

Machine learning approach for intrusion detection system using dimensionality reduction

Deepa Manikandan, Jayaseelan Dhilipan

Department of Computer Science and Applications, Faculty of Science and Humanities, SRM Institute of Science and Technology, Ramapuram Campus, Chennai, India

Article Info

Article history:

Received Jul 15, 2023

Revised Nov 26, 2023

Accepted Nov 29, 2023

Keywords:

Database management

Dimensionality reduction

Feature selection

Intrusion detection system

Machine learning

ABSTRACT

As cyberspace has emerged, security in all the domains like networks, cloud, and databases has become a greater concern in real-time distributed systems. Existing systems for detecting intrusions (IDS) are having challenges coping with constantly changing threats. The proposed model, DR-DBMS (dimensionality reduction in database management systems), creates a unique strategy that combines supervised machine learning algorithms, dimensionality reduction approaches and advanced rule-based classifiers to improve intrusion detection accuracy in terms of different types of attacks. According to simulation results, the DR-DBMS system detected the intrusion attack in 0.07 seconds and with a smaller number of features using the dimensionality reduction and feature selection techniques efficiently.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Deepa Manikandan

Department of Computer Science and Applications

Faculty of Science and Humanities, SRM Institute of Science and Technology, Ramapuram Campus

Ramapuram, Chennai, Tamil Nadu, India

email: dm8027@srmist.edu.in

1. INTRODUCTION

The daily evolution of threats to computer and internet security has forced distributed systems to be embedded with effective network security challenges. A new set of detection based security solutions is being developed to accommodate and detect these fresh dangers [1], [2]. This system employs supervised machine learning (ML) to create a classifier that can recognize freshly created anomalies and attack variants rather than relying solely on established signatures. To handle the challenge of spotting tiny attacks, the system incorporates a priority resolver and a specially designed iterative algorithm. ML techniques are used to create a classifier that can discriminate between various sorts of attacks [3]. Rule-based classifiers are used to generate a set of conditional branching-based rules like IFTHEN language rules, which are then used by the classifier. Due to the rules that these rule-based classifiers produce, they are regarded as knowledge-based systems.

The area of research in focus revolves around cybersecurity, specifically intrusion detection systems (IDSs). In today's interconnected world, where organizations heavily rely on computer networks and data exchange, the threat of cyber-attacks has become increasingly prevalent. These detection methods play a crucial role in safeguarding these networks by detecting and preventing unauthorized access, malicious activities, and potential security breaches. The field of IDS involves the development and implementation of techniques and algorithms that can effectively identify and respond to various types of attacks on computer networks. These attacks can range from traditional methods like viruses, worms, and denial-of-service attacks [4]–[6] to more sophisticated threats like advanced persistent threats and zero-day exploits. Traditionally,

these security systems relied on rule-based approaches, which involved defining a set of predefined rules to detect known attack patterns. However, with the rapid advancements in technology and the increasing complexity of cyber-attacks, there is a need for more intelligent and adaptive methods. This has led to the integration of ML and deep learning techniques into IDSs [7]. By leveraging these advanced algorithms, IDSs can learn from past data, identify patterns, and make accurate predictions about potential attacks. This shift towards ML and deep learning has opened up new possibilities in terms of the accuracy, efficiency, and adaptability of IDSs [8], [9]. The overarching goal of research in this area is to develop robust and effective security systems that can proactively identify and mitigate threats in real-time [10]. By continuously analyzing network traffic and learning from new data, these systems can adapt to emerging attack techniques and provide a reliable defense against cyber threats [11]. Given the ever-evolving nature of cyber-attacks, research in this area is crucial to stay ahead of the curve and ensure the security of computer networks and sensitive data [12]. The integration of ML and deep learning techniques holds great promise in enhancing the capabilities of and bolstering cybersecurity in the face of evolving threats [13]-[15].

Research gap analysis: in the context of intrusion detection, there has been significant research on the application of machine learning and learning algorithms. Research gaps for intrusion attack detection using dimensionality reduction techniques, classifier models, and ML algorithms can be identified as follows:

- Evaluation of dimensionality reduction techniques: while dimensionality reduction techniques are commonly used to reduce the feature space and improve the efficiency of IDSs, there is a need for a comprehensive evaluation of different dimensionality reduction methods. Research can focus on comparing the performance, computational efficiency, and robustness of various techniques such as principal component analysis (PCA), linear discriminant analysis (LDA), and t-distributed stochastic neighbor embedding (t-SNE) in the context of intrusion attack detection.
- Optimization of classifier models: classifier models play a crucial role in accurately identifying and classifying intrusion attacks. However, there is still room for improvement in optimizing these models for intrusion detection. Research can explore techniques to enhance the performance of classifier models by fine-tuning hyperparameters, selecting appropriate feature subsets, and considering ensemble methods, to combine multiple classifiers.
- Investigation of hybrid approaches: hybrid approaches that combine dimensionality reduction techniques with classifier models have shown promising results in intrusion attack detection. However, there is a need for further research to explore the optimal combinations of dimensionality reduction techniques and classifier models. This could involve investigating the impact of different feature selection methods, feature extraction techniques, and classifier model architectures on the overall performance of IDSs.
- Handling imbalanced datasets: imbalanced datasets, where the number of normal instances outweighs the number of intrusion instances, pose a challenge for intrusion detection. Research can focus on developing techniques to effectively handle imbalanced datasets, such as oversampling, under-sampling, or hybrid sampling methods. This would help improve the performance of intrusion attack detection models by addressing the issue of class imbalance.
- Real-time intrusion attack detection: real-time detection of intrusion attacks is crucial for timely response and mitigation. However, most existing research focuses on offline analysis of network traffic data. Further research is needed to develop real-time IDSs that can efficiently process and analyze streaming data, leveraging dimensionality reduction techniques and classifier models.

Overall, conducting research in these identified gaps would contribute to the advancement of intrusion attack detection using dimensionality reduction techniques, classifier models, and ML algorithms. It would help improve the accuracy, efficiency, and real-time capabilities of IDSs, ultimately enhancing network security.

