scientific reports

Check for updates

OPEN

# Botnet detection in internet of things using stacked ensemble learning model

Mudasir Ali[1,10], Muhammad Faheem Mushtaq[2,10], Urooj Akram[2], Daniel Gavilanes Aray[3,4,5], Manuel Masias Vergara[3,6,7], Hanen Karamti[8]✉ & Imran Ashraf[9]✉

**Botnets are used for malicious activities such as cyber-attacks, spamming, and data theft and have become a significant threat to cyber security. Despite existing approaches for cyber attack detection, botnets prove to be a particularly difficult problem that calls for more advanced detection methods. In this research, a stacking classifier is proposed based on K-nearest neighbor, support vector machine, decision tree, random forest, and multilayer perceptron, called KSDRM, for botnet detection. Logistic regression acts as the meta-learner to combine the predictions from the base classifiers into the final prediction with the aim of increasing the overall accuracy and predictive performance of the ensemble. The UNSW-NB15 dataset is used to train machine learning models and evaluate their effectiveness in detecting cyber-attacks on IoT networks. The categorical features are transformed into numerical values using label encoding. Machine learning techniques are adopted to recognize botnet attacks to enhance cyber security measures. The KSDRM model successfully captures the complex patterns and traits of botnet attacks and obtains 99.99% training accuracy. The KSDRM model also performs well during testing by achieving an accuracy of 97.94%. Based on 3, 5, 7, and 10 folds, the k-fold cross-validation results show that the proposed method's average accuracy is 99.89%, 99.88%, 99.89%, and 99.87%, respectively. Further, the demonstration of experiments and results shows the KSDRM model is an effective method to identify botnet-based cyber attacks. The findings of this study have the potential to improve cyber security controls and strengthen networks against changing threats.**

**Keywords** Internet of things network, Botnets, Cyber security, Stacking model, Machine learning

The Internet of Things (IoT) is a rapidly developing and widely adopted technology for daily life gadgets connected to the Internet. The IoT is a vast network of interconnected devices that uses information and communication technologies as its main foundation according to the International Standardization Unit (ITUT). The interconnected devices exchange large amounts of data and security is the key issue in IoT networks[1]. Today, IoT is incorporated into industries for extra reliability and time-saving[2]. However, with such wide use in daily life and industrial applications, there are significant cybersecurity concerns. Botnets have emerged as one of the largest threats, responsible for a large extent of malicious activities from distributed denial of service (DDoS) attacks to spamming and phishing[3]. In addition, recent advancements in artificial intelligence (AI) make it easier for attackers to develop fresh and intuitive ways to hack IoT systems. When it comes to defending against the specific security vulnerabilities that are anticipated in the IoT network, typical machine learning (ML) techniques and strategies are insufficient[4]. Therefore, to address the security flaws in the developing IoT network, security experts and researchers would need to assess current procedures and improve them to cope with these challenges[5].

To effectively detect botnets, it is important to understand what they are and how they work. A botnet is typically a collection of compromised workstations and servers that are used to leverage untapped processing

[1]Department of Computer Science, The Islamia University of Bahawalpur, Bahawalpur 63100, Pakistan. [2]Department of Artificial Intelligence, The Islamia University of Bahawalpur, Bahawalpur 63100, Pakistan. [3]Universidad Europea del Atlántico, Isabel Torres 21, 39011 Santander, Spain. [4]Universidad Internacional Iberoamericana, Campeche 24560, Mexico. [5]Universidade Internacional do Cuanza, Cuito, Bie, Angola. [6]Universidad Internacional Iberoamericana Arecibo, Puerto Rico 00613, USA. [7]Universidad de La Romana, La Romana, República Dominicana. [8]Department of Computer Sciences, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, 11671 Riyadh, Saudi Arabia. [9]Department of Information and Communication Engineering, Yeungnam University, Gyeongsan 38541, Republic of Korea. [10]Mudasir Ali and Muhammad Faheem Mushtaq have contributed equally to this work. ✉email: hmkaramti@pnu.edu.sa; imranashraf@ynu.ac.kr

power for computationally challenging tasks. A botnet can comprise numerous internet-connected gadgets that might have received malicious software on purpose from attackers. With botnets, it is possible to conduct DDoS attacks, steal data, and get access to devices. A malicious attack called a botnet attack employs a collection of linked computers to take down a network, network device, website, or IT environment[6]. It is only committed with the goal to obstruct ordinary business operations or to degrade the common carrier's standards. Therefore, the most important aspect of computer security would be the successful detection and prevention of botnets. Botnet detection is an essential part of cyber defense as it helps identify and stop botnets from causing damage. One popular open-source tool for botnet detection is the Snort intrusion detection system[7]. This tool is designed to monitor network traffic and identify potential threats such as botnets.

Botnet detection techniques include monitoring network traffic, analyzing system logs, and scanning for known botnet signatures. Botnet detection is crucial for protecting against cyber threats. The use of tools like Snort and understanding the characteristics of botnets can help identify and prevent potential attacks[8]. To keep IoT entities, organizations, and people safe, modern protection mechanisms should be implemented on the network. IoT botnet-based DDoS attacks, where hackers infect the devices with the script, are the main security threat[9]. Botnet detection is a challenging problem in cyber security. Botnet detection often involves analyzing large volumes of network traffic and system logs. Ensuring the accuracy and completeness of this data can be difficult, particularly in complex network environments where there are many different types of devices and protocols[10]. Botnets can be very large, involving thousands or even millions of infected devices. Detecting and mitigating these threats at scale requires sophisticated tools and techniques, as well as a well-trained security team. A variety of ML techniques can be used to identify and segregate these botnet devices when more devices become candidates to be botnet devices[11].

As more units become candidates for being botnet devices, identifying and discriminating these botnet units can be automated using ML techniques. For this purpose, state-of-the-art ML algorithms need to be utilized to identify botnets and malicious visitors. Today, ML approaches are being used more and more in various social, medical, business, and industrial applications[12]. Existing studies have demonstrated that ML can be potentially important in identifying botnet attacks. The creation of a botnet detection model utilizing ML can significantly increase the botnet detection accuracy and reduce the processing time[13]. Keeping in view the performance and potential of ML models, this study presents an ML approach for automated botnet attack detection and makes the following contributions.

- A botnet detection approach is proposed in this study to enhance IoT network security based on ML. The proposed approach is a stacking model comprising k nearest neighbor (KKN), support vector machine (SVM), decision tree (DT), random forest (RF), and multilayer perceptron (MLP) as the base classifier while logistic regression (LR) is used as the meta classifier.
- The dataset used for experiments consists of various botnet attacks including generic, exploits, fuzzers, denial of service (DoS), reconnaissance, analysis, backdoor, shellcode, and worms attack types. Preprocessing is applied to the dataset such as null value removal and categorical value handling using label encoding to improve model performance.
- A thorough evaluation is carried out comprising a variety of metrics. Additionally, k-fold cross-validation is employed to investigate the generalizability of the proposed approach. Performance comparison with existing approaches is also carried out for further corroboration.

The preceding part is distributed into the following sections: Section "Related work" presents the literature review of the current research works and technologies. Section "Materials and methodology" involves introducing the overall research approach, describing the data collection methods, and discussing the models utilized in this study. Section "Results and discussion" shows the results and discussion of the proposed approach. Section "Conclusion" presents the conclusion.

## Related work

This section discusses the existing botnet detection and prevention strategies using ML models.

Botnets can be used for a variety of cyber-attacks and distributed denial of attacks (DDoS) are among the most common attacks. DDoS attacks can be launched by Botnets on targeted servers or applications by flooding them with a high volume of traffic, rendering them unavailable to legitimate users. With botnets, it is possible to conduct distributed denial-of-service (DDoS) assaults, steal data, and get access to devices[14]. In addition, cyber-attack fuzzers combined with algorithmic learning for machines, support vector machines (SVM), artificial neural network (ANN), k-nearest neighbor (KNN), decision tree (DT), and multilayer perceptron (MLP) are examples of these algorithms. The most accurate algorithms are SVM and KNN as reported in[15].

### Cyber security

A successful botnet detection strategy is necessary for preserving cyber security because botnets are one of the most important cyber security risks in recent years. Infected computer networks known as botnets are used by a bot master to conduct illegal operations like spamming, DDoS attacks, data theft, etc.[16]. There are several widely used strategies for overall cyber security. In order to address cyber security challenges, we primarily focus on approaches involving the use of ML. The outcomes show that this approach outperforms neural network and tree algorithm-based alternatives in terms of detection rate, processing time, and computational cost, and achieved an accuracy of 94%[17].

Cyber security is the protection of data programs and connections between computers from various undesirable threats like unauthorized attacks, alteration, and fabrications represented by systems or software because traditional security techniques are insufficient for identifying network security. The study[18] developed

an intrusion detection system (IDS) to look into and ascertain the system's security. The authors investigated various ML models to categorize cyber-attacks and report results for the models such as Bayesian network (BN) 90%, Naive Bayes (NB) 91%, random forest (RF) 94%, DT 93%, random tree (RT) 90%, decision table (DTb) 92%, and ANN 91%[19].

