SSRG International Journal of Electronics and Communication Engineering
ISSN: 2348-8549/ https://doi.org/10.14445/23488549/IJECE-V12I2P102

Volume 12 Issue 1, 12-26, February 2025
© 2025 Seventh Sense Research Group®

Original Article

Enhanced Intrusion Detection Systems via Feature Selection and Dimensionality Reduction Techniques

Ranjeeth Kumar Sundararajan[1], R. Rajakumar[2], N. Kurinjivendhan[3], Banumathi Subramanian[4]

[1,2]*Department of Computer Science and Engineering, Srinivasa Ramanujan Centre, SASTRA University, Tamilnadu, India.*
[3]*Department of Computer Science, SRM Institute of Science and Technology, Trichy, Tamilnadu, India.*
[4]*Department of Computer Applications, Bishop Heber College (Autonomous), Trichy, Tamilnadu, India.*

[2]*Corresponding Author : rajakumar@src.sastra.edu*

Received: 05 December 2024 Revised: 03 January 2025 Accepted: 03 February 2025 Published: 26 February 2025

Abstract *- Due to the rapid development of modern networks, nowadays Intrusion Detection Systems (IDS) play a significant role in network security by detecting malicious activities or anomalies. However, the large dimensionality of network traffic data can undermine IDS efficacy, increasing processing complexity and perhaps reducing accuracy. This study explores these challenges by proposing a hybrid approach integrating Feature Selection (FS) and Dimensionality Reduction (DR) techniques. On the benchmark datasets NSL-KDD and KDDcup99, we compare various FS methods, such as filter-based, embedded-based, and wrapper-based, by selecting subsets of features to optimize IDS performance. Additionally, we explore Dimensionality Reduction techniques (DR) such as Principal Component Analysis (PCA) and Singular Values Decomposition (SVD), reducing the number of the features while maintaining accuracies of 88.91% and 70.61% on the KDDcup99 and NSL KDD test datasets and also with minimal runtime of 0.0513 and 0.0127 seconds respectively. Our findings show that hybrid FS and DR techniques improve IDS accuracy while drastically reducing computational overhead and increasing the efficiency and reliability of intrusion detection systems in real-world applications.*

Keywords *- Intrusion Detection Systems (IDS), Anomaly detection, Machine Learning, Feature Selection, Dimensionality Reduction.*

1. Introduction

In recent years, the fast expansion of network-based attacks has highlighted the crucial importance of robust Intrusion Detection Systems (IDS) in protecting digital assets and maintaining network integrity. IDS are critical for detecting and mitigating malicious actions, anomalies, and potential security breaches in network traffic. However, the increasing volume and complexity of network data provide substantial hurdles to IDS effectiveness, such as increased computational needs and worse detection accuracy due to irrelevant or duplicate information. To address these issues, this study focuses on improving IDS performance using Feature Selection (FS) and Dimensionality Reduction (FE). The major goal is to simplify the feature space of IDS input data while maintaining or improving detection accuracy and reducing computational overhead.

We begin by comparing filter-based, embedded-based, and wrapper-based FS approaches on two commonly used benchmark datasets, NSL-KDD and KDDcup99. These strategies are used to pick subsets of features that are most important for detecting network intrusions. The filter-based approach uses correlation matrices to find strongly associated features, which reduces redundancy and improves computational efficiency. Embedded-based approaches use machine learning models like Decision Trees (DT), Random Forests (RF), and Extra Trees (ET) to evaluate feature relevance in terms of IDS detection accuracy and runtime performance. Furthermore, wrapper-based algorithms like Recursive Feature Elimination (RFE) are used to iteratively pick features that improve classification results. Furthermore, to reduce the computational cost of high dimensional data, we analyze dimensionality reduction techniques such as Principal Component Analysis (PCA) and Singular Values Decomposition (SVD). These techniques convert the original feature space into lower-dimensional representations while retaining critical information for intrusion detection. PCA and SVD strive to improve computing efficiency by reducing the number of dimensions while maintaining IDS accuracy.

2. Related Works

The works of Alhelli S et al. [1] concentrate on the effective procedure of Feature Selection and Dimensionality Reduction (FSDR). In this study, a subset of features was chosen based on the Empirical Cumulative Density Function (ECDF), which was then applied to the selected subset of features to determine the uncertainty value in Mutual Information (MI). Subsequently, the back-propagation neural network technique was employed to identify various DDoS attacks, and the Singular Value Decomposition (SVD) algorithm was refined to reduce the dimensions of the new space containing the selected MI features. Primary experiments are carried out in the MATLAB environment. The experiment's outcomes show that our approach can select the best feature combination from two datasets.

A range of Machine Learning (ML) techniques were studied by Ankit Thakkar et al. [2] with the specific goal of attack detection and categorization for intrusion detection systems. Not every feature that is taken out of network packets aids in the classification or identification of attacks. Therefore, the research aims to examine the effects of various feature selection procedures on IDS performance. Methods for selecting features identify relevant features and group them into subgroups. This paper employs Chi-Square, Information Gain (IG), and Recursive Feature Elimination (RFE) feature selection techniques using machine learning classifiers, including Support Vector Machine, Naïve Bayes, Decision Tree Classifier, Random Forest Classifier, k-nearest neighbors, Logistic Regression, and Artificial Neural Networks respectively. The methods are tested on the NSL-KDD dataset, and the outcomes are contrasted.

In order to decrease the dimensionality of the data, Arshid Ali et al. [3] use Correlation-Based Feature Selection (CFS) techniques and Naïve Bayes (NB) classifiers. The suggested Intrusion Detection System (IDS) identifies attacks by use of a Multilayer Perceptron (MLP) and an Instance-Based Learning algorithm (IBK). Out of 78 features, the newly introduced IDS had an accuracy of 99.87% and 99.82% for IBK, with only 5 and 3 features, respectively. Other measures, such as the Receiver Operating Curve (ROC), F-measure, accuracy, and recall, support IBK's superior main performance over MLP.

In order to lower the computational cost, Dittakavi et al. [4] developed an IDS based on dimensionality reduction. For the CSE-CIC-IDS2018 dataset, we employed Principal Component Analysis (PCA), Non-negative Matrix Factorization (NMF), and High Correlation Filter as dimensionality reduction methods. The dataset spans a variety of assault types and contains over 16 million instances. The final feature set was narrowed down to 12 essential features: metrics for large transfers in both forward and backward directions, metrics for packet counts and sizes in both directions, metrics for flows' inter-arrival times in both directions, metrics for TCP flags, metrics for header sizes, and metrics for flow duration and rate.

In order to create a novel hybrid dimensionality reduction technique for intrusion detection, Fadi Salo et al. [5] developed an ensemble classifier based on Support Vector Machines (SVM), Instance-Based Learning Algorithms (IBK), and Multilayer Perceptrons (MLP) combined with the methods of Information Gain (IG) and Principal Component Analysis (PCA). The performance of the IG-PCA-Ensemble approach was evaluated on three popular datasets: ISCX 2012, NSL-KDD, and Kyoto 2006+. Experimental results show that the hybrid dimensionality reduction strategy, which leverages the ensemble of base learners to contribute more important features, outperforms individual approaches with good accuracy and low false alarm rates.

M. Di Mauro et al. [6] presented comparison surveys in multiple ways: (i) we summarize the most reliable FS approaches in intrusion detection, such as Multi-Objective Evolutionary techniques; (ii) we evaluate more recent datasets (updated with respect to obsolete KDD 99) using a custom Python procedure; (iii) we evaluate various experimental analyses, such as feature correlation, time complexity, and performance. Network/security managers can benefit from our comparisons as they offer valuable guidance when it comes to resource consumption and performance trade-offs in the context of network intrusion detection.