The intrusion detection industry has undergone significant development over the past few decades, making research a constant process. Much research employing classic ML techniques for database security has been proposed including the hidden markov model (HMM), and K-nearest neighbor (KNN). To display superior assessment metrics, Elbasiony *et al.* [6] devised an integrated ML approach that merged k-means and clustering algorithms. The authors used the KDD CUP99 dataset for the verification of the method, albeit with redundancy, which leads the classifier to be biased towards more often occurring records. The new version dataset of the KDD '99 dataset was used by Wang and Shen [5] to evaluate their approach in their paper, "An effective intrusion detection framework based on support vector machine with feature augmentation". They claimed that because their plan had an efficacy rate of 99.92%, it was better than other strategies. Support vector machine is a poor choice for analyzing large amounts of network traffic for intrusion detection since its performance degrades when too much data is involved. Organizations are assisted in identifying and preventing unauthorized access by systems for detecting intrusions. In their paper "Random forest (RF) modelling for Network IDS," Farnaaz and Jabbar [9] constructed an intrusion detection model on a new version dataset of the KDD '99 dataset, tested its performance, and discovered that it had a 99.67% higher detection rate than J48. In their paper "Intrusion detection based on a novel hybrid learning

approach," Khalvati *et al.* [16] detected internet of things (IoT) intrusions using a support vector machine and Bayesian by adopting the KDD CUP 99 dataset with an accuracy of 91.50%. The most significant weakness of the RF method is the possibility of real-time forecast unresponsiveness caused by an excessive number of trees [16], [17].

Kabla *et al.* [18], worked on a design that was based on a modified Bayesian model and derived for the test statistic. Their system outperformed the other available intrusion attack techniques. Paricherla *et al.* [19] presented a two-dimensionality reduction approach (linear discriminant analysis), to address concerns with extra dimensions in datasets. Ali and Jawhar [20], in their paper titled "Detecting network attacks model based on a convolution neural network", have discussed convolutional neural network (CNN) networks using deep learning convolutional networks. Ramasamy and Eric [21], in their paper "A tree growth based forward feature selection algorithm for IDS on convolution neural network", developed a data-driven intelligence model that incorporated a fog-enabled artificial intelligence (AI) engine and threat intelligence and provided optimal training models for threat detection and efficacy in real-time cyber-attacks [21]–[24].

2. THE PROPOSED METHOD

The proposed system dimensionality reduction in database management system (DR-DBMS) combines classifiers, priority resolvers, supervised ML algorithms, and specifically created iterative methods to improve the accuracy of detecting seldom detectable assaults and can handle imbalanced datasets in an efficient manner. Since traditional IDSs are becoming less effective against emerging threats in cyberspace, the aim is to improve the identification of rarely detectable intrusions. The DR-DBMS system combines ML, rule-based classifiers, and cutting-edge methodologies to increase the precision and efficacy of intrusion detection, particularly against newly generated irregularities and attack variations that cannot be effectively detected using conventional signature-based IDS approaches. This methodology included a number of steps, such as dimensionality reduction, feature selection, and evaluation metrics for the dataset. The performance and effectiveness of the model as a whole are affected by each stage. The suggested model seeks to enhance intrusion detection and adapt to the changing environment of security threats by utilizing these strategies. In order to detect sophisticated attackers and enhance intrusion detection, ML methods are used in this work. Computers can learn and recognize patterns in data by ML, which makes them useful for jobs like spotting malware in encrypted transmission and identifying insider hazards. Techniques for dimensionality reduction and feature selection are used to improve the model's performance and effectiveness. The aim of the dimensionality reduction (DR) method is to reduce information loss while minimizing the features in the dataset. This aids in simplifying the model and enhancing its functionality. Additionally, limiting the number of dimensions in the data can make it simpler to visualize and analyze. By choosing the most pertinent and instructive features for training, feature selection further improves the model's performance. The model can focus on the most crucial aspects of the data by choosing the optimum set of features, improving accuracy and training efficiency.

The system develops a classifier to distinguish between various attacks using ML techniques. For the system to function effectively in real-world circumstances, it must have the ability to recognize modern threats fast. The number of features in the proposed system is reduced from 87 to 67 using dimensionality reduction, which is combined with feature selection to speed up training and get rid of extraneous data. This improves the model's performance and makes it possible to visualize data more easily. The effectiveness of the training process is improved by choosing the most pertinent features and dimensionality-reducing the model's complexity. The importance of pre-processing the data is emphasized because it is closely related to how well the trained model performs in any machine-learning activity.

Dimensionality reduction is used after pre-processing to lower the number of features while keeping the most data. As a result, the model is less complex and performs better. Then, feature selection is used to determine which elements are most crucial, enhancing the effectiveness of the model-building process. Together, these steps improve the system's ability to handle security-related activities.

Data pre-processing entails normalizing, encoding, and cleaning the data before it is used. The pre-processed dataset is divided into two sets: a training set and a validation or testing set. This process can lessen computational complexity and enhance model performance. The method of choosing the features with the greatest relevance and an important influence on the target variable is used to further refine the feature dataset. Recursive feature elimination (RFE), correlation analysis, or feature importance ranking.

Priority resolver: in this stage, the selected features are given weights or importance scores to indicate their relative significance in the finished model. Final training model: the model is trained using pre-processed data, reduced dimensions, and selected features. Grid search or random search approaches are used in conjunction with different ML algorithms like linear regression, and decision tree algorithms to find the best combination. Figure 1 explains the architecture of the DR DBMS model.

The DR-DBMS model has three phases. The Phase 1 contains dataset collection and pre-processing. Phase 2 explains a dimensionality reduction prediction classifier and feature selection. Phase 3 describes performance evaluation to improve the system performance.