One of the well-known technologies used in the cyber business is signature-based network intrusion detection. The intrusion DTree[20] model was integrated with a number of well-known ML algorithms, including NB, LR, SVM, and KNN in order to evaluate the performance of the system security model. NB, LR, SVM, and KNN show accuracy scores of 0.90, 0.94, 0.96, and 0.94, respectively while the Intrusion DTree model showed superior performance with a 0.98 accuracy score. Malicious attack technologies are developing more quickly than defense methods. The objectives of cyber security are data protection, resource protection, data privacy, and data integrity. In this context[21], carried out experiments involving SVM, DT, deep belief network (DBN), RF, NB, and ANN models and reported excellent results with DT.

Along the same lines[22], utilized different models for cyber security evaluation and reported promising results. NB, SVM, and KNN achieved 95.55%, 97.78%, and 93.34% accuracy, respectively. In addition, the study[23] utilized KNN, RF, DT, and MLP with the DT classifier obtaining the best performance.

## Machine learning

Various ML models have been utilized for botnet attack detection. For example, a KNN is adopted for botnet attack detection in[24] along with other ML models. KNN obtained an accuracy of 98.55% while SVM, DBN, recurrent neural network (RNN), convolutional neural network, and DT obtained an accuracy of 82.31%, 93.49%, 77.55%, 99.41%, and 99.8%, respectively. The non-parametric KNN method is applied to classify various patterns. It determines how similar all training samples are to the unlabeled samples[25]. Each of the training samples has been assigned and is a vector with a class label in a multidimensional feature space. KNN chooses 6% and 26% fewer characteristics than PSO-KNN and GA-KNN, respectively. As can be shown, TSA-KNN not only performs superior to the other two algorithms with respect to speed but also accuracy (5.5% and 10.8% higher). In essence, TSA-KNN outperforms the competition in terms of total performance[26].

SVM also is frequently employed for classification and utilizing datasets with a high dimensionality is a strength of SVM. SVM, RF, bagging, DT, and KNN are used in[27] for malware detection. Using the IoT malware dataset, SVM showed a 99.3% accuracy. Similarly[28], utilized SVM for intrusion detection with an accuracy of 69.79%, while recall, precision, and F1 scores are 0.80, 0.70 and (0.65, respectively.

The study[29] uses a dataset gathered from object technology and data security (OTDS)[30] and Czech University (CTU-13) for experiments involving SVM, DT and MLP. DT model exhibits good performance with an accuracy of 92% while the precision, recall and f1 scores are 0.922, 0.922, and 0.92, respectively. Similarly[31], employed DT to classify network packets into normal and malicious traffic. An accuracy of 97.2% is reported.

In[32], RF performs well for the majority of attack types with accuracy higher than 95%, whereas KNN has slightly lower accuracy. RF performed better achieving 95.7% accuracy and 93.9% recall. An ML-based botnet detection model is developed by utilizing domain name service query data and assesses its performance using well-known ML methods[33]. The findings of the research demonstrate the viability of applying ML techniques for botnet detection, with the random forest method providing the best overall detection accuracy of over 90%.

Employing MLP for intrusion detection and comparing it to eight different classifiers or methodologies, 99.63% of true positive attacks are detected in[34]. With a false positive rate of just 0.47 percent, this intrusion detection system is effective. Similarly[35], used DT, MLP, NB, and SVM for botnet attack detection. The analysis showed that DT achieves the highest accuracy and provides the best performance. The obtained accuracy is 99.3%, 98.9%, 96.9%, and 97.1% for DT, MLP, NB, and SVM, respectively[36].

Another important work in this context is[37] which discusses the vulnerabilities of deep learning approaches with respect to malware detection. The study investigated data poisoning with noise injection and gradient-based data poisoning in detail employing several DL models and exposing their weaknesses for such attacks. The authors propose a framework called 'Deep Image' in[38] to perform malware detection in an IoT environment. Dynamic features are obtained from the attacks and converted into binary images. The transformed images are later used with clustering, probabilistic and DL approaches for attack classification. The study reports superior results for the proposed approach.

## Materials and methodology

Since most attacks in the IoT context occur in real time, a quick attack detection system with improved accuracy is needed. This might be accomplished by utilizing fewer features, which would simplify the system and speed up execution thereby being useful in the real-time detection of attacks. We suggest using ML as a strategy for Botnet detection in IoT devices to address this problem. Figure 1 illustrates the architecture of the proposed system. It is based on the stacking of KNN, SVM, DT, RF, and MLP, called KSDRM.

The proposed research focused on analyzing the performance of ML classification for botnet detection. The ML classifiers, such as KNN, SVM, DT, RF, MLP, and the proposed model KSDRM have been implemented using the COLAB platform. The UNSW-NB15 dataset is used to train ML models. Figure 2 shows the workflow of the proposed approach. The dataset used for experiments consists of 82332 occurrences and 45 features based on IoT network communication. A train-test split is mostly used to estimate how well a model will perform on untested data. The train test split distributes the dataset into training and testing data with a ratio of 80:20. The testing set is used to assess the model's performance while the training set is used to fit the model. Six ML models such as KNN, SVM, DT, RF, MLP, and stacking KSDRM.

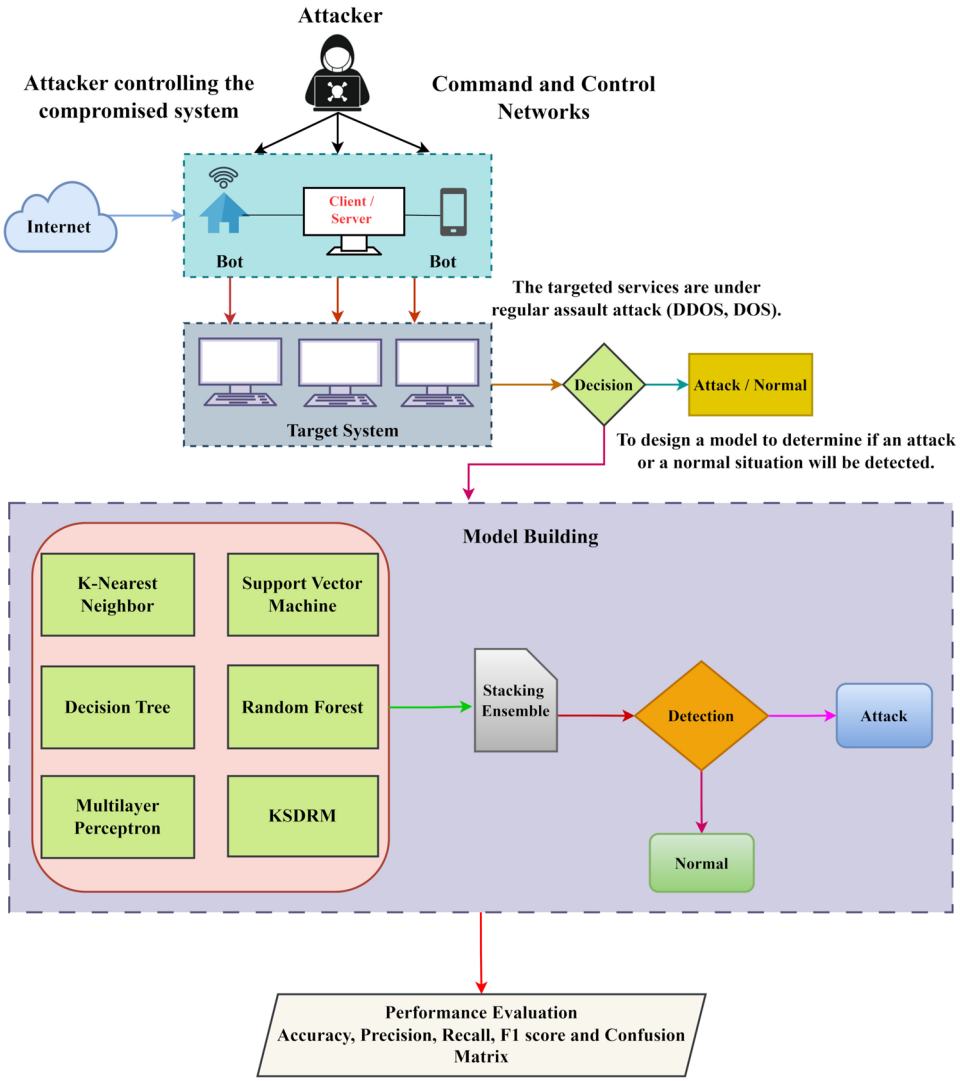



**Fig. 1**. Botnet detection system to enhance the security using ML.

## Dataset description

The dataset is collected from the well-known dataset repository Kaggle. The UNSW-NB15 dataset comes with both a training set and a testing set. UNSW_NB15_training-set.csv and UNSW_NB15_testing-set.csv are the training set and testing set for the dataset, respectively. The main dataset used to evaluate the performance of the model is the training file set, which is then partitioned into training and testing datasets for additional processing in the ratio of 0.8 to 0.2. The data consists of 82332 instances with 45 features. The details of the dataset are given in Table 1.

## Data preprocessing

Data preprocessing has been applied to the dataset to prepare the data for model training and testing. In order to do this, the dataset must be loaded, cleaned, adjusted, and changed into a form that is suitable for the ML models. The dataset suggests that class distribution is imbalanced, as shown in Fig. 3. Label encoding is used to convert categorical features to numerical values.