In an effort to reduce the size of the dataset, Ghanshyam Prasad Dubey et al. [7] propose two techniques-Dense_FR and Sparse_FR-for generating the optimal feature subset using mutual information and Kendall's correlation coefficient. Mutual information is a key and often used statistic in feature selection. It makes an effort to reduce the amount of ambiguity by adding more attributes. Kendall's correlation coefficient is more exacting and dependable than Pearson's or Spearman's correlation coefficients. The names Dense_FR and Sparse_FR describe the number of features generated in the best feature subsets; the Sparse_FR strategy yields a more feature-rich optimal subset than the Dense_FR approach.

Numerous types of intrusion detection systems are offered by Gulab Sah et al.'s [8] investigations, along with machine learning reduction and classification techniques like K-nearest neighbors, Support vector machines, Random Forests, and Naive Bayes. The main objective of this work is to propose a method for determining whether or not the accuracy rate will rise when the selected characteristics are used in contrast to the accuracy rate when all features are used.

In order to detect network invasions, Majdi Maabreh et al. [9] employed one million random cases from the CSE-CIC-IDS2018 big data set, four feature selection techniques, seven traditional machine learning algorithms, and a deep learning algorithm. The feature selection techniques demonstrated that the Two Flow Measurements (FLOW) and the Forwarding Direction (FWD) features were important in identifying the binary traffic type-attack or benign. Furthermore, by utilizing four unanimity features instead of all traffic features, training time can be lowered by 10% to 50%.

This paper presents an overview of feature selection and ensemble methodologies used in anomaly-based IDS research, as described in Majid Torabi et al.'s [10] studies. Following a review of dimensionality reduction techniques, the effectiveness of feature selection strategies for both training and detection is categorized. The process of identifying the most relevant features from data is essential to the development of anomaly-based intrusion detection systems since it has been demonstrated to increase detection efficiency in terms of both accuracy and processing efficiency. Next, we analyze and discuss various IDS-based approaches employing various detection models (either ensemble-based or single-classifier-based) to illustrate the significance and efficacy of these machine learning techniques in the field of intrusion detection.

An enhanced Intrusion Detection System (IDS) based on a hybrid feature selection approach and a Deep Neural Network (DNN) based classifier was presented by M. Naveed et al. [11]. A hybrid feature selection model uses three techniques (PCA, chi-square, and ANOVA) to provide a subset of reduced and optimal features that can be used for classification. These methods are referred to as "the big three." Following NSL-KDD dataset training, the suggested model is evaluated. The suggested strategy successfully achieved 40% less input data, an average accuracy of 99.73%, a precision score of 99.75%, an F1 score of 99.72%, and an average training and testing time. Using the NSL-KDD dataset, Razan Abdulhammed et al. [12] evaluate the top 10 machine learning algorithms. The best machine learning algorithm is then determined by ranking these algorithms based on a number of criteria, such as accuracy, sensitivity, and specificity. It is clear from examining the top 4 algorithms that they take a long time to develop models. Feature selection is used to identify intrusions as fast as possible without compromising accuracy.

Two techniques for lowering feature dimensionality were developed by Razan Abdulhammed et al. [13] and are employed in this study: (i) Principle Component Analysis (PCA) and (ii) Auto-Encoder (AE), an example of deep learning. Following the acquisition of low-dimensional features from both techniques, further classifiers such as Random Forest (RF), Bayesian Network, Linear Discriminant Analysis (LDA), and Quadratic Discriminant Analysis (QDA) are built to generate an IDS. The experimental results with low-dimensional features show enhanced performance in terms of Detection Rate (DR), F-Measure, False Alarm Rate (FAR), and Accuracy in binary and multi-class classification. This study lowered the feature dimensions of the CICIDS2017 dataset from 81 to 10, all the while maintaining a high degree of accuracy in binary and multi-class classification, reaching 99.6%.

Ahmad Rami and others [14] In order to balance them, this study examined techniques for improving the detection of DoS anomalies and power reservation in WSNs. The CH_Rotations algorithm is a new clustering technique that improves anomaly detection effectiveness across a WSN's lifetime. Furthermore, the effect of feature selection techniques on WSN lifetimes was evaluated, as was the application of these techniques in the machine learning algorithms-based analysis of WSN node traffic. The evaluation findings showed that, in terms of average performance accuracy, the Water Cycle (WC) feature selection performed better than Particle Swarm Optimization (PSO), Simulated Annealing (SA), Harmony Search (HS), and Genetic Algorithm (GA) by 2%, 5%, 3%, and 3%, respectively.

Yadav Surendra and others [15] This paper proposes an IoT network-based threat mitigation solution based on the Recurrent Neural Network (RNN) algorithm. RNN uses pre-processed and feature-extracted data to classify attributes associated with assaults. The XGBoost model chooses features after preprocessing the datasets with the min-max scaling method. The following metrics are used to evaluate the simulations: f-measure, accuracy, precision, and recall. They are conducted using KDD datasets.

The SFSDT model, which was created by Thi-Thu-Huong et al. [16], combines the Sequence Forward Selection (SFS) method with the Decision Tree (DT) model. The second stage is constructing various IDS models to train on the best-selected feature subset. There are various forms of Recurrent Neural Networks (RNNs), some of which include Traditional RNN, Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU). The trial-learned models are applied to two IDS datasets: NSL-KDD (2010) and ISCX (2012). Dimensionality reduction approaches were developed by B. Venkatesh et al. [17] in order to reduce source dimensions. The two techniques for managing Dimensionality Reduction (DR) are Feature Selection (FS) and Feature Extraction (FE). This study focuses on feature selection approaches, and the survey length indicates that most FS techniques use static data. However, since web-based apps and the Internet of Things have grown in popularity, data are continuously generated and evolving, raising the possibility of noisy data.

An improved one-dimensional convolutional neural network and feature selection-based intrusion detection model were put into practice by Qingfeng et al. [18]. Using the extreme gradient boosting decision tree (XGboost) method to sort the preprocessed data, this model then used the comparison to remove 55 characteristics that contributed more. After that, the retrieved features were used to train the enhanced one-dimensional convolutional neural network (I1DCNN), which was then used to complete the final classification task. The enhanced one-dimensional convolutional neural network (FS-I1DCNN) intrusion detection model and feature selection addressed the drawbacks of the conventional machine learning approach, which relies on expert experience to extract features.

A novel intrusion detection system based on feature selection and ensemble learning techniques was created by Yuyang Zhou et al. [19]. First, a heuristic approach called CFS-BA is proposed for dimensionality reduction. It assesses the correlation between attributes to determine which subset is optimal. Next, we introduce an ensemble approach that combines the Random Forest (RF), Forest by Penalizing Attributes (Forest PA), and C4.5 algorithms. Finally, the voting mechanism is used to aggregate the probability distributions of the base learners for attack recognition. The trials were conducted using the NSL-KDD, AWID, and CIC-IDS2017 datasets. The results demonstrate that the proposed CFS-BA-Ensemble method outperforms other comparable and state-of-the-art approaches on several criteria.

3. Materials and Methods

3.1. Major Contributions

Based on this related work, we identified certain significant shortcomings that still persist in the IDS environment. We have outlined significant difficulties in existing work as follows:

3.1.1. Issues and Challenges

High Dimensionality**:** The complexity of high-dimensional network traffic data poses challenges to IDS efficacy and processing efficiency. Computational Overhead**:** Traditional IDS solutions may struggle to meet considerable computational demands, limiting real-time detection capabilities.