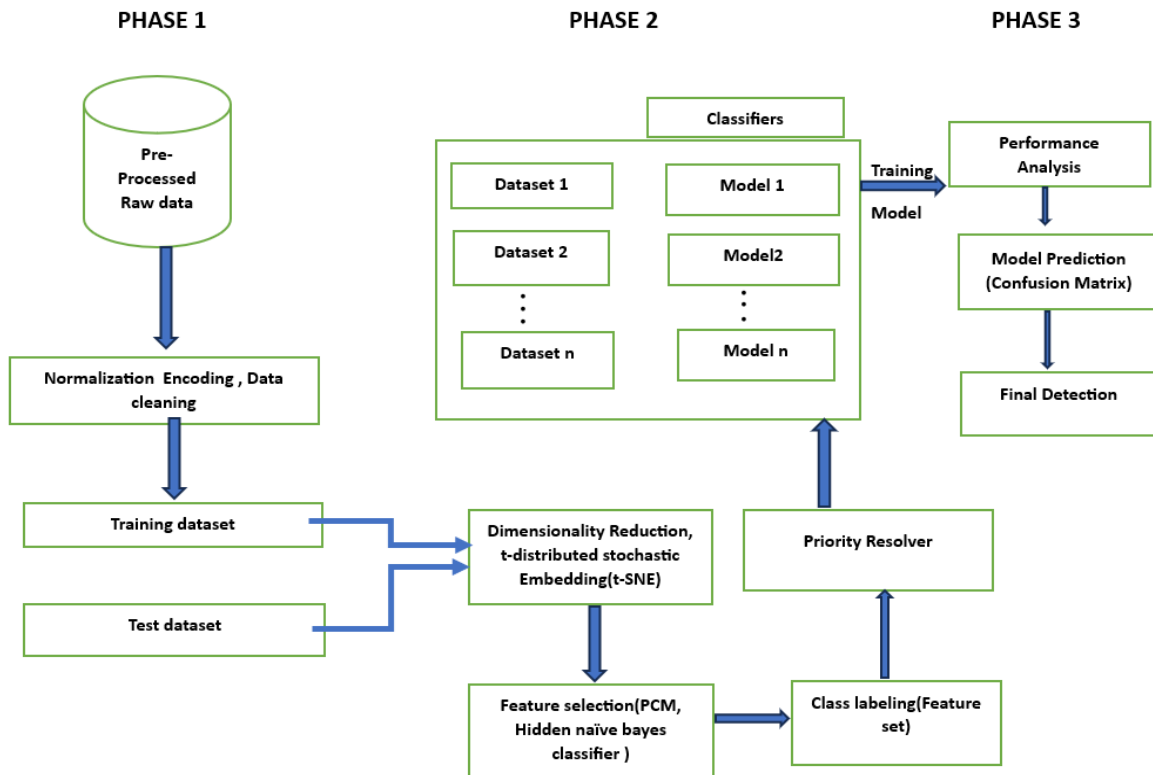


Figure 1. The architecture of the DR DBMS Model

3. METHOD

3.1. Phase 1: dataset collection and preprocessing

The selection of datasets for experimental purposes is a critical effort as the efficiency of the system is dependent on the correctness of a dataset. It can be claimed that the system's effectiveness rises in direct proportion to the accuracy of the data. The suggested approach for intrusion detection is validated using the distributed denial of service (DDoS) [24]–[27], CIC-DDoS2019 dataset (87 features) in order to get over these challenges as is represented in Figures 2 and 3. In this phase, all 87 features of the dataset is collected and preprocessed for checking the validation of data.

```

    Avg Fwd Segment Size
    Avg Bwd Segment Size
    Fwd Header Length.1
    Bwd Avg Bulk Rate
    Subflow Fwd Packets
    Subflow Fwd Bytes
    Subflow Bwd Packets
    Subflow Bwd Bytes
    Init_win_bytes_forward
    Init_win_bytes_backward
    act_data_pkt_fwd
    min_seg_size_forward
    Active Mean
    Active Std
    Active Max
    Active Min
    Idle Mean
    Idle Std
    Idle Max
    Idle Min
    Label
    Test mode: split 80.0% train, remainder test
    
```

Figure 2. Feature set (1)

```

=== Run information ===
Scheme:      weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:    UDPLag_data_2_0_per-weka.filters.unsupervised.attribute.Remove-R22-23,38-41,51-58,64-68,86-87
Instances:   370605
Attributes:  67
            Unnamed: 0
            Flow ID
            Source IP
            Source Port
            Destination IP
            Destination Port
            Protocol
            Timestamp
            Flow Duration
            Total Fwd Packets
            Total Backward Packets
            Total Length of Fwd Packets
            Total Length of Bwd Packets
            Fwd Packet Length Max
            Fwd Packet Length Min
            Fwd Packet Length Mean
            Fwd Packet Length Std
            Bwd Packet Length Max
            Bwd Packet Length Min
            Bwd Packet Length Mean
            Bwd Packet Length Std
            Flow IAT Mean
            Flow IAT Std
            Flow IAT Max
            Flow IAT Min
            Fwd IAT Total
            Fwd IAT Mean
            Fwd IAT Std
            Fwd IAT Max
            Fwd IAT Min
            Bwd IAT Total
            Bwd IAT Mean
            Bwd IAT Std
            Bwd IAT Max
            Bwd IAT Min
            Fwd Header Length
            Bwd Header Length
            Fwd Packets/s
            Bwd Packets/s
            Min Packet Length
            Max Packet Length
            Packet Length Mean
            Packet Length Std
            Packet Length Variance
            Down/Up Ratio
            Average Packet Size

```

Figure 3. Feature set (2)

Sanitized datasets, simulated datasets, testbed datasets, and standard datasets are only a few of the many methods for gathering the data. The first three methods can be used, however, there are obstacles. While sanitized traffic is expensive, real traffic is risky. A simulation system's development is challenging and complex. Additionally, modeling diverse network attacks requires different types of traffic, which is difficult and expensive.