## Categorical value handling

When working with categorical data, one common task is to encode the categorical values into numerical representations that can be used by ML algorithms. One popular technique for this purpose is using a label encoder, which assigns a unique numerical label to each category in the data. To handle missing values in categorical data while using a label encoder, a few approaches can be used. One option is to replace the missing values with a specific placeholder, such as 'NaN', before applying the label encoder. Label encoding is available as a feature in various libraries, including scikit-learn. The Label Encoder class in scikit-learn can be used for this purpose, focusing on encoding target labels rather than input features.

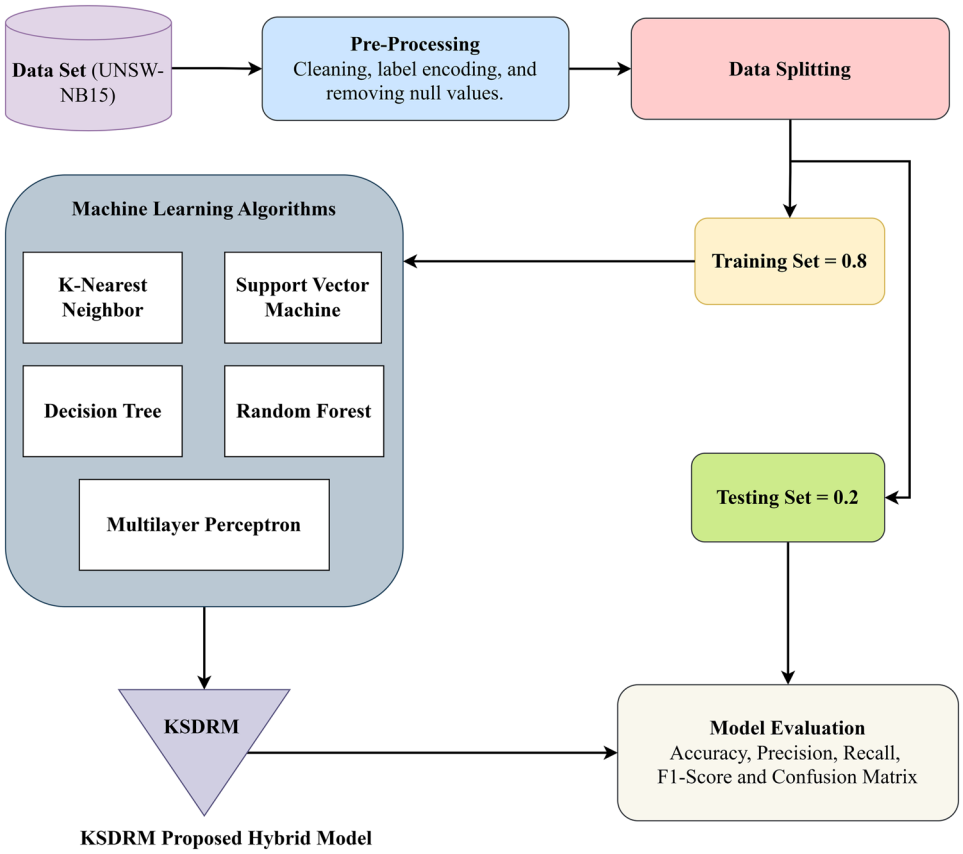



**Fig. 2**. Workflow of the proposed approach.

| Attack type | Absolute frequencies | Absolute percentages |
|---|---|---|
| 1 | 45332 | 0.5506 |
| 0 | 37000 | 0.4494 |

**Table 1**. The statistics of the training dataset.

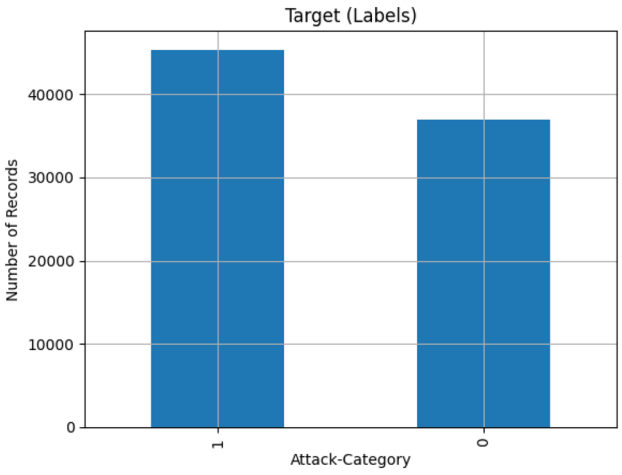



**Fig. 3**. Distribution of records for different classes.

*Machine learning algorithms*

ML algorithms are computational models designed to automatically predict the label of a class, rather than relying on explicit programming by a developer. This study employs the following popular ML algorithms.

- K-Nearest neighbors: KNN is an easy-to-interpret algorithm that categorizes new data points based on how similar they are to nearby data points. Based on the majority class of the data point's closest neighbors, a new sample is assigned to a class. In this model, the value of *k*, which denotes the number of neighbors, is critical and should be selected based on the properties of the information.
- Support vector machines: SVM is a potent method used for both regression and classification tasks. It locates the best hyperplane in the data space for the greatest class separation. SVM is useful for handling complicated and nonlinear data. It seeks to produce decision boundaries with the widest margin between classes. It is renowned for handling small to medium-sized datasets efficiently and can handle high-dimensional data as well.
- Decision tree: DT is a hierarchical model that produces predictions using a tree-like structure. Each leaf node represents a class label, whereas each internal node represents a test on an attribute and its result. DT can handle categorical and numerical data and is easily interpretable and comprehendible. However, if it is not correctly regularized, it could experience overfitting.
- Random forest: To generate predictions, an ensemble learning model called RF, combines different decision trees. It generates a collection of decision trees, each of which independently predicts the outcome. The projections of each individual tree are combined to get the final prediction. The robustness and capacity of RF to manage high-dimensional data are well recognized.
- Multilayer perceptron: An artificial neural network called MLP, is made up of many layers of interconnected nodes (neurons). It is a flexible algorithm used for both regression and classification tasks. Through its hidden layers, MLP may discover intricate relationships and patterns in data. By adjusting the weights and biases, MLP can approximate any function, making it a powerful algorithm. However, training an MLP requires careful selection of network architecture and tuning of hyperparameters to achieve optimal performance.

## Proposed stacking KSDRM model

An ML technique called stacking is used to integrate the results of various classification models to get a more potent and precise final prediction. To improve the overall predictive performance, stacking combines several ML techniques. In this study, KNN, RF, SVM, DT, and MLP are combined. By stacking models, we can benefit from the strengths and diversity of different algorithms. Each algorithm may have its own unique way of capturing the data's patterns that make a prediction more reliable and precise. Stacking helps in reducing bias and variance, improving generalization, and reducing overfitting to the training data.

Stacking is a powerful ensemble learning technique that leverages the strengths of multiple ML algorithms. Stacking can enhance the overall predictive performance by combining the predictions in a layered fashion, leading to improved accuracy and robustness in ML models. Thus, the proposed KSDRM model's architecture is displayed. The strength of several ML algorithms, including KNN, SVM, DT, RF, and MLP, has been harnessed in stacking, a type of sophisticated ensemble learning. In contrast to ensemble approaches (KNN+SVM+DT+RF+MLP), which combine several models for greater generalization and interpretability, ML stacking methods take more data and compute resources and provide less interpretability, even when they address very complex problems. By layering predictions together, this stacking can increase the accuracy and resilience of ML models, which in turn can enhance overall prognostic performance. Stacking becomes more resilient to different data distributions when an ensemble approach is used. Furthermore, as computational complexity rises, the model becomes more challenging to comprehend, necessitating a thorough evaluation of the combination and training procedure. Combining models under stacking produces accurate outcomes, while individual models won't produce very good results. To evaluate the final prediction made via stacking ensembles for predictive modeling, LR is used, as shown in Fig. 4.

The decision to use a stacking ensemble model over deep learning techniques such as CNNs, LSTMs, or RNNs stems from both practical and performance-based considerations. While deep learning models excel at capturing complex temporal and spatial dependencies in IoT data, they often require large datasets, intensive computational resources, and longer training times. In contrast, stacking leverages the strengths of multiple lightweight models, combining their diverse learning behaviors through a meta-learner for better generalization and reduced overfitting. Experiments showed that KSDRM performs competitively without the overhead of deep neural networks. Additionally, stacking offers greater interpretability and is less vulnerable to adversarial manipulation compared to deep learning, as highlighted in recent studies. For resource-constrained environments like IoT-based botnet detection, the stacking model provides a more scalable, explainable, and efficient solution. Including deep learning models in future comparisons would be valuable, but the results show that stacking offers an optimal balance between accuracy, efficiency, and robustness.

In addition to base models, the choice of using the LR model as a meta-learner comes from its unique advantages. The LR is more suitable for interpretability compared to other ML models. Being a linear model it is marked by low variance. In addition, when used as a meta-learner it does not add complexity to the based models, thereby making it suitable to reduce the probability of overfitting. It is also computationally efficient to train and particularly useful when the stacking involves multiple models which is the case in the current study. The weighted average of the based models can be efficiently obtained by the LR model and it can decide the weights of based models, both for strong and weak models. Using LR as the meta-learner improves accuracy better than voting.

