[19] F. Hussain, S. A. Hassan, R. Hussain, and E. Hossain, "Machine Learning for Resource Management in Cellular and IoT Networks: Potentials, Current Solutions, and Open Challenges," *IEEE Commun. Surv. Tutorials*, vol. 22, no. 2, pp. 1251–1275, Apr. 2020, doi: 10.1109/COMST.2020.2964534.

[20] F. Hussain, R. Hussain, S. A. Hassan, and E. Hossain, "Machine Learning in IoT Security: Current Solutions and Future Challenges," *IEEE Commun. Surv. Tutorials*, vol. 22, no. 3, pp. 1686–1721, Jul. 2020, doi: 10.1109/COMST.2020.2986444.

[21] Y. X. Junxiong Chai, "Secure multi-channel information encryption based on integrated optical device," *Expert Syst. Appl.*, vol. 249, 2024, doi: https://doi.org/10.1016/j.eswa.2024.123837.

[22] M. Mynuddin *et al.*, "Automatic Network Intrusion Detection System Using Machine learning and Deep learning," *IEEE Int. Conf. Artif. Intell. Mechatronics Syst.* , Feb. 2024, doi: 10.36227/TECHRXIV.170792293.35058961/V1.

[23] F. F. Nahida Islam, "Towards Machine Learning Based Intrusion Detection in IoT

Networks," *Comput. Mater. Contin.*, vol. 69, no. 2, pp. 1801–1821, 2021, doi: 10.32604/cmc.2021.018466.

[24] S. A. Abdulazeez Khlaif, "Intrusion Detection System Based on Machine Learning Algorithms:(SVM and Genetic Algorithm)," *Babylonian J. Mach. Learn.*, 2024, [Online]. Available: https://mesopotamian.press/journals/index.php/BJML/article/view/256

[25] E. A. Muhammad Muntazir Khan, Muhammad Zubair Rehman, Abdullah Khan, "Anomaly detection in network traffic with ELSC learning algorithm," *Electron. Lett.*, vol. 16, no. 14, 2024, doi: https://doi.org/10.1049/ell2.13235.

[26] M. I. & M. A. Yousef Alhwaiti, Muntazir Khan, Muhammad Asim, Muhammad Hameed Siddiqi, "Leveraging YOLO deep learning models to enhance plant disease identification," *Sci. Rep.*, vol. 15, p. 7969, 2025, doi: https://doi.org/10.1038/s41598-025-92143-0.

[27] S. A. Lashari, M. M. Khan, A. Khan, S. Salahuddin, and M. N. Ata, "Comparative Evaluation of Machine Learning Models for Mobile Phone Price Prediction: Assessing Accuracy, Robustness, and Generalization Performance," *J. Informatics Web Eng.*, vol. 3, no. 3, pp. 147–163, Oct. 2024, doi: 10.33093/JIWE.2024.3.3.9.

[28] M. Imran, J. Usman, M. Khan, and A. Khan, "A Hybrid Deep Learning VGG-16 Based SVM Model for Vehicle Type Classification," *J. Informatics Web Eng.*, vol. 4, no. 1, pp. 152–167, Feb. 2025, doi: 10.33093/JIWE.2025.4.1.12.

[29] M. B. M. S. Asfandyar Khan, Abdullah Khan, Muhammad Muntazir Khan, Kamran Farid, Muhammad Mansoor Alam, "Cardiovascular and Diabetes Diseases Classification Using Ensemble Stacking Classifiers with SVM as a Meta Classifier," *Diagnostics*, vol. 12, no. 11, p. 2595, 2022, doi: https://doi.org/10.3390/diagnostics12112595.

[30] M. I. Muhammad Muntazir Khan, "Network Traffic Classification in SDN Networks Using PCA Integrated Boosting Algorithms," *Int. J. Innov. Sci. Technol.*, vol. 7, no. 2, pp. 856–870, 2025, [Online]. Available: https://www.researchgate.net/publication/393148501_Network_Traffic_Classification_in_SDN_Networks_Using_PCA_Integrated_Boosting_Algorithms