3.2. Phase 2: dimensionality reduction with Prediction classifier and feature selection

The final performance of ML algorithms can devalue the performance of training models with too many features in the training dataset as well as retain the maximum information as much as possible. As a result, complexity is decreased with increased performance and for visualization of data in various graphical formats. The notion that increasing features to a predictive modeling task makes it more challenging to model is known as the "curse of dimensionality". For data visualization, statistics supporting dimension reduction techniques are preferred. Achieving a minimum number of input variables in a dataset is referred to as dimensionality reduction. The drawback of dimensionality [25], is the idea of including extra features in a predictive modeling task which makes it harder to model. Still, these methods can be used in applied ML to simplify classification methods can be used in applied ML to simplify a classification or regression dataset so that a predictive model fits it better.

Various techniques are used for dimensionality reduction which includes PCA, singular value decomposition (SVD), and linear discriminant analysis (LDA). Each technique uses a different method to display the data in smaller dimensions with no information loss. In this reduction technique, the first step is to take all the 'n' variables of the dataset and train the model. Once the training is done performance of the model is validated. After this step, one by one feature is removed and the model is trained on (n-1) features for n number of times. After sufficient features are removed and checked that the model is still efficient with very minimal change or no change then the model is trained with the features thereby selecting only the optimal features with maximum efficiency and maximum performance. In feature selection, the relevant features are selected to build a model of maximum accuracy. The most common methods that are used for selecting the features of the dataset are:

- i) Correlation: this is the statistical technique that is used to determine how one variable determines the changes in relation to another variable this is given by an analysis called bi-variate analysis which

completely explains the degree of relationship between any two variables. This method helps in finding the important variables on which other variables depend. It plays a vital role in various modelling techniques, for a better understanding of the nature of data about their positive and their negative correlationship.

- ii) Chi-square test: this is a statistical test for the data represented by χ^2 and is used to test if the frequency distribution of the variable represented categorically is different from the expected output and to test for the relationship between two different variables.
- iii) ANOVA: it helps in selecting the best features to minimize the input variables so that the complexity of a single-factor model.
- iv) Information entropy (IE): it can be defined as a measure of the information a feature can provide about a class. It determines the order of attributes in a training model. This is represented by:

$$IE = -\sum_{i=1}^c P_i \log_2 P_i \quad (1)$$

where p_i is the probability of randomly picking an element of class i . When building a ML model for a real-world dataset, a lot of features are found in the dataset. Not all of these features are always required. Including redundant features during training stages makes the model less accurate overall, more complicated, less able to generalize, and more biased.

3.3. Phase 3: priority resolver and training developer

In this phase in order to increase the task scheduling reaction time, Algorithm 1 divides the priority instances into various priority queues. For training the model, the algorithm takes in 87 features as input (CIC-DDoS2019 dataset), and is given to the procedure `data_processing` selection where priority resolver is used as the classifier and dimensionality reduction method [26], [27] is applied to reduce features from 87 to 67 maintaining the efficiency and giving the required accuracy and other evaluation metrics. Algorithm: the model is designed to perform the following actions: input data, perform feature selection based on priority resolver, train the model, and classification of label data.

Algorithm 1. Proposed system for dimensionality reduction

```

Step 1. Import the necessary modules:
- `from sklearn.feature_selection import SelectKBest`
- `from sklearn.feature_selection import mutual_info_classif`
- `from sklearn.naive_bayes import MultinomialNB`
Step 2. Read the training data and define the variables `X` and `y`:
- `X = train_data.iloc[:, :-1]`
- `y = train_data.iloc[:, -1]`
Step 3. Perform feature selection using SelectKBest:
- Initialize an instance of SelectKBest with the desired number of features: `skb = SelectKBest(k=1)`
- Fit the SelectKBest instance on the training data: `skb.fit(X, y)`
- Transform the training data to retain only the selected features: `X_new = skb.transform(X)`
Step 4. Perform feature selection using mutual information:
- Initialize an instance of mutual_info_classif: `mic = mutual_info_classif(X, y)`
- Fit the mutual_info_classif instance on the training data: `mic.fit(X, y)`
- Transform the training data to retain only the selected features: `X_new = mic.transform(X)`
Step 5. Train a Multinomial Naive Bayes classifier:
- Initialize an instance of MultinomialNB: `nb = MultinomialNB()`
- Fit the MultinomialNB instance on the selected features: `nb.fit(X_new, y)`
procedure data_processing (selection)
Input (CIC-DDoS2019 dataset) (87 features)
Feature selection with classifier Naïve Bayes Multinomial Updateable feature selection (Input)
return  $t_1$  to  $t_{24}$ ,  $t_{37}$ ,  $t_{42}$  to  $t_{50}$ ,  $t_{59}$  to  $t_{63}$ ,  $t_{69}$  to  $t_{85}$ ;
procedure priority_resolver classifies [( $t_1$  to  $t_{24}$ ,  $t_{37}$ ,  $t_{42}$  to  $t_{50}$ ,  $t_{59}$  to  $t_{63}$ ,  $t_{69}$  to  $t_{85}$ ), trueclass1];
Model=training [( $t_1$  to  $t_{24}$ ,  $t_{37}$ ,  $t_{42}$  to  $t_{50}$ ,  $t_{59}$  to  $t_{63}$ ,  $t_{69}$  to  $t_{85}$ ), trueclass1];
Predicted class=testing [Model ( $v_1$  to  $v_{24}$ ,  $v_{24}$  to  $v_{37}$ ,  $v_{42}$  to  $v_{50}$ ,  $v_{59}$  to  $v_{63}$ ,  $v_{69}$  to  $v_{85}$ )];
Accuracy  $\frac{1}{4}$  Confusion mat (Predicted Class trueclass1);
return Timewise, Accuracy, Labeled data.

```

This algorithm reads in train and test data and defines the features and target variable. It then performs mutual information-based feature selection using select best with a k -value of 10. It trains a multinomial Naive Bayes classifier on the selected features and predicts the test data. Finally, the code evaluates the performance of the classifier using accuracy, precision, recall, and F1 score.

4. RESULTS AND DISCUSSION

The developed model of the DR DBMS model has been tested on the CIC-DDoS2019 dataset. Around 370,606 datasets were tested 70% of the dataset were implemented for training the DR-DBMS model and 30% for testing the model. The model was simulated using Python programming language embedded with Tensor Flow as a basic library module. The performance of the DR DBMS algorithm is compared with various algorithms and the statistical results proved that usage of this algorithm outperforms other algorithms in terms of accuracy.