Stacking, also known as stacked generalization, is an ensemble approach that reduces the generalization error of many base models by combining their predictions.

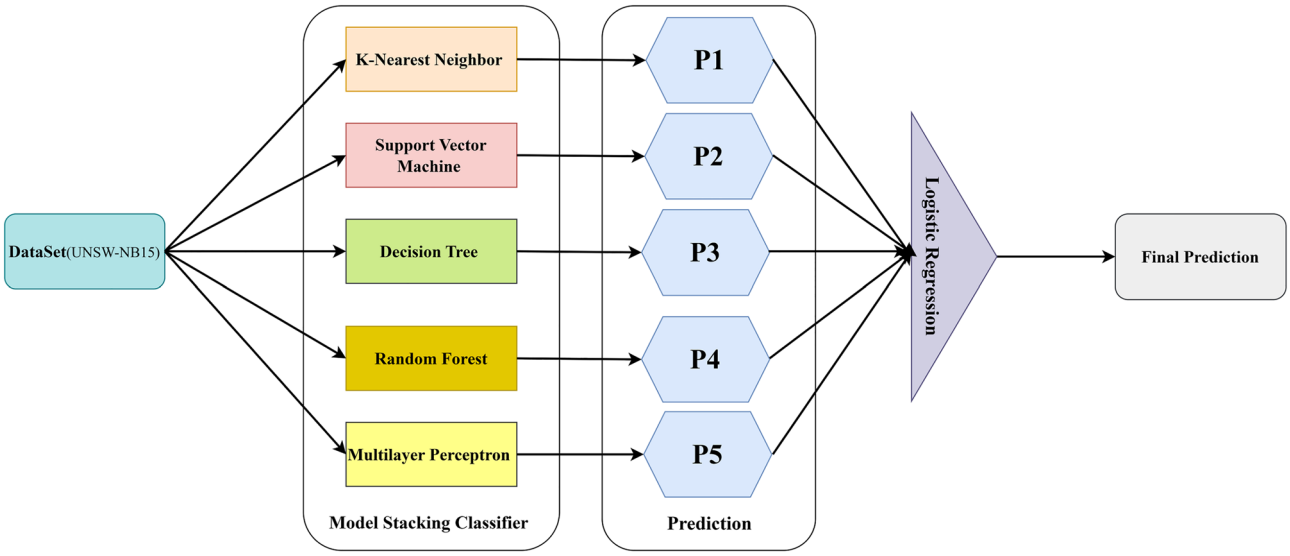



**Fig. 4**. Proposed stacking model.

Base models:

$$h_1(x), h_2(x), \ldots, h_n(x)$$

Final prediction by meta-model:

$$\hat{y} = f(h_1(x), h_2(x), \ldots, h_n(x))$$

Having low bias but high variance, the realm of high overfitting provides an example:

- Decision Trees can memorize training data (low bias, high variance).
- The small data set and poorly tuned kernel may lead to overfitting in SVM.

A single model's error can be decomposed as:

$$\text{Error} = \text{Bias}^2 + \text{Variance} + \text{Irreducible Error}$$

Overfitting = High Variance Poor generalization on unseen data.

Ensemble learning generates a model capable of blending the best entity predictions so as to enhance all:

Variance reduction: The variance of an ensemble (with uncorrelated models) is reduced:

$$\text{Var}_{\text{ensemble}} = \frac{1}{n^2} \sum_{i=1}^{n} \text{Var}(h_i)$$

This means the ensemble is more stable than any single model.

Bias-Variance Trade-off: Stacking helps reach a balance by averaging out the error of DT, an underfitted model (like KNN on complex data), and other models.

$$\text{Bias}_{\text{meta}} \approx \text{low}, \quad \text{Variance}_{\text{meta}} \ll \text{any single model}$$

*Validation result*

The essential evaluation method for the KSDRM architecture is k-fold cross-validation. The idea here is to split the data into k folds, or sections, of approximately equal size. The remaining folds serve as the model's training data, and the model is utilized as a separate split for validation k times during the training process. This lowers the chance of overfitting or underfitting by offering a thorough evaluation of that model. Lastly, for a more accurate estimation of the model performances, the performance metrics extracted from each fold can be averaged. This comprehensive view sheds light on the model's capacity for generalization and opens the door for certain areas of development. The proposed KSDRM is shown in Fig. 5.

## Evaluation parameters

Well-know metrics for evaluating the effectiveness of ML models, particularly for classification tasks, include accuracy, recall, precision, and F1 score. These metrics are used to evaluate how accurately the model predicts various classes of the input data. Accuracy is calculated using.

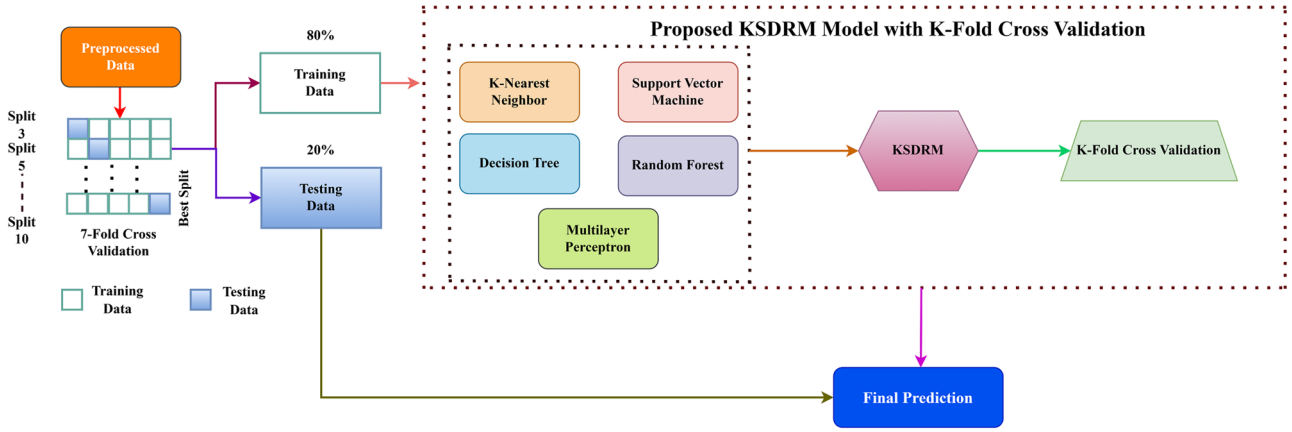



**Fig. 5**. Structure of the proposed approach using the k-fold cross-validation with the KSDRM model.

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{1}$$

where *TP* stands for true positive, which is the number of attacks that are accurately predicted, *FP* is false positive which refers to a normal packet that is mistakenly detected as an attack, TN stands for true negative, demonstrating the normal packet predicted as normal packet and FN is false negative denotes an attack that is predicted as a normal packet.

Although accuracy is a commonly used statistic, it may not be enough especially when working with unbalanced datasets. Recall also referred to as sensitivity or true positive rate, is the ratio of correctly predicted positive cases to actual positive occurrences. It shows that the model is capable of correctly identifying all pertinent positive cases. Increased recall value implies that the model is successful in reducing false negatives.

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

Precision also known as positive predictive value is the ratio of correctly predicted positive instances to all predicted positive instances. It shows how accurate the model is at identifying positive cases and reducing false positives. An elevated value suggests that the model produces few false positives.

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

A frequent statistic used to assess the efficacy of binary classification algorithms is the F1 score. By merging precision and recall into a single score, it provides a fair evaluation of the model's correctness. The following formula is used to determine the F1 score.

$$F1score = \frac{2 * (Precision * Recall)}{(Precision + Recall)} \tag{4}$$

## Results and discussion

This section provides experimental results and a discussion of the results. The effectiveness of ML models is assessed using accuracy, precision, recall, and F1 score. All models are implemented in Python programming language on Colab Notebook on a system running on a Windows 10 operating system with 16GB RAM.

### Performance of machine learning models

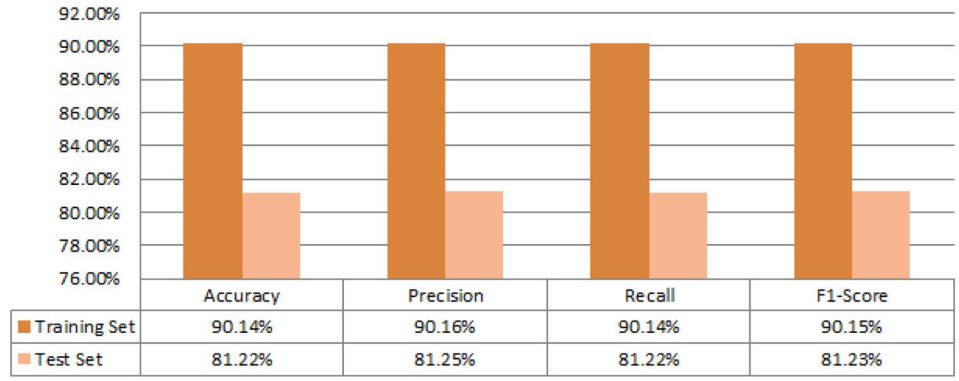
Various models are employed to analyze their performance in comparison to the proposed approach. Table 2 displays the results of all ML models that are utilized in this study. Experimental results reveal that RF obtains the best performance for botnet attack detection with 97.34% accuracy. It is followed by a 92.45% accuracy by DT. SVM and MLP show poor performance with 75.62% and 75.66% accuracy, respectively while KNN obtains an 81.22% accuracy.