3.1.2. Contributions of this Study

The above challenges are addressed by our current study using different methodologies like Feature Selection (FS) and Dimensionality Reduction (DR) strategies. Our experimental results show improved model performance in terms of accuracy and reduced computational overhead. We have listed significant contributions of this work as follows:

1. Hybrid Approach: Integrates Feature Selection (FS) and Dimensionality Reduction (FE) approaches to maximize IDS performance.
2. Method Evaluation: Examines FS methods (filter-based, embedded-based, wrapper-based) and dimensionality reduction techniques (PCA, SVD) on the NSL-KDD and KDDcup99 datasets.
3. Performance Enhancement: Achieves high accuracy (up to 99.00%) with low runtimes, boosting IDS efficiency and reliability for real-world applications.

3.2. Dataset

Initially, we employed two different IDS datasets for performance analysis: NSL-KDD and KDDCup1999. Both are benchmark datasets sourced from the regular UCI repository. It has 41 input features and 1 target label.

Figures 1 to 4 show Exploratory Data Analytics (EDA) for understanding the NSL-KDD and KDDCup1999 datasets. It contains five classes: Normal, Probe, DOS, R2L, and U2R, which are unequally distributed across both datasets.

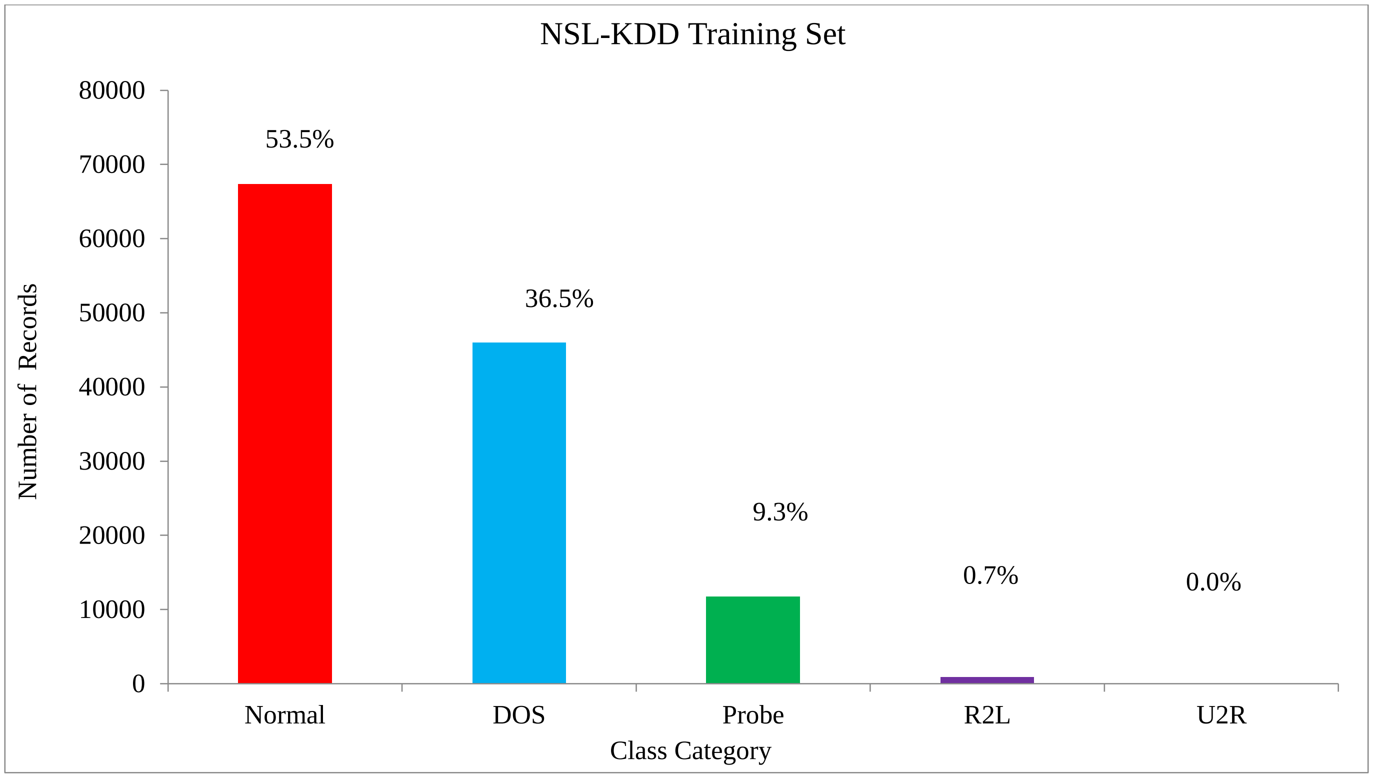


Fig. 1 NSL-KDD training set-class distribution

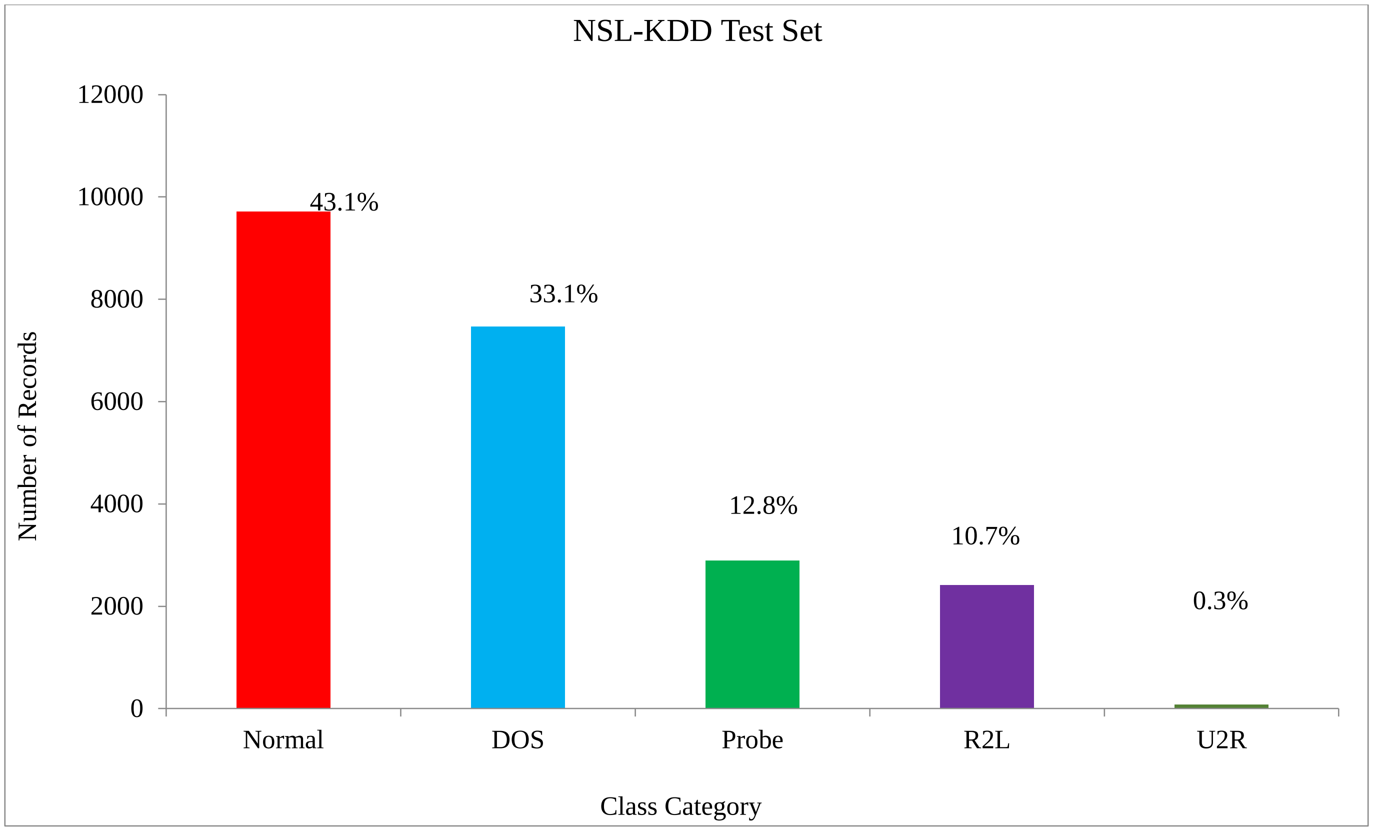


Fig. 2 NSL-KDD test set-class distribution

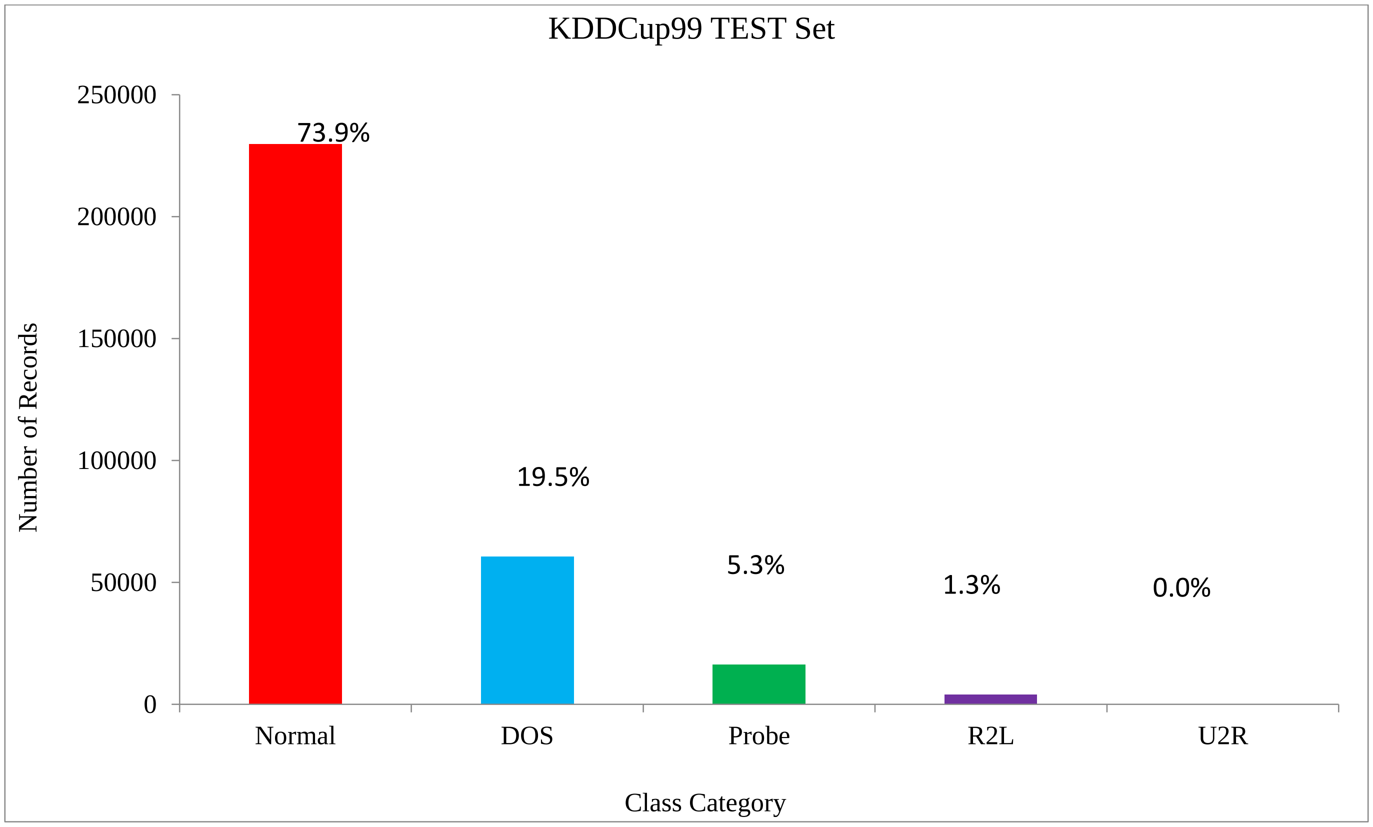


Fig. 3 KDDCup99 training set-class distribution

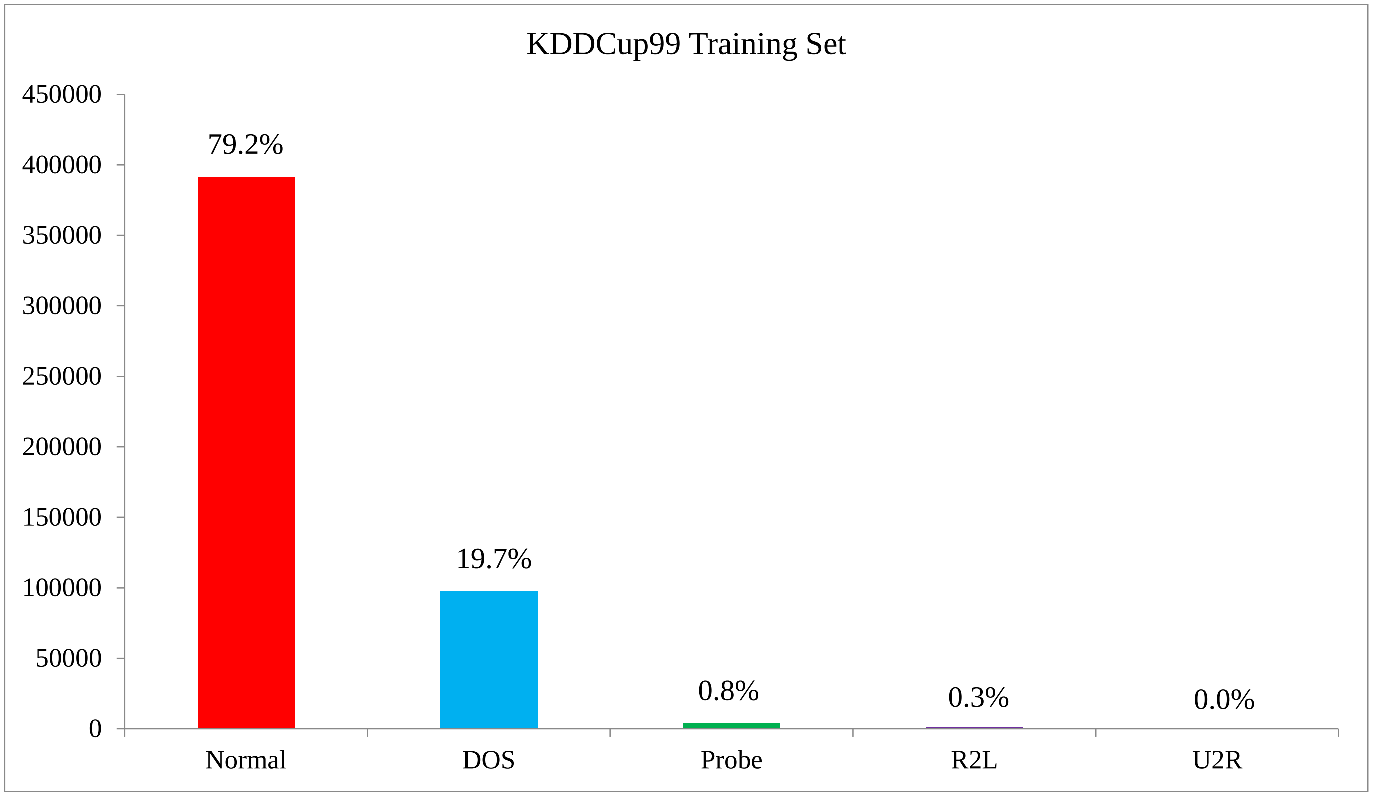


Fig. 4 NSL-KDD test set-class distribution

3.3. Preprocessing

We performed a data preprocessing task using IDS datasets to enhance the model performance. Data preprocessing is a vital task that ensures data quality and optimizes model performance. Initially, employ the removeDuplicates () and isnull () methods to manage any missing values and remove redundancy, particularly in the NSL-KDD and KDDCup99 datasets, which may contain duplicates and inconsistent information.