4.1. Performance evaluation

The performance of the suggested approach has been measured and evaluated using a number of indicators and criteria. The evaluation metrics implemented are accuracy, precision, recall, F-measure, Mathew correlation coefficient (MCC) area, and receiver operating characteristic (ROC) area. Accuracy: accuracy is computed as:

$$Accuracy = \frac{TP+TN}{P+N} \quad (2)$$

where P represents the total number of positive, N indicates the total number of negative, TP denotes the total number of true positive and TN describe true negative. Precision and recall are key performance factors in ML for pattern identification and classification.

$$Precision = \frac{TP}{TP+FP} \quad (3)$$

Recall indicates the true column of the actual or real values is used to compute the model and F-measure, which can be modeled as the harmonic mean of recall and precision. Recall and F-measure computation is shown in (3) and (4).

$$Recall = \frac{TP}{TP+FN} \quad (4)$$

$$F - Measure = 2 * \frac{Precision*Recall}{Precision+Recall} \quad (5)$$

4.2. Simulation setup

For computing, WEKA software is used for three phases of the proposed system. Table 1 demonstrates the simulation parameters of the proposed system. This table contains user datagram protocol (UDP) lag, Benign, and WebDoS.

Table 1. Simulation parameters

	UDP Lag	Benign	WebDDoS
No.of dataset	366,461	3,705	439
Kappa statistic	0.9545	0.9641	0.98556
Mean absolute error	0.0007	0.0006	0.00007
Root mean squared error	0.0258	-	0.0114
Relative absolute error	3.6997	-	-

4.2.1. Experimental results

The following are the results and observations recorded along during the training and testing of datasets with the proposed model. This mechanism detects the DoS attacks by applying proposed mechanism. The suggested algorithm uses the F1-score as the optimal criterion for choosing the best ML technique for each class, however, accuracy and recall rate are also employed to compare results. The recall rate demonstrates the model's dependability in identifying TP, whereas accuracy measures the proportion of correctly classified forecasts to all predictions. Figure 4 explains an Identified attacks in the datasets. Figure 5 describes the simulation results with evaluation metrics.

The performance of the system is also simulated for conditions in active and idle state of the system and the result is shown in Figure 6 which depicts variation for different combinations of featureset in the dataset. It explains the overall performance of 3 different feature sets using the DR-DBMS algorithm. Figure 6(a) explains feature set (1), and Figure 6(b) explains feature set (2). The performance of the system is defined by the effective reduction of attributes with limited time as shown in Figure 6. Here the instances taken for training the model is 370,605 with 87 features which undergoes dimensionality reduction and returns 67 features that can detect an intrusion attack in 0.07 seconds.

```

Destination IP = 54.68.141.132: UD (0.0)
Destination IP = 172.217.10.2: UD (0.0)
Destination IP = 172.217.7.2: UD (0.0)
Destination IP = 172.217.11.34: UD (0.0)
Destination IP = 23.194.140.15: UD (0.0)
Destination IP = 204.154.111.119: UD (0.0)
Destination IP = 216.58.219.206: UD (0.0)
Destination IP = 173.194.206.156: UD (0.0)
Destination IP = 54.192.49.13: UD (0.0)
Destination IP = 204.154.111.117: UD (0.0)
Destination IP = 54.192.49.19: UD (0.0)
Destination IP = 23.194.142.15: UD (0.0)
Destination IP = 172.217.10.99: UD (0.0)
Destination IP = 172.217.11.10: UD (0.0)
Destination IP = 172.217.7.14: UD (0.0)
Destination IP = 52.203.87.59: UD (0.0)
Destination IP = 23.194.141.17: UD (0.0)
Destination IP = 31.13.71.2: UD (0.0)
Destination IP = 52.7.108.194: UD (0.0)
Destination IP = 125.56.201.115: UD (0.0)
Destination IP = 52.36.47.72: UD (0.0)
Destination IP = 52.41.60.30: UD (0.0)
Destination IP = 192.0.73.2: UD (0.0)
Destination IP = 38.69.238.19: UD (0.0)
Destination IP = 224.0.0.251: BN (1.0)
Destination IP = 172.217.11.35: UD (0.0)
Destination IP = 172.217.7.1: UD (0.0)
Destination IP = 52.1.164.212: UD (0.0)
Destination IP = 204.154.111.131: UD (0.0)
Destination IP = 173.194.66.154: UD (0.0)
Destination IP = 54.192.49.188: UD (0.0)
Destination IP = 204.154.111.106: UD (0.0)
Destination IP = 54.192.49.191: BN (2.0)
Destination IP = 204.154.111.112: UD (0.0)
Destination IP = 172.217.10.4: UD (0.0)
Destination IP = 34.205.244.84: UD (0.0)
Destination IP = 35.173.44.140: UD (0.0)
Destination IP = 209.85.201.108: BN (1.0)
Destination IP = 52.40.109.206: UD (0.0)
Destination IP = 52.38.9.173: UD (0.0)
Destination IP = 172.217.6.194: UD (0.0)
Destination IP = 38.69.238.16: UD (0.0)
Destination IP = 40.124.45.19: UD (0.0)
Destination IP = 204.154.111.134: UD (0.0)
Destination IP = 52.34.248.21: UD (0.0)
Destination IP = 52.35.21.241: UD (0.0)
Destination IP = 172.217.3.98: BN (1.0)
Destination IP = 208.185.50.55: UD (0.0)
Destination IP = 208.185.50.90: UD (0.0)
Destination IP = 52.114.128.8: UD (0.0)
Destination IP = 52.216.164.85: UD (0.0)
    
```