Models show different results for training and test datasets, as shown in Figs. 6 and 7. KNN successfully predicted using the training set with an accuracy of 90.14%, precision of 90.16%, recall of 90.14%, and F1 score of 90.15%. This performance is reduced when using a testing set with an accuracy of 81.22%, precision of 81.25%, recall of 81.22%, and F1 score of 81.23%.

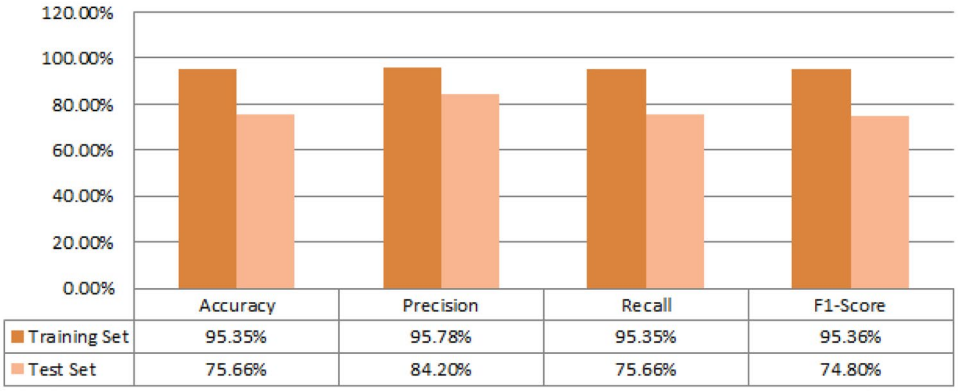
SVM works by discovering the most suitable hyperplane that maximizes the margin between the extraordinary instructions of statistics points. With accuracy, precision, recall, and F1 score of 95.35%, 95.78%, 95.35%, and

| Model | Accuracy (%) | Precision (%) | Recall (%) | F1 Score (%) |
|---|---|---|---|---|
| KNN | 81.22 | 81.25 | 81.22 | 81.23 |
| SVM | 75.66 | 84.20 | 75.66 | 74.80 |
| DT | 92.45 | 92.45 | 92.45 | 92.45 |
| RF | 97.34 | 97.35 | 97.34 | 97.34 |
| MLP | 75.62 | 77.96 | 75.62 | 75.53 |

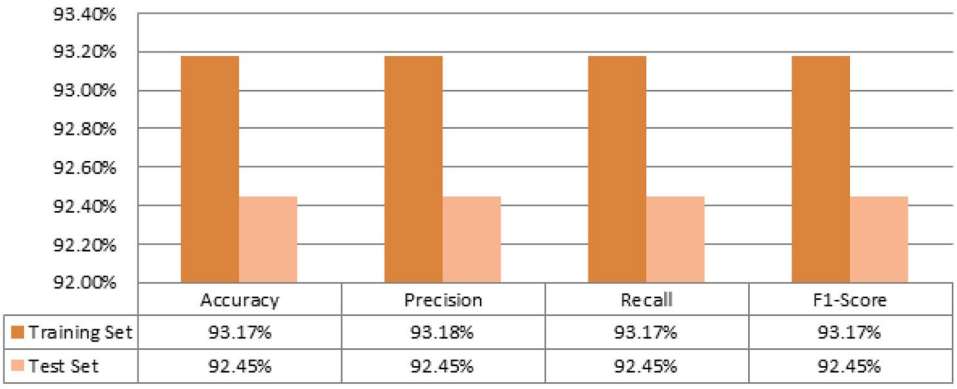
**Table 2**. Experimental results of all ML models.




(a)

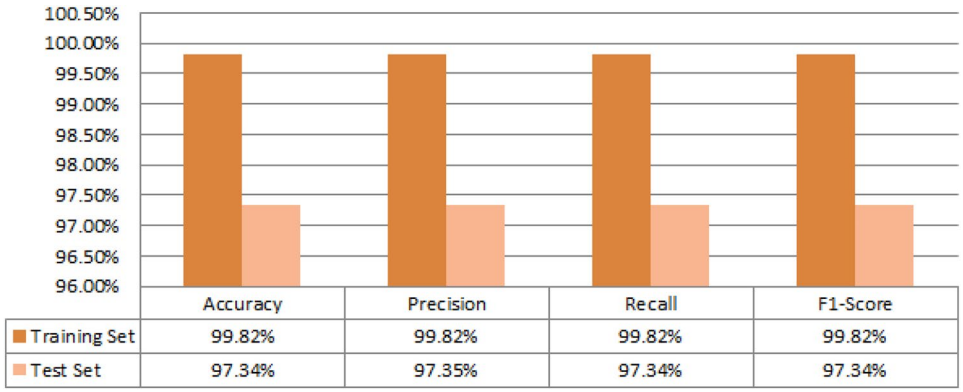



(b)



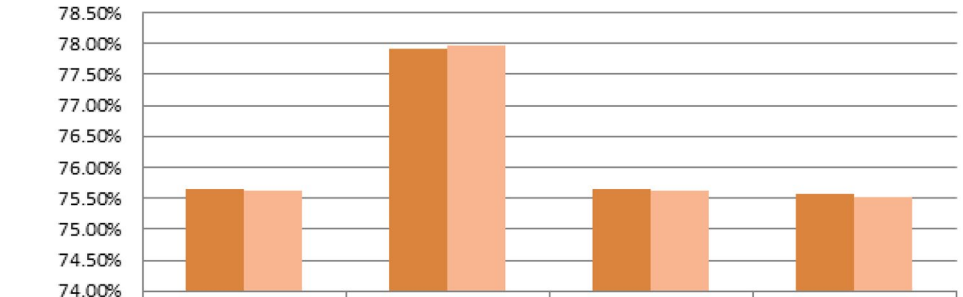

(c)

**Fig. 6**. Performance analysis of ML models used in this study, (**a**) K nearest neighbor, (**b**) Support vector machine, and (**c**) Decision tree.




| | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| Training Set | 99.82% | 99.82% | 99.82% | 99.82% |
| Test Set | 97.34% | 97.35% | 97.34% | 97.34% |

(a)




| | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| Training Set | 75.65% | 77.91% | 75.65% | 75.58% |
| Test Set | 75.62% | 77.96% | 75.62% | 75.53% |

(b)

**Fig. 7**. Performance analysis of ML models used in this study, (**a**) Random forest, and (**b**) Multilayer perceptron.

95.36%, respectively, it shows superb performance on the training set. However, for the testing set it obtains an accuracy of 75.66%, precision of 84.20%, recall of 75.66%, and F1 score of 74.80%.

The effectiveness of the DT algorithm was evaluated. With an accuracy of 93.17%, it correctly predicted the outcomes of the model performance on the training set. It also obtains good results regarding precision, recall, and F1 scores with 93.18%, 93.17%, and 93.17%, respectively. DT obtains very good results for the testing set as well and achieves 92.45% accuracy while precision, recall, and F1 scores are 92.45%, 92.45%, and 92.45%, respectively.

RF model is recognized for its potential to manage complicated information and high-dimensional feature spaces. It is also efficient in handling model overfitting. It showed superior performance both with the training and testing data. For the training dataset, it obtains 99.82% accuracy, 94.82% precision, 99.82% recall, and 99.82% F1 score. It outperforms all other models with a superb 97.35% accuracy for the testing test. Similarly, superior results are obtained concerning precision, recall, and F1 scores.

MLP is a powerful supervised learning model that is highlighted for its ability to learn the complex relationship between the data and class labels. Despite its ability to produce good results, its performance in botnet attack detection is not good. It obtains an accuracy of 75.65%, precision of 77.91%, recall of 5.65%, and F1 score of 75.58%, which is inferior to those of RF and DT models. Its performance with the testing set is also mediocre with an accuracy of 75.62%, precision of 77.96%, recall of 75.62%, and F1 score of 75.53%.

Figure 8 illustrates the performance of the models regarding correct and false predictions for normal and attack traffic. The diagonal values show the correct predictions concerning TP and TN. It shows that KNN has 13,374 correct predictions while 3,093 predictions are wrong. A total of 12,459 predictions are correct while 4,008 are wrong predictions. For DT, 15,224 predictions are correct with only 1,243 wrong predictions making it one of good performing models for botnet attack detection. On the other hand, RF shows superb performance with a remarkable 16,029 correct predictions and has only 438 wrong predictions. FN and FP predictions by RF are also the lowest showing its performance concerning specificity and sensitivity. MLP shows poor performance with 4,015 wrong predictions and 12,452 correct predictions, thereby showing the least number of correct predictions among the employed models.