The Labelencoder () method converts the category values protocol_type, land, and flag to numerical values. These preprocessing efforts produce a refined dataset that can be utilized to train powerful IDS models capable of identifying and mitigating cybersecurity threats across a wide range of network circumstances.

3.4. Standard Machine Learning (ML) with IDS Dataset

After the preprocessing, we compared model performance using standard classifiers such as Decision Tree (DT), Random Forest (RF), Extra Trees (ET), and Voting Classifier. Table 1 shows each model's performance in terms of accuracy and model running time. Based on the results, both ML models provide better training and test accuracy for the KDDCup99 and NSL-KDD datasets, with DT providing 92.62% test set accuracy and 0.1896 seconds for model running time. The model computational complexity happens over the KDDCup99 dataset, which contains 41 features.

Table 1. KDDCup99-performance analysis using 41 features

Model	Train Acc$_y$ (%)	Test Acc$_y$ (%)	Train Time (Sec)	Test Time (Sec)
DT	**99.90**	**92.62**	**1.6386**	**0.1896**
RF	99.37	92.38	4.7084	1.6832
ET	99.41	92.26	1.1121	0.7673
Voting	99.99	92.43	6.8054	5.7322

Table 2. NSL-KDD-performance a nalysis using 41 features

Model	Train Acc$_y$ (%)	Test Acc$_y$ (%)	Train Time (Sec)	Test Time (Sec)
DT	**99.99**	**75.31**	**0.6950**	**0.0408**
RF	99.99	75.51	2.0691	0.3389
ET	99.99	74.50	0.5543	0.1639
Voting	99.99	75.42	3.1050	1.1350

Table 2 shows the classification performance and computational time on the NSL-KDD dataset using different classifiers such as DT, RF, ET, and Voting classifiers using 41 features. The results showed that DT provided 75.31% test set accuracy and a less computational time of 0.0408 seconds. Based on these analyses, both models achieved good accuracy.

However, computation time is an important consideration. Hence, we focus on Feature Selection (FS) and Dimensionality Reduction (DR) techniques.

3.5. Feature Selection Techniques with IDS Performance

3.5.1. Feature Selection

According to earlier findings, standard classifiers have a high computation time on the KDDcup99 and NSL-KDD datasets. So, Feature Selection (FS) comes into play to reduce the computational overhead for the aforesaid task. Feature selection is the process of selecting a subset of the most relevant features from the original feature set while deleting redundant, unnecessary, or noisy features. In this study, we have implemented different feature selection methods, such as statistical or filter-based, embedded-based, and wrapper-based. When we applied the FS approaches, we realized a greater reduction in computational time complexity than before. Detailed comparison steps are as follows.

3.5.2. Filter-Based Method

Initially, we applied a filter-based approach to select the best features from the 41 features on the KDDCup99 and NSL-KDD datasets. Filter-based methods rank the features using statistical measures like correlation, mutual information, and chi-squared tests. The model is built around the features that received the highest scores.

Correlation Matrix (CM)

A correlation matrix is simply a table showing variables' correlation coefficients. A matrix displays the relationship between all possible pairings of values in a table. It is an effective technique for summarizing large datasets and identifying and visualizing patterns in the given data. A correlation matrix is made up of rows and columns that represent the variables. Each table cell contains the correlation coefficient. It is crucial to optimizing the NSL-KDD and KDDCup99 datasets for Intrusion Detection Systems (IDS). Here's a thorough description of the correlation matrix approach and its mathematical foundations, along with steps for effective feature selection:

- Matrix Calculation: The correlation matrix R is computed as, given a dataset represented as an n×m matrix X (where n is the number of samples and m is the number of features), using the following formula:

$$R = \frac{1}{n-1}(X - \bar{X})^T (X - \bar{X}) \quad (1)$$

  Where $\bar{X}$ is the mean vector of X.

- Eigenanalysis: Eigenanalysis of the correlation matrix R helps know the key components and variance explained by various features. After performing the correlation matrix approach on the NSL-KDD and KDDCup99 datasets, the original number of features 41 was reduced to 30 and 29 on the KDDcup99 and NSL-KDD datasets, respectively. Tables 3 and 4 display extracted features with DT, RF, ET, and voting classifier performance and computational time for 30 and 29 features from the NSL-KDD and KDDCup99 datasets. Based on these studies, the results are more accurate and computationally efficient than our earlier study using 41 features.

Table 3. KDDCup99-performance analysis using 29 features

Model	Train Acc$_y$ (%)	Test Acc$_y$ (%)	Train Time (Sec)	Test Time (Sec)
DT	**99.41**	**92.27**	**1.0337**	**0.1479**
RF	99.93	92.48	3.6878	1.6314
ET	99.99	92.27	1.1405	0.7509
Voting	99.41	92.46	5.6982	5.4901

Table 4. NSL-KDD-performance analysis using 30 features

Model	Train Acc$_y$ (%)	Test Acc$_y$ (%)	Train Time (Sec)	Test Time (Sec)
DT	**99.97**	**71.89**	**0.5027**	**0.0401**
RF	99.97	74.21	1.7681	0.3489
ET	99.97	74.57	0.4671	0.1636
Voting	99.97	74.92	2.4914	1.0355

Tables 3 and 4 show the KDDCup99 NSL-KDD dataset model performance and time complexity comparisons for each model using the correlation matrix method. Based on this depiction, DT has a less computational time of 0.1479 and 0.0401 seconds for test sets, respectively; compared to the previous computation time, it has decreased computational overhead and provides good model accuracy.

3.5.3. Embedded-Based Method

In order to improve model accuracy and reduce computational overhead, we applied an embedded-based feature selection strategy to the NSL-KDD and KDD Cup datasets. Embedded-based feature selection techniques combine the benefits of filter and wrapper methods by considering feature interactions while remaining computationally efficient. These quick processing methods are comparable to the filter method but are more accurate. The following embedded-based feature importance approaches are focused on the NSL-KDD and KDDCup99 datasets, along with the underlying mathematics:

Random Forest Feature Importance

Random Forest, a common ensemble learning technique, are noted for their effectiveness and interpretability. They work by training a large number of decision trees, with the final prediction being the average of the individual tree predictions. For a Random Forest model, the Mean Decrease Impurity (MDI) of feature X_j is computed by averaging the decrease in impurity across all decision trees:

$$\mathrm{MDI}_{X_j} = \frac{1}{N_{trees}} \sum_{i=1}^{N_{trees}} \mathrm{impurity\_decrease}\ X_j^{(i)} \quad (2)$$

Where N_{trees} is the number of trees in the forest, and impurity decreases $X_j^{(i)}$ X_j in the I - th tree.