Figure 4. Identified attacks in the datasets

```

=== Evaluation on training set ===

Time taken to test model on training data: 6.79 seconds

=== Summary ===

Correctly Classified Instances      338377      91.3039 %
Incorrectly Classified Instances    32228      8.6961 %
Kappa statistic                    0.152
Mean absolute error                0.0635
Root mean squared error            0.2406
Relative absolute error            430.46 %
Root relative squared error        280.1448 %
Total Number of Instances         370605

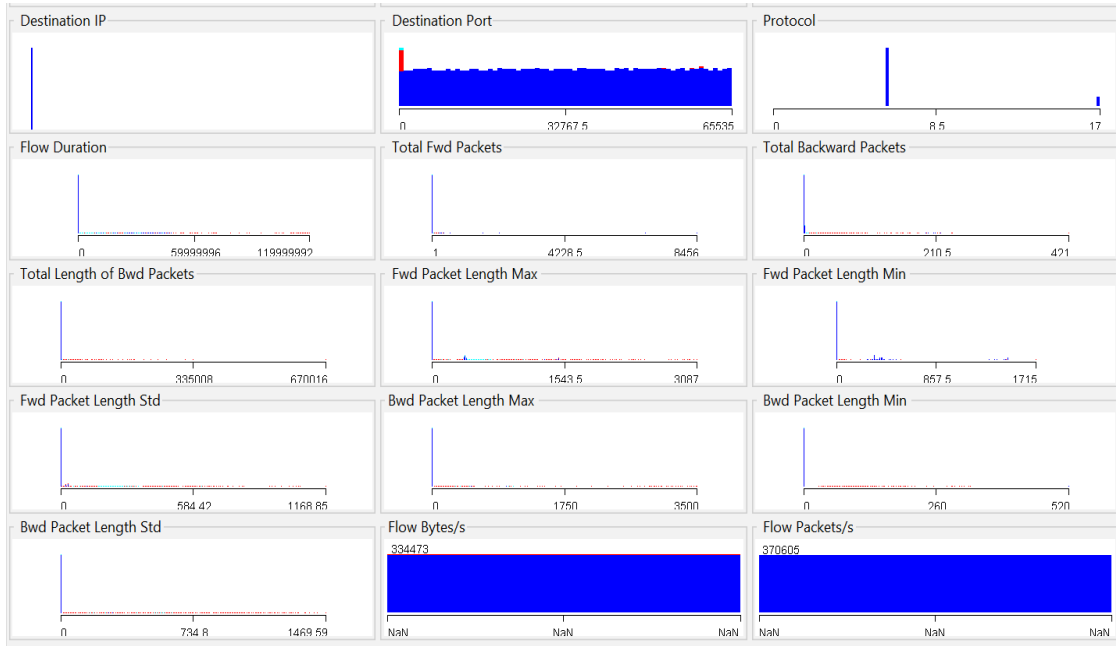
=== Detailed Accuracy By Class ===

      TP Rate  FP Rate  Precision  Recall  F-Measure  MCC      ROC Area  PRC Area  Class
      0.916   0.102   0.999     0.916   0.956     0.296   0.957    0.999    UD
      0.566   0.009   0.398     0.566   0.468     0.469   0.990    0.387    BN
      0.973   0.077   0.015     0.973   0.029     0.115   0.982    0.192    WD
Weighted Avg.  0.913   0.101   0.992     0.913   0.950     0.297   0.958    0.992

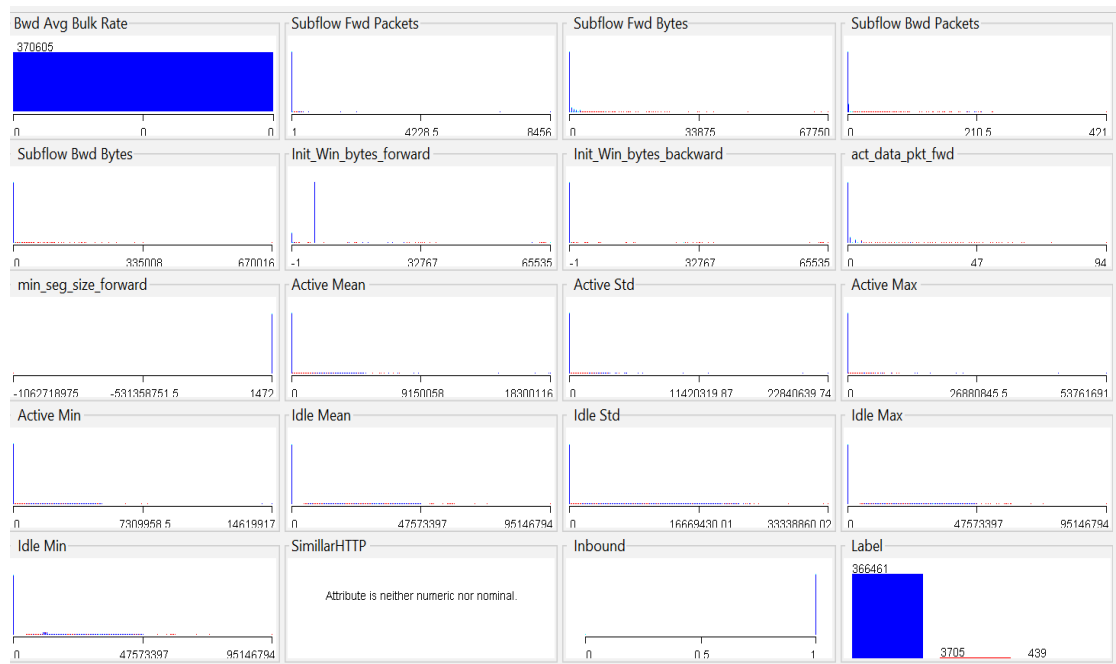
=== Confusion Matrix ===

      a      b      c  <-- classified as
335854  3157  27450 |      a = UD
   417  2096  1192 |      b = BN
     4     8   427 |      c = WD
    
```

Figure 5. Simulation results with evaluation metrics



(a)



(b)

Figure 6. Overall performance of 2 different feature sets using the DR-DBMS algorithm (a) feature set (1) and (b) feature set (2)

4.2.2. Performance analysis of different ML algorithms

Table 2 shows the comparative statistics of various ML algorithms in terms of measure, precision, recall, accuracy, and time (taken to build the model). Figure 7 explains the performance analysis of various ML algorithms based on F-measure and precision and Figure 8 demonstrates the comparative analysis based on recall, accuracy and time. In Figure 7 and Figure 8 various algorithms are compared for their F-measure, precision, recall, accuracy, and time taken to detect the attacks, the proposed DR-DBMS model exhibits maximum F-measure, accuracy, and precision thereby proving its efficiency in detecting the attacks. Table 2 demonstrates the computation of recall, accuracy, and time taken for various ML algorithms.