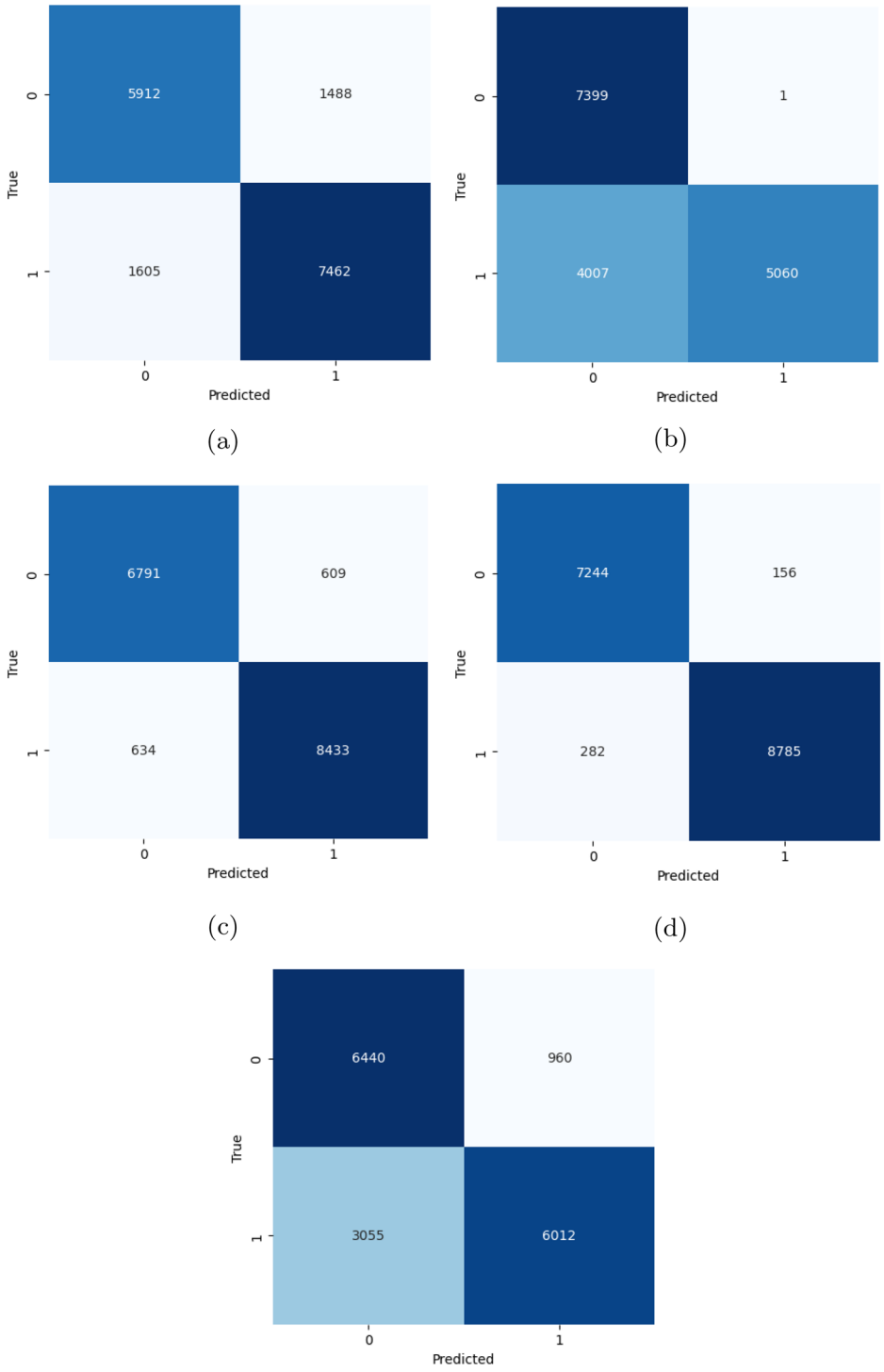



**Fig. 8**. Confusion matrices of all ML models, (**a**) K nearest neighbor, (**b**) Support vector machine, (**c**) Decision tree, (**d**) Random forest, and (**e**) Multilayer perceptron.

## Performance of proposed stacking model

The proposed model is tested in comparison to other employed models. Experimental results show that it obtains the best results compared to other models with an accuracy of 97.94% which is the highest among all models. Table 3 shows the results of all employed models concerning the accuracy, precision, recall, and F1 score. Results indicate that the proposed model performs better than other models and obtains the best results for all performance evaluation metrics.

| Models | Accuracy (%) | Precision (%) | Recall (%) | F1-Score (%) |
|---|---|---|---|---|
| KNN | 81.22 | 81.25 | 81.22 | 81.23 |
| SVM | 75.66 | 84.20 | 75.66 | 74.80 |
| DT | 92.45 | 92.45 | 92.45 | 92.45 |
| RF | 97.34 | 97.35 | 97.34 | 97.34 |
| MLP | 75.62 | 77.96 | 75.62 | 75.53 |
| Stacking | 97.94 | 97.53 | 97.48 | 97.98 |

**Table 3**. Analysis of all employed models.

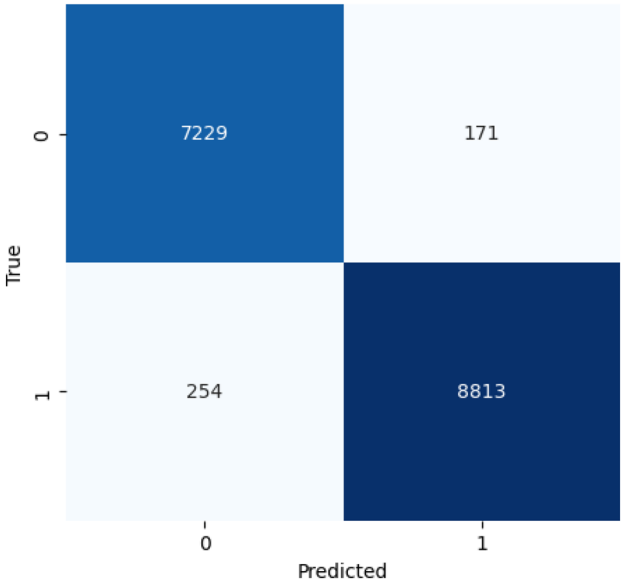



**Fig. 9**. Confusion matrix of the proposed stacking model.

| Fold | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| 1 | 0.9982 | 0.9983 | 0.9980 | 0.9981 |
| 2 | 0.9981 | 0.9979 | 0.9982 | 0.9980 |
| 3 | 0.9976 | 0.9975 | 0.9978 | 0.9976 |

**Table 4**. Model performance with k =3.

Figure 9 shows the performance of the proposed stacking model concerning the number of correct and wrong predictions. It indicates the model makes 16,042 correct predictions which are higher than RF which obtained the best performance among all the models. Similarly, the 425 wrong predictions by the proposed stacking model are less than the best-performing RF model which has 438 wrong predictions. Combining multiple models in stacking tends to overcome the deficiencies of various models and produces good results.

## Results of K-fold cross-validation

The proposed approach has to use k-fold cross-validation in order to verify the KSDRM model's performance. Tables 4, 5, 6 and 7 display the outcomes for k values 3, 5, 7, and 10, in that order. The outcomes demonstrate that Table 6 with k=7 performs noticeably better than any other table. With an f1 score of 99.87%, 99.89% accuracy, 99.88% precision, and 99.89% recall, it effectively strikes a solid compromise between separating positive prediction occurrences from proven dependability in classification overall. Tables 4, 5, 6 and 7 displays the measurement's accuracy, f1 score, recall, and precision validation findings. Additionally, each of these findings is significant for evaluating the classifiers' botnet detection capabilities.

## Performance comparison

In order to assess the effectiveness of the proposed model, a performance comparison with current state-of-the-art models is desired. Several previous studies that used ML models for botnet detection have been chosen for this purpose. For example, studies[39–42] deployed RF and DT models and obtained very good results. Similarly[43],

| Fold | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| 1 | 0.9984 | 0.9982 | 0.9986 | 0.9984 |
| 2 | 0.9980 | 0.9978 | 0.9981 | 0.9979 |
| 3 | 0.9978 | 0.9975 | 0.9979 | 0.9977 |
| 4 | 0.9983 | 0.9980 | 0.9985 | 0.9982 |
| 5 | 0.9981 | 0.9980 | 0.9982 | 0.9981 |

**Table 5**. Model performance with k =5.

| Fold | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| 1 | 0.9980 | 0.9981 | 0.9982 | 0.9983 |
| 2 | 0.9985 | 0.9983 | 0.9986 | 0.9987 |
| 3 | 0.9989 | 0.9988 | 0.9989 | 0.9987 |
| 4 | 0.9982 | 0.9984 | 0.9983 | 0.9985 |
| 5 | 0.9986 | 0.9985 | 0.9984 | 0.9988 |
| 6 | 0.9987 | 0.9989 | 0.9986 | 0.9985 |
| 7 | 0.9984 | 0.9983 | 0.9989 | 0.9982 |

**Table 6**. Model performance with k =7.

| Fold | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| 1 | 0.9981 | 0.9981 | 0.9982 | 0.9981 |
| 2 | 0.9983 | 0.9983 | 0.9983 | 0.9984 |
| 3 | 0.9981 | 0.9981 | 0.9982 | 0.9981 |
| 4 | 0.9982 | 0.9983 | 0.9982 | 0.9982 |
| 5 | 0.9983 | 0.9985 | 0.9983 | 0.9983 |
| 6 | 0.9982 | 0.9979 | 0.9982 | 0.9982 |
| 7 | 0.9983 | 0.9983 | 0.9985 | 0.9982 |
| 8 | 0.9983 | 0.9983 | 0.9983 | 0.9982 |
| 9 | 0.9979 | 0.9979 | 0.9979 | 0.9979 |
| 10 | 0.9981 | 0.9980 | 0.9982 | 0.9981 |

**Table 7**. Model performance with k =10.

proposed an SVM model and utilized the same dataset that is used in this study. Both[44] and[45–47] made use of KNN, MLP, and RF models for the same task while[48] leveraged an SVM model. Results comparison given in Table 8 reveals that the proposed model outperforms these studies and obtains a better accuracy using the same dataset UNSW-NB15.