Decision Tree Feature Importance

Decision tree feature importance is a way of determining the significance of each feature in a decision tree model for predicting the target variable. Unlike other models, which derive feature importance from coefficients or weights, decision trees determine feature importance based on how each feature contributes to lowering impurity or entropy in the tree's nodes. Firstly, we train a decision tree model on the dataset (NSL-KDD or KDDcup1999). The decision tree algorithm splits the data at each node based on features that best separate the target variable (e.g., attack type or normal).

Gini Importance

In decision trees, Gini importance measures the contribution of each feature Xj to reducing impurity (Gini impurity) across all decision nodes where it is used for splitting. The Gini impurity at a node i is defined as:

$$I_{G(i)} = 1 - \sum_{c}^{C} (\ p\ (\ i\ , c\ )\ )^2 \quad (3)$$

Where:

- C is the number of classes.
- p (i, c) is the proportion of samples of class ccc in node i.

Equation 3. The Gini importance I_{Gini} (Xj) of feature Xj is computed by summing the purity decreases (Gini impurity decrease) achieved by splitting on Xj across all relevant decision nodes:

$$I_{Gini}(X_j) = \sum_{nodes} p\ (i)\ .\ \mathrm{purity\_decrease}_{X_j} \quad (4)$$

Where purity_decrease$_{Xj}$ represents the decrease in Gini impurity caused by splitting on feature Xj.

Entropy Importance

Alternatively, in decision trees using entropy as a criterion, the importance of feature Xj can be quantified based on the information gain achieved by splitting on that feature.

Equation (5) the entropy H at a node i is given by:

$$H(i) = -\sum_{c=1}^{C} p(i, c) \log_2 p(i, c) \quad (5)$$

Equation (6) the information gain IG (X_j) by splitting on feature X_j is calculated as the difference between the entropy of the parent node H(parent) and the weighted average of the entropies of its child nodes:

$$IG(X_j) = H(parent)\ - \sum_{children} \frac{N_{child}}{N_{parent}} H(child) \quad (6)$$

Where

- N_{child} and N_{parent} are the number of samples in the child and parent nodes, respectively.

Extra Trees Feature Importance

Extra Trees evaluates feature importance based on how much each feature reduces impurity (typically Gini impurity) across all decision nodes used for splitting. The following steps are as follows:

1. Train the ExtraTrees Model: Fit an ExtraTrees classifier or regressor on your dataset.
2. Impurity Decrease at Each Node: Calculate the impurity decrease purity decreases $_{Xj}$ for each feature X_j at nodes used for splitting.
3. Aggregate Importance: Sum up the impurity decreases across all decision nodes to obtain the MDI for each feature X_j:

$$I_{MDI}(X_j) = \sum_{nodes} p(i)\ .\ \mathrm{purity\_decrease}_{X_j} \ (7)$$

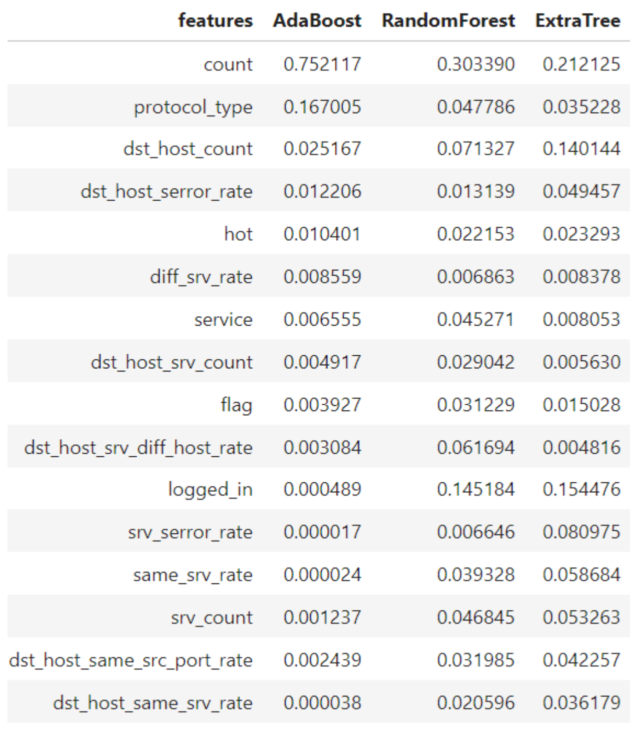
Where p(i) is the proportion of samples reaching node i.

4. Interpretation: Features with higher MDI values are more important as they contribute more to reducing impurity, thereby improving the model's predictive performance.

Extra Trees feature importance via MDI provides a straightforward way to assess the relevance of features during model training, helping analysts to focus on the most informative features for their predictive tasks. After applying the above embedded-based technique, we obtained 16 and 17 reduced subset features from 29,30 features in the KDDcup99 and NSL KDD datasets, respectively. Figure 7 depicts the top features ranked for further analysis using these embedded-based FS approaches.

Table 5. KDDCup99 -Embedded-based feature selection performance analysis using 16 features

Model	Train Acc$_y$ (%)	Test Acc$_y$ (%)	Train Time (Sec)	Test Time (Sec)
DT	99.99	92.38	0.5789	0.0946
RF	99.99	92.58	3.6538	1.4417
ET	89.99	92.41	0.9353	0.6721
Voting	99.98	92.49	5.0519	5.3113



features	AdaBoost	RandomForest	ExtraTree
count	0.752117	0.303390	0.212125
protocol_type	0.167005	0.047786	0.035228
dst_host_count	0.025167	0.071327	0.140144
dst_host_serror_rate	0.012206	0.013139	0.049457
hot	0.010401	0.022153	0.023293
diff_srv_rate	0.008559	0.006863	0.008378
service	0.006555	0.045271	0.008053
dst_host_srv_count	0.004917	0.029042	0.005630
flag	0.003927	0.031229	0.015028
dst_host_srv_diff_host_rate	0.003084	0.061694	0.004816
logged_in	0.000489	0.145184	0.154476
srv_serror_rate	0.000017	0.006646	0.080975
same_srv_rate	0.000024	0.039328	0.058684
srv_count	0.001237	0.046845	0.053263
dst_host_same_src_port_rate	0.002439	0.031985	0.042257
dst_host_same_srv_rate	0.000038	0.020596	0.036179

Fig. 5 Embedded-based feature selection ranked on features

Table 6. NSL-KDD -Embedded based feature selection performance analysis using 17 features

Model	Train Acc$_y$ (%)	Test Acc$_y$ (%)	Train Time (Sec)	Test Time (Sec)
DT	**99.97**	**72.03**	**0.2936**	**0.0340**
RF	99.76	72.57	1.4580	0.3110
ET	99.77	71.02	0.3939	0.1768
Voting	99.77	72.68	2.0012	1.0996

We evaluate the model performance using DT, RF, ET, and Voting classifiers with 16 and 17 features on the KDDcup99 and NSL-KDD datasets. Tables 5 and 6 show that DT provides less computing time (0.0946 and 0.0340 seconds on the KDDcup99 and NSL-KDD datasets respectively) when compared to our previous FS approach, the current Embedded-based FS strategy performs well in terms of computation time and accuracy for training and test sets, with 99.97% and 72.03%, respectively for NSL-KDD dataset and 99.99% and 92.38% for KDDCup99 dataset.