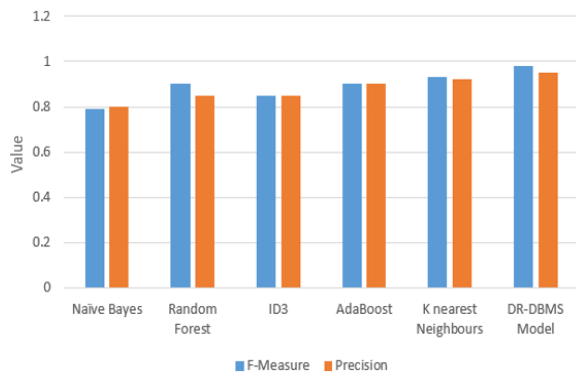


Figure 7. Performance analysis of various machines learning algorithms based on F-measure and precision

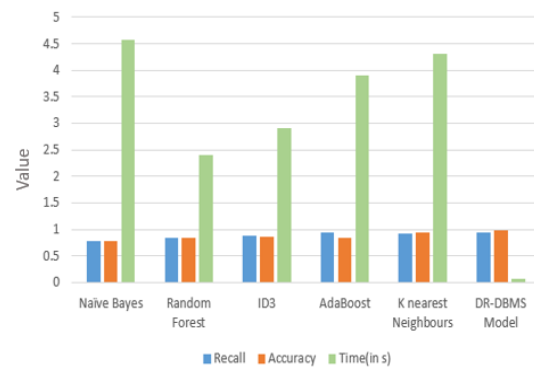


Figure 8. Comparative analysis based on recall, accuracy, and time

Table 2. Computation of recall, accuracy, time taken for various ML algorithms

Name of the algorithm	Recall	Accuracy	Time (in sec)
Naive Bayes	0.78	0.86	4.576
Random forest	0.83	0.87	24.739
ID3	0.87	0.86	29.284
AdaBoost	0.95	0.84	391.804
K-nearest neighbors	0.93	0.84	431.304
DR-DBMS model	0.95	0.98	0.076

5. CONCLUSION

To address intrusion detection in database management systems, the proposed DR-DBMS model is designed to support real-time datasets. A dimensionality reduction and priority classifier are embedded as a best-effort approach to develop a model to obtain high precision, and accuracy with net change in redundant instances. This mechanism also detects the DoS efficiently. The proposed system proves to be a proactive technique. Extensive simulations and experiments were employed to test, fine-tune, and validate the algorithm using the WEKA tool. According to simulation findings, the proposed system outperforms the existing algorithms by detecting the intrusion attack in 0.07 seconds and with a smaller number of features using the dimensionality reduction and feature selection techniques efficiently. The future scope is to use different combinations of hybrid deep learning algorithms for detecting intrusion attacks in cloud networking and Parallel computing.




REFERENCES

- [1] W. Zhong and F. Gu, "A multi-level deep learning system for malware detection," *Expert Systems with Applications*, vol. 133, pp. 151–162, Nov. 2019, doi: 10.1016/j.eswa.2019.04.064.
- [2] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, p. 20, Dec. 2019, doi: 10.1186/s42400-019-0038-7.
- [3] A. Abusitta, M. Bellaiche, M. Dagenais, and T. Halabi, "A deep learning approach for proactive multi-cloud cooperative intrusion detection system," *Future Generation Computer Systems*, vol. 98, pp. 308–318, Sep. 2019, doi: 10.1016/j.future.2019.03.043.
- [4] T. Aldwairi, D. Perera, and M. A. Novotny, "An evaluation of the performance of restricted boltzmann machines as a model for anomaly network intrusion detection," *Computer Networks*, vol. 144, pp. 111–119, Oct. 2018, doi: 10.1016/j.comnet.2018.07.025.
- [5] L. Wang and J. Shen, "Data-intensive service provision based on particle swarm optimization," *International Journal of Computational Intelligence Systems*, vol. 11, no. 1, pp. 330–339, 2018, doi: 10.2991/ijcis.11.1.25.
- [6] R. M. Elbasiony, E. A. Sallam, T. E. Eltobely, and M. M. Fahmy, "A hybrid network intrusion detection framework based on random forests and weighted k-means," *Ain Shams Engineering Journal*, vol. 4, no. 4, pp. 753–762, Dec. 2013, doi: 10.1016/j.asej.2013.01.003.
- [7] A. Liu and B. Sun, "An intrusion detection system based on a quantitative model of interaction mode between ports," *IEEE Access*, vol. 7, pp. 161725–161740, 2019, doi: 10.1109/ACCESS.2019.2951839.
- [8] A. M. Zarca, J. B. Bernabe, A. Skarmeta, and J. M. A. Calero, "Virtual IoT HoneyNets to mitigate cyberattacks in SDN/NFV-enabled IoT networks," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 6, pp. 1262–1277, Jun. 2020, doi: 10.1109/JSAC.2020.2986621.
- [9] N. Farnaaz and M. A. Jabbar, "Random forest modeling for network intrusion detection system," *Procedia Computer Science*, vol. 89, pp. 213–217, 2016, doi: 10.1016/j.procs.2016.06.047.
- [10] H. Sadreazami, A. Mohammadi, A. Asif, and K. N. Plataniotis, "Distributed-graph-based statistical approach for intrusion detection in cyber-physical systems," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 4, no. 1, pp. 137–147, Mar. 2018, doi: 10.1109/TSIPN.2017.2749976.