## Limitations of study

In spite of the better results compared to existing approaches, this study has several limitations. The proposed KSDRM model demonstrates high accuracy and robustness in detecting botnet attacks, however, resilience against adversarial machine learning (AML) attacks remains an important area for future consideration. Adversarial tactics, such as evasion attacks, can subtly manipulate input data to bypass detection without altering malicious behavior. Traditional models in the KSDRM stacking like DTs and RF offer some natural resistance to gradient-based attacks due to their non-differentiable nature, unlike neural networks. However, a dedicated adversarial robustness evaluation was not conducted in this study. Incorporating adversarial training, defensive distillation, or input sanitization techniques could further enhance the security of the model. We acknowledge this as a limitation and propose that future work should focus on systematically testing and improving the model's defenses against such adversarial threats, ensuring more secure deployment in IoT environments.

## Conclusion

Botnet detection is a difficult yet important task in computer security. Since botnet attacks are evolving over time, attackers are always coming up with new strategies to make it hard to detect and stop such attacks. This makes it difficult for security experts to stay up-to-date on threats and modify their detection techniques as necessary. Machine learning models can be employed for botnet and malicious traffic identification. This study proposes a stacking model where various models are combined to leverage the advantage of their predictive capability. Experiments are performed employing five classifiers DT, RF, SVM, KNN, and MLP to evaluate their

| References | Model | Accuracy (%) | Precision (%) | Recall (%) | F1 score (%) |
|---|---|---|---|---|---|
| [39] | RF | 92.5 | 88.9 | 88.7 | 92.93 |
| [40] | DT | 81.42 | – | – | – |
| [44] | KNN | 72.30 | – | – | – |
| [45] | MLP | 75 | 71 | – | – |
| [41] | DT | 85.56 | – | – | – |
| [46] | MLP | 74.90 | – | – | – |
| [47] | RF | 75.56 | – | – | – |
| [48] | SVM | 65.3 | 99.8 | 49.2 | 65.9 |
| [42] | RF | 86.16 | – | – | – |
| [43] | SVM | 77.08 | 77 | 77 | 77 |
| [49] | RF | 96.89 | 96.91 | 97.41 | 97.16 |
| [50] | XGBoost | 95.5 | – | – | – |
| [51] | DeepSVM | 98 | 97 | 98 | – |
| [52] | RF | 92 | 88 | 89 | 89 |
| [53] | Voting Classifier | – | – | – | 97.79 |
| [54] | RF Gini | 92.61 | 91 | 89 | – |
| [55] | KNN | 99 | 98 | 98 | 98 |
| [56] | DT+RFXGBoost | – | – | – | 97.77 |
| [57] | KNN+RF | 87.10 | 81.79 | 98.48 | 89.36 |
| [58] | RF | 94 | – | – | – |
| Proposed | KSDRM | 97.94 | 97.53 | 97.48 | 97.98 |
| Proposed | KSDRM with K-fold | 99.89 | 99.88 | 99.89 | 99.87 |

**Table 8**. Comparative analysis with existing models.

performance in comparison to the proposed stacking model. Experimental results reveal that RF shows the best performance with a 97.34% accuracy, followed by the DT model which obtains a 92.45% accuracy. However, the proposed stacking model achieves the highest accuracy of 97.94% and outperforms all employed models. Its specificity and sensitivity are also high showing its capability to identify true positive and true negative instances. Performance comparison with existing state-of-the-art models further validates its performance.

## Data availability

The dataset can be requested from the corresponding authors.

Received: 27 September 2024; Accepted: 9 May 2025
Published online: 01 July 2025

## References

1. Sarwar, A. et al. Iot networks attacks detection using multi-novel features and extra tree random-voting ensemble classifier (ervec). *J. Ambient. Intell. Humaniz. Comput.* **14**(12), 16637–16651 (2023).
2. Ali, M., Shahroz, M., Mushtaq, M.F., Alfarhood, S., Safran, M. & Ashraf, I. Hybrid machine learning model for efficient botnet attack detection in iot environment. IEEE Access (2024).
3. Rustam, F. et al. Denial of service attack classification using machine learning with multi-features. *Electronics* **11**(22), 3817 (2022).
4. Ali, M., Mushtaq, M.F., Shahroz, M., Majeed, R., Samad, A. & Akram, U. Elderly fall activity detection using supervised machine learning models. In: International Conference on Soft Computing and Data Mining. Springer, pp. 331–340 (2022)
5. Kumar, S. Botnet detection techniques and research challenges. In: 2019 International Conference on Recent Advances in Energy-efficient Computing and Communication (ICRAECC). IEEE, pp. 1–6 (2019)
6. Al Shorman, A., Faris, H. & Aljarah, I. Unsupervised intelligent system based on one class support vector machine and grey wolf optimization for Iot botnet detection. *J. Ambient Intell. Humanized Comput.* **11**, 2809–2825 (2020).
7. Alothman, Z., Alkasassbeh, M. & Al-Haj Baddar, S. An efficient approach to detect Iot botnet attacks using machine learning. *J. High Speed Netw.* **26**(3), 241–254 (2020).
8. Ali, M., Mushtaq, M.F., Akram, U., Ramzan, S., Tahir, S., & Ahsan, M. An ensemble approach for firewall log classification using stacked machine learning models. J. Comput. & Biomed. Inform. **8**(02) (2025).
9. Apostol, I., Preda, M., Nila, C. & Bica, I. Iot botnet anomaly detection using unsupervised deep learning. *Electronics* **10**(16), 1876 (2021).
10. Akram, U. et al. Iottps: Ensemble rksvm model-based internet of things threat protection system. *Sensors* **23**(14), 6379 (2023).
11. Alshamkhany, M., Alshamkhany, W., Mansour, M., Khan, M., Dhou, S. & Aloul, F. Botnet attack detection using machine learning. In: 2020 14th International Conference on Innovations in Information Technology (IIT). IEEE, pp. 203–208 (2020)
12. Majeed, R., Abdullah, N. A., Faheem Mushtaq, M., Umer, M. & Nappi, M. Intelligent cyber-security system for Iot-aided drones using voting classifier. *Electronics* **10**(23), 2926 (2021).
13. Leevy, J.L., Hancock, J., Khoshgoftaar, T.M. & Peterson, J. Detecting information theft attacks in the bot-iot dataset. In: 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA). IEEE, pp. 807–812 (2021).
14. Pimentel, M. A., Clifton, D. A., Clifton, L. & Tarassenko, L. A review of novelty detection. *Signal Process.* **99**, 215–249 (2014).
15. Zhai, Y., Lu, B. & Li, X. Tpe-mha: A malicious traffic detection model based on time position encoding and multi-head attention. in: *2021 IEEE 21st International Conference on Communication Technology (ICCT). IEEE*, pp. 143–151 (2021).