3.5.4. Wrapper Based Method

In order to reduce the time complexity, we used wrapper-based approaches to eliminate features. It's often known as greedy algorithms, which train the algorithm iteratively with a subset of features. Feature addition and removal occur based on the findings drawn from previous training of the model. Stopping criteria for selecting the best subset are typically pre-defined by the person training the model, such as when the model's performance decreases or a certain number of features has been achieved.

Recursive Feature Elimination (RFE)

A wrapper-based feature selection method recursively removes features from a model before selecting features based on their relevance. Here's a step-by-step overview of how RFE can be used with datasets like NSL-KDD and KDD Cup.

i. Step 1: Fit the model M on the dataset D with n features. M1=fit (M, D)
ii. Step 2: Obtain feature importance scores or coefficients β_1 from M_1.
$\beta1$=importance scores ($M_{1)}$
iii. Step 3: Rank features based on $\beta1$.
rank (f_1, f_2 ,…., fn) based on $\beta1$
iv. Step 4: Remove the least important feature f_n and obtain a new dataset D′ with n-1 feature.
$D'=D-f_n$
v. Step 5: Fit the model M on D′
M2=fit (M, D′)
vi. Step 6: Evaluate the performance $perf_2$ of M_2 using a performance metric.
perf2=evaluate (M_2, metric)
vii. Step 7: Iterate steps 2-6 until the desired number of features k is achieved or performance stabilizes.

Following the RFE approach, we extracted 8 optimum features from 16 and 17 input features on the KDDcup99 and NSL-KDD datasets. Tables 7 and 8 compare model performance based on these eight features.

Table 7. KDDCup99-wrapper-based feature elimination performance analysis using 8 features

Model	Train Acc$_y$ (%)	Test Acc$_y$ (%)	Train Time (Sec)	Test Time (Sec)
DT	99.87	92.33	0.3437	0.0731
RF	99.96	92.30	2.5731	1.5704
ET	99.96	92.35	0.8922	0.8252
Voting	99.96	92.30	3.7410	6.1301

Table 8. NSL KDD-wrapper based feature elimination performance analysis using 8 features

Model	Train Acc$_y$ (%)	Test Acc$_y$ (%)	Train Time (Sec)	Test Time (Sec)
DT	98.72	69.49	0.1359	0.0189
RF	99.52	71.44	1.1827	0.3628
ET	99.53	72.16	0.3408	0.1912
Voting	99.53	69.56	1.7740	1.1245

Tables 7 and 8 display the RFE technique comparison results for the KDDCup99 and NSL-KDD datasets, respectively. Based on this comparison analysis, the current RFE method's number of input features was reduced to six, and the results provided 92.33% and 69.49% accuracy for both KDDcup99 and NSL-KDD datasets using the DT classifier. Furthermore, the computation time was reduced to 0.0731 and 0.0189 seconds for both datasets utilizing the DT classifier. When compared to previous feature selection methods, the RFE achieves better accuracy and reduces computational time.

3.6. Proposed Work

The feature selection approaches yielded superior results; however, we created a novel hybrid model for the IDS environment to expedite computations.

This model integrates feature selection and feature reduction techniques, as illustrated in Figure 6, outlining its components and implementation steps.

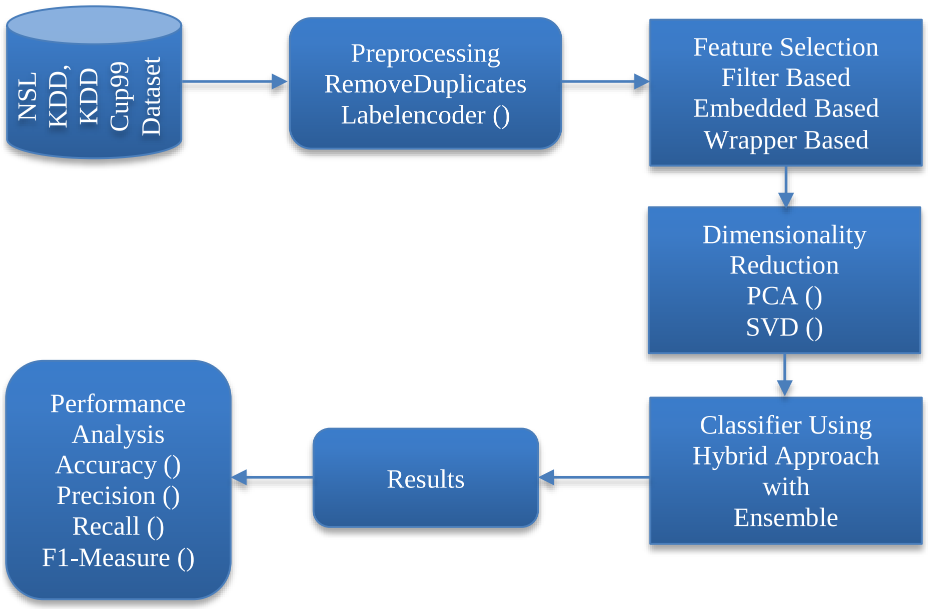

NSL KDD, KDD Cup99 Dataset
Preprocessing RemoveDuplicates Labelencoder ()
Feature Selection Filter Based Embedded Based Wrapper Based
Dimensionality Reduction PCA () SVD ()
Classifier Using Hybrid Approach with Ensemble
Results
Performance Analysis Accuracy () Precision () Recall () F1-Measure ()

Fig. 6 Proposed architecture

3.6.1. Dimensionality Reduction with IDS Performance

We utilized two dimensionality reduction methods, such as Principal Component Analysis (PCA) and Singular Value Decomposition (SVD), to decrease the number of features (dimensions) in our dataset while preserving maximum variance or information. Specifically, we applied these techniques to reduce the dimensionality to six components for both the KDDCup99 and NSL-KDD datasets.

Principal Component Analysis (PCA)

Initially, we applied the Principal Component Analysis (PCA) technique on the KDDcup99 and NSL-KDD dataset with 8 input features. It is a statistical technique that transforms a high-dimensional dataset into a set of linearly uncorrelated components known as principal components.

The principle components are ordered so that the first component captures the maximum variance in the data, the second component captures the second-highest variance, and so on. The main goal of PCA is to reduce the number of components k to six from 8 on the KDDCup99 and NSL-KDD datasets. The mathematical steps are as follows:

1. Covariance Matrix: Given a dataset X with n samples and d features, the covariance matrix Σ is computed as:

$$\sum = \frac{1}{n} X^T \tag{8}$$

2. Eigenvalue Decomposition: Compute the eigenvalues λ_i and eigenvectors vi of Σ:

$$\sum v_i = \lambda_i v_i \tag{9}$$

3. Selecting Components: Select the top k eigenvectors corresponding to the largest eigenvalues to form matrix $V_k = [v_1, v_2, \ldots, v_k]$.

4. Projection: Project the original dataset X onto the reduced-dimensional space:

$$X_{reduced} = XV_k \tag{10}$$

Here, V_k is d × k, and $X_{reduced}$ becomes n × k, representing the dataset with reduced dimensions. After computing the eigenvectors and eigenvalues of the covariance matrix, we select the top six eigenvectors corresponding to the largest eigenvalues. These eigenvectors form the basis for the reduced-dimensional space in which the data is projected. Tables 9 and 10 demonstrate the PCA performance analysis on the KDDCup99 and NSL-KDD datasets, respectively. When we applied the PCA, we achieved higher accuracy and reduced computational time compared to the previous feature selection approach.