- [11] E. Viegas, A. Santin, A. Bessani, and N. Neves, "BigFlow: real-time and reliable anomaly-based intrusion detection for high-speed networks," *Future Generation Computer Systems*, vol. 93, pp. 473–485, Apr. 2019, doi: 10.1016/j.future.2018.09.051.
- [12] H. H. Pajouh, R. Javidan, R. Khayami, A. Dehghantanha, and K.-K. R. Choo, "A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in IoT backbone networks," *IEEE Transactions on Emerging Topics in Computing*, vol. 7, no. 2, pp. 314–323, Apr. 2019, doi: 10.1109/TETC.2016.2633228.
- [13] M. H. Ali, B. A. D. Al Mohammed, A. Ismail, and M. F. Zolkipli, "A new intrusion detection system based on fast learning network and particle swarm optimization," *IEEE Access*, vol. 6, pp. 20255–20261, 2018, doi: 10.1109/ACCESS.2018.2820092.
- [14] V. Zhou, C. Leckie and S. Karunasekera, "A Survey of Coordinated Attacks and Collaborative Intrusion Detection," *Computers & Security*, vol. 29, no. 1, pp. 124–40, 2010, doi: <https://doi.org/10.1016/j.cose.2009.06.008>.
- [15] S. Xu, Y. Qian, and R. Q. Hu, "Data-driven network intelligence for anomaly detection," *IEEE Network*, vol. 33, no. 3, pp. 88–95, May 2019, doi: 10.1109/MNET.2019.1800358.
- [16] L. Khalvati, M. Keshtgary, and N. Rikhtegar, "Intrusion detection based on a novel hybrid learning approach," *Journal of AI and Data Mining*, vol. 6, no. 1, pp. 157–162, 2018, doi: 10.22044/jadm.2017.979.
- [17] A. Verma and V. Ranga, "Evaluation of network intrusion detection systems for RPL based 6LoWPAN networks in IoT," *Wireless Personal Communications*, vol. 108, no. 3, pp. 1571–1594, Oct. 2019, doi: 10.1007/s11277-019-06485-w.
- [18] A. H. H. Kabla *et al.*, "Machine and deep learning techniques for detecting internet protocol version six attacks: a review," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 13, no. 5, pp. 5617–5631, Oct. 2023, doi: 10.11591/ijece.v13i5.pp5617-5631.
- [19] M. Paricherla, M. Ritonga, S. R. Shinde, S. M. Chaudhari, R. Linur, and A. Raghuvanshi, "Machine learning techniques for accurate classification and detection of intrusions in computer network," *Bulletin of Electrical Engineering and Informatics*, vol. 12, no. 4, pp. 2340–2347, Aug. 2023, doi: 10.11591/eei.v12i4.4708.
- [20] T. A. J. Ali and M. M. T. Jawhar, "Detecting network attacks model based on a convolutional neural network," *International Journal of Electrical and Computer Engineering*, vol. 13, no. 3, pp. 3072–3078, 2023, doi: 10.11591/ijece.v13i3.pp3072-3078.
- [21] M. Ramasamy and P. V. Eric, "A tree growth based forward feature selection algorithm for intrusion detection system on convolutional neural network," *Bulletin of Electrical Engineering and Informatics*, vol. 12, no. 1, pp. 472–482, 2023, doi: 10.11591/eei.v12i1.4015.
- [22] M. Rajmohan, C. Srinivasan, O. R. Babu, S. Murugan, and B. S. Kumar Reddy, "Efficient Indian sign language interpreter for hearing impaired," in *2023 Second International Conference On Smart Technologies For Smart Nation (SmartTechCon)*, Aug. 2023, pp. 914–917, doi: 10.1109/SmartTechCon57526.2023.10391782.
- [23] M. Roopa and S. Selvakumar Raja, "Intelligent intrusion detection and prevention system using smart multi-instance multi-label learning protocol for tactical mobile adhoc networks," *KSII Transactions on Internet and Information Systems*, vol. 12, no. 6, pp. 2895–2921, 2018, doi: 10.3837/tiis.2018.06.025.
- [24] M. Roopa and S. S. Raja, "An intelligent network algorithm for enhanced security in a mobile ad hoc network," *International Journal of Networking and Virtual Organisations*, vol. 17, no. 2–3, pp. 126–136, 2017, doi: 10.1504/ijnvo.2017.085516.
- [25] Y. Ayachi, Y. Mellah, M. Saber, N. Rahmoun, I. Kerrakchou, and T. Bouchentouf, "A survey and analysis of intrusion detection models based on information security and object technology-cloud intrusion dataset," *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 11, no. 4, pp. 1607–1614, Dec. 2022, doi: 10.11591/ijai.v11.i4.pp1607-1614.
- [26] M. Ramasamy and P. V. Eric, "A novel classification and clustering algorithms for intrusion detection system on convolutional neural network," *Bulletin of Electrical Engineering and Informatics*, vol. 11, no. 5, pp. 2845–2855, 2022, doi: 10.11591/eei.v11i5.4145.
- [27] H. Kamel and M. Z. Abdullah, "Distributed denial of service attacks detection for software defined networks based on evolutionary decision tree model," *Bulletin of Electrical Engineering and Informatics*, vol. 11, no. 4, pp. 2322–2330, 2022, doi: 10.11591/eei.v11i4.3835.

BIOGRAPHIES OF AUTHORS



Deepa Manikandan    received PG degree in Masters of Computer Applications from Madras University, India, in 2006. She is currently pursuing her doctorate degree from SRM Institute of Science and Technology. Her research area includes database management systems security, network security, cloud security, data analysis, deep learning, and machine learning. She has published in various international conferences and journals. She can be contacted at email: dm8027@srmist.edu.in.



Dr. Jayaseelan Dhilipan    Professor and Head, Department of MCA, SRMIST, Ramapuram. He completed his M.Sc., MBA, and MPhil and completed his Doctorate in 2014. His area of interest is cloud computing, machine learning, e-commerce, and data analysis. He has published 35 research articles in reputed journals and conference proceedings. He can be contacted at email: hod.mca.rmp@srmist.edu.in.