16. Shaukat, K. et al. Performance comparison and current challenges of using machine learning techniques in cybersecurity. *Energies* **13**(10), 2509 (2020).
17. Yavanoglu, O., Aydos, M.: A review on cyber security datasets for machine learning algorithms. in: *2017 IEEE International Conference on Big Data (big Data). IEEE*, pp. 2186–2193 (2017)
18. Sudhakar, M. & Kaliyamurthie, K. Machine learning algorithms and approaches used in cybersecurity. in: *2022 IEEE 3rd Global Conference for Advancement in Technology (GCAT). IEEE*, pp. 1–5 (2022)
19. Alqahtani, H., Sarker, I.H., Kalim, A., Minhaz Hossain, S.M., Ikhlaq, S. & Hossain, S. Cyber intrusion detection using machine learning classification techniques. In: *Computing Science, Communication and Security: First International Conference, COMS2 2020, Gujarat, India, March 26–27, 2020, Revised Selected Papers 1*. Springer, pp. 121–131 (2020)
20. Sarker, I. H., Abushark, Y. B., Alsolami, F. & Khan, A. I. Intrudtree: A machine learning based cyber security intrusion detection model. *Symmetry* **12**(5), 754 (2020).
21. Agarwal, A., Sharma, P., Alshehri, M., Mohamed, A. A. & Alfarraj, O. Classification model for accuracy and intrusion detection using machine learning approach. *PeerJ Comput. Sci.* **7**, 437 (2021).
22. Sarker, I. H. et al. Cybersecurity data science: An overview from machine learning perspective. *J. Big Data* **7**, 1–29 (2020).
23. Kilincer, I. F., Ertam, F. & Sengur, A. Machine learning methods for cyber security intrusion detection: Datasets and comparative study. *Comput. Netw.* **188**, 107840 (2021).
24. Chen, F., Ye, Z., Wang, C., Yan, L. & Wang, R. A feature selection approach for network intrusion detection based on tree-seed algorithm and k-nearest neighbor. in: *2018 IEEE 4th International Symposium on Wireless Systems Within the International Conferences on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS-SWS). IEEE*, pp. 68–72 (2018)
25. Sabar, N. R., Yi, X. & Song, A. A bi-objective hyper-heuristic support vector machines for big data cyber-security. *IEEE Access* **6**, 10421–10431 (2018).
26. Nguyen, H.-T., Ngo, Q.-D., Nguyen, D.-H. & Le, V.-H. Psi-rooted subgraph: A novel feature for Iot botnet detection using classifier algorithms. *ICT Express* **6**(2), 128–138 (2020).
27. Raman, M. G., Somu, N., Kirthivasan, K., Liscano, R. & Sriram, V. S. An efficient intrusion detection system based on hypergraph-genetic algorithm for parameter optimization and feature selection in support vector machine. *Knowl.-Based Syst.* **134**, 1–12 (2017).
28. Narayan, O., Qazi, M. A. & Kannan, R. Benchmarking supervised learning frameworks for engineering highly scalable intelligent systems: Cs9223/cs6513 cse nyu tandon school of engineering. *Procedia Comput. Sci.* **140**, 216–222 (2018).
29. Junejo, K.N. & Goh, J. Behaviour-based attack detection and classification in cyber physical systems using machine learning. in: *Proceedings of the 2nd ACM International Workshop on Cyber-physical System Security*, pp. 34–43 (2016).
30. Moorthy, R. S. S. & Nathiya, N. Botnet detection using artificial intelligence. *Procedia Comput. Sci.* **218**, 1405–1413 (2023).
31. Motylinski, M., MacDermott, Á., Iqbal, F. & Shah, B. A GPU-based machine learning approach for detection of botnet attacks. *Comput. Security* **123**, 102918 (2022).
32. Hoang, X. D. & Nguyen, Q. C. Botnet detection based on machine learning techniques using DNS query data. *Fut. Internet* **10**(5), 43 (2018).
33. Soni, S. & Bhushan, B. Use of machine learning algorithms for designing efficient cyber security solutions. in: *2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT). IEEE*, vol. 1, pp. 1496–1501 (2019).
34. Gautam, S. K. & Om, H. Computational neural network regression model for host based intrusion detection system. *Perspect. Sci.* **8**, 93–95 (2016).
35. Meitei, I.L., Singh, K.J. & De, T. Detection of ddos dns amplification attack using classification algorithm. in: *Proceedings of the International Conference on Informatics and Analytics*, pp. 1–6 (2016).
36. Majeed, R., Abdullah, N. A. & Mushtaq, M. F. Iot-based cyber-security of drones using the naïve Bayes algorithm. *Int. J. Adv. Comput. Sci. Appl.* **12**(7), 1–6 (2021).
37. Taheri, R., Shojafar, M., Arabikhan, F. & Gegov, A. Unveiling vulnerabilities in deep learning-based malware detection: Differential privacy driven adversarial attacks. *Comput. Secur.* **146**, 104035 (2024).
38. Ghahramani, M., Taheri, R., Shojafar, M., Javidan, R. & Wan, S. Deep image: A precious image based deep learning method for online malware detection in Iot environment. *Internet Things* **27**, 101300 (2024).
39. Dahiya, P. & Srivastava, D. K. Network intrusion detection in big dataset using spark. *Procedia Comput. Sci.* **132**, 253–262 (2018).
40. Khammassi, C. & Krichen, S. A GA-LR wrapper approach for feature selection in network intrusion detection. *Comput. Secur.* **70**, 255–277 (2017).
41. Kasongo, S. M. & Sun, Y. Performance analysis of intrusion detection systems using a feature selection method on the unsw-nb15 dataset. *J. Big Data* **7**, 1–20 (2020).
42. Lopez-Martin, M., Carro, B., Sanchez-Esguevillas, A. & Lloret, J. Shallow neural network with kernel approximation for prediction problems in highly demanding data networks. *Expert Syst. Appl.* **124**, 196–208 (2019).
43. Khan, N.M., Madhav C, N., Negi, A. & Thaseen, I.S. Analysis on improving the performance of machine learning models using feature selection technique. in: *Intelligent Systems Design and Applications: 18th International Conference on Intelligent Systems Design and Applications (ISDA 2018) Held in Vellore, India, December 6-8, 2018. Springer*, Volume 2, pp. 69–77 (2020)
44. Vinayakumar, R. et al. Deep learning approach for intelligent intrusion detection system. *IEEE Access* **7**, 41525–41550 (2019).
45. Almomani, O., Almaiah, M.A., Alsaaidah, A., Smadi, S., Mohammad, A.H. & Althunibat, A. Machine learning classifiers for network intrusion detection system: comparative study. in: *2021 International Conference on Information Technology (ICIT). IEEE*, pp. 440–445 (2021)
46. Moustafa, N. & Slay, J. The evaluation of network anomaly detection systems: statistical analysis of the unsw-nb15 data set and the comparison with the kdd99 data set. *Inform. Secur. J.: A Global Perspect.* **25**(1–3), 18–31 (2016).
47. Disha, R.A. & Waheed, S. A comparative study of machine learning models for network intrusion detection system using unsw-nb 15 dataset. In: *2021 International Conference on Electronics, Communications and Information Technology (ICECIT). IEEE*, pp. 1–5 (2021)
48. Kilincer, I. F., Ertam, F. & Sengur, A. A comprehensive intrusion detection framework using boosting algorithms. *Comput. Electr. Eng.* **100**, 107869 (2022).
49. Suchetha, G. & Pushpalatha, K. Optimizing botnet detection in Iot networks: Feature selection analysis on the unsw-nb15 dataset. in: *2024 IEEE International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER). IEEE*, pp. 120–125 (2024).
50. Rao, K.S., Reddy, D.M.: Comprehensive intrusion detection for investigating network traffic and botnet attacks. in: *2024 IEEE 6th International Conference on Cybernetics, Cognition and Machine Learning Applications (ICCCMLA). IEEE*, pp. 339–344 (2024).
51. Antony, V. & Thangarasu, N. Deepsvm-a novel approach for early detection and classification of Iot botnet attacks. in: *2024 Second International Conference on Inventive Computing and Informatics (ICICI). IEEE*, pp. 152–158 (2024).
52. Ramesh, R.B. & Thangaraj, S.J.J. Analyzing and detecting botnet attacks using anomaly detection with machine learning. in: *2023 5th International Conference on Inventive Research in Computing Applications (ICIRCA). IEEE*, pp. 911–915 (2023)
53. Anoh, N.G., Kone, T., Adepo, J.C., Mmoh, J.F. & Babri, M. Iot intrusion detection system based on machine learning algorithms using the unsw-nb15 dataset (2024).
54. Agrawal, K., Chittineni, S., Reddy, P. D. P. & Subhadra, K. Intrusion detection for cyber security: A comparative study of machine learning, deep learning and transfer learning methods. *Int. J. Microsyst. IoT* **2**(1), 483–491 (2024).

55. Sadhwani, S., Ajit, N., Muthalagu, R. & Pawar, P.M. An efficient and smart botnet attack detection model for iiot 4.0. in: *2024 4th International Conference on Artificial Intelligence and Signal Processing (AISP). IEEE*, pp. 1–6 (2024).
56. Esha, H., Hadimani, B.S., Devika, S., Shanthala, P. & Bhavana, R. Iot botnet creation and detection using machine learning. In: *2023 International Conference on Advancement in Computation & Computer Technologies (InCACCT). IEEE*, pp. 123–128 (2023).
57. Saxena, P. & Patel, R. Efficient hybrid model for botnet detection using machine learning. In: *2023 4th International Conference on Computation, Automation and Knowledge Management (ICCAKM). IEEE*, pp. 1–7 (2023).
58. Samantaray, M., Barik, R. C. & Biswal, A. K. A comparative assessment of machine learning algorithms in the Iot-based network intrusion detection systems. *Dec. Anal. J.* **11**, 100478 (2024).

## Acknowledgements

The authors extend their gratitude to the Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R192), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

## Author contributions

MA conceived the idea, performed data analysis and wrote the original draft. MFM conceived the idea, performed data curation and wrote the original draft. UA performed data curation, formal analysis, and designed methodology. DGA performed formal analysis and visualization and designed methodology. MMV acquired the funding for research, and performed visualization and initial investigation. HK dealt with software, performed visualization and carried out project administration. IA supervised the study, performed validation and reviewed and edited the manuscript. All authors read and approved the final manuscript.

## Funding

This study was funded by the European University of Atlantic. This study was also funded by the Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R192), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

## Declarations

## Competing interests

The authors declare no competing interests.

## Additional information

**Correspondence** and requests for materials should be addressed to H.K. or I.A.