Table 9. KDDCup99 PCA performance analysis using 6 features

Model	Train Acc_y (%)	Test Acc_y (%)	Train Time (Sec)	Test Time (Sec)
DT	99.98	88.19	0.8531	0.0542
RF	99.96	85.92	10.010	1.7656
ET	99.68	91.61	0.8346	0.6527
Voting	99.96	86.26	11.3700	5.3921

Table 10. NSL KDD PCA performance analysis using 6 features

Model	Train Acc_y (%)	Test Acc_y (%)	Train Time (Sec)	Test Time (Sec)
DT	97.32	70.41	1.0555	0.0214
RF	99.51	72.01	9.2912	0.3391
ET	99.53	72.36	0.4235	0.1939
Voting	99.53	72.07	12.1623	1.0518

Tables 9 and 10, based on these findings, both prediction models produced higher accuracy. They reduced model performance time, with 70.42%, 72.01%, 72.36%, and 72.07% accuracy for test sets on the NSL-KDD dataset employing DT, RF, ET, and voting classifiers, respectively. Using the same classifiers, test accuracy on the KDDCup dataset was 88.19%, 85.92%, 91.61%, and 86.26%, respectively. Compared to earlier findings, it produced the best accuracy for both models. In addition, PCA with DT, PCA with RF, PCA with ET, and PCA with Voting classifiers required less computing time for both datasets. Tables 9 and 10 show detailed analysis and computation times.

Singular Value Decomposition (SVD)

SVD is another method for dimensionality reduction, particularly useful for datasets that are not centered or have missing values.

1. Decomposition: SVD decomposes the original n × d dataset matrix X into three matrices: U, Σ, and V^T.
   - U is an n×n orthogonal matrix of left singular vectors.
   - Σ is an n×d diagonal matrix of singular values.
   - V^T is a d×d orthogonal matrix of right singular vectors.
2. Reducing Dimensions: To reduce the dimensionality to k=6, we truncate U, Σ, and V^T matrices:
   - Keep the first k columns of U as U_k.
   - Keep the first k rows and columns of Σ as $Σ_k$.
   - Keep the first k rows of V^T as V^T_R.
3. Projection: The reduced dataset is obtained by multiplying X with $V^k{}_T$ (or equivalently $U_kΣ_k$).

Tables 11 and 12 provide the SVD performance analysis results for the KDDCup99 and NSL-KDD datasets, respectively. When we used the SVD, we got the best accuracy and reduced computational time compared to the previous methods.

Table 11. KDDCup99 SVD performance analysis using 6 features

Model	Train Acc_y (%)	Test Acc_y (%)	Train Time (Sec)	Test Time (Sec)
DT	99.82	88.19	0.6924	0.0513
RF	99.96	85.92	8.8102	1.7875
ET	99.96	88.82	1.0388	0.6753
Voting	99.96	86.03	11.2704	6.0915

Table 12. NSL KDD- SVD performance analysis using 6 features

Model	Train Acc_y (%)	Test Acc_y (%)	Train Time (Sec)	Test Time (Sec)
DT	97.32	70.61	1.2298	0.0127
RF	99.51	72.01	9.3120	0.3348
ET	99.53	72.53	0.4119	0.1934
Voting	99.53	72.00	11.0200	1.0685

Tables 11 and 12 show that both prediction models provided higher accuracy and reduced model performance time, with 70.61%, 72.01%, 72.53%, and 72.08% accuracy for test sets on the NSL-KDD dataset using DT, RF, ET, and voting classifiers, respectively. Using the same classifiers, test accuracy on the KDDCup dataset was 88.19%, 85.92%, 88.72%, and 86.03%. When compared to previous results, it acheived better accuracy for both models. Tables 10 and 11 show that SVD with DT, SVD with RF, SVD with ET, and SVD with Voting classifiers used less computation time for both datasets.

SVD reduces dimensionality to 6 components by retaining the first 6 columns of U, the first 6 rows and columns of Σ, and the first 6 rows of V^T. The reduced dataset is then calculated by multiplying the original dataset X by $V_k{}^T$.

In both PCA and SVD, the choice of 6 components is based on retaining a significant amount of variance or information from the original dataset while reducing its dimensionality. The specific mathematical operations involve matrix multiplications, Eigenvalue Decompositions (PCA), and Singular Value Decompositions (SVD) to achieve the desired dimension reduction. These techniques are crucial for optimizing data representation in machine learning applications, including intrusion detection systems, as they reduce computing complexity while potentially boosting model performance.

4. Results and Discussion

In this study, we implemented feature selection and dimensionality reduction approaches to enhance IDS performance. Table 13 presents the comparison results of prediction models based on their performance metrics and computation time. According to the findings, the base models achieved better accuracy but required higher computation times, utilizing 41 features. Specifically, they achieved 99.99% and 92.62% training and testing accuracy, respectively, on the KDDCup99 dataset, with computation times of 1.6386 seconds and 0.1896 seconds for training and testing analysis. Conversely, while other classifiers also exhibited strong accuracy on this dataset, they required longer computation times. To optimize computation time while preserving average accuracy, we applied a filter-based feature selection approach using a correlation matrix. This method identified 29 features from the initial 41 features in the KDDCup99 dataset. Utilizing these 29 selected features, our DT analysis with the Correlation Matrix approach reduced computation times to 1.0337 seconds and 0.1479 seconds for training and test datasets, respectively. The embedded feature selection approach reduced the feature set from 29 to 16 features, significantly reducing computation times to 0.5789 seconds for training and 0.0946 seconds for testing using DT classifiers on the same dataset. Similarly, other classifiers showed marked reductions in computation times with these models. Meanwhile, the wrapper-based feature selection approach further reduced the feature set from 16 to 8 features, achieving less computation times of 0.3437 seconds for training and 0.0731 seconds for testing datasets. Moreover, it maintained better accuracy for both prediction models.

Table 13. Feature selection and dimensionality reduction performance analysis on the KDDCup99 dataset

Model	Train Acc$_y$ (%)	Test Acc$_y$ (%)	Train Time (Sec)	Test Time (Sec)
DT	97.32	70.61	1.2298	0.0127
RF	99.51	72.01	9.3120	0.3348
ET	99.53	72.53	0.4119	0.1934
Voting	99.53	72.00	11.0200	1.0685

Table 14. Feature selection and dimensionality reduction comparison analysis on the KDDCup99 dataset

Model	No. of Features	Train Acc$_y$ (%)	Train Time (Sec)	Test Acc$_y$ (%)	Test Time (Sec)
DT	41	97.32	1.2298	92.62	0.0127
DT with filter-based Approach	29	99.41	1.0337	92.27	0.1479
DT with Embedded-based Approach	16	99.99	0.5789	92.38	0.0946
DT with Wrapper-based Approach	8	99.87	0.3437	92.33	0.0731
DT with PCA	6	99.88	0.8531	88.19	0.0542
DT with SVD	6	99.82	0.6924	88.91	0.0513

Finally, our proposed hybrid feature selection with a dimensionality reduction approach produced the number of features from 8 to 6, showing reduced computation time for both models. The PCA with the DT model achieved 99.88% accuracy in training and 88.19% in testing, and the computation times were 0.8531 seconds and 0.0542 seconds, respectively. Similarly, the SVD with DT model achieved 99.82% and 88.91% accuracy, with the model running times of 0.0924 seconds and 0.0513 seconds on the KDDCup dataset. Based on these results, our hybrid approach achieves improved computation times while maintaining average accuracy for both models. Therefore, we recommend using this model for real-time IDS environments. Table 14 presents the performance of the NSL-KDD dataset for both methods.

Our proposed hybrid feature selection and dimensionality reduction approach achieved 97.31% and 70.61% accuracy for training and test sets, with model training and testing times of 1.0555 seconds and 0.0214 seconds using the DT with the PCA model. The SVD with DT model achieved a similar accuracy of 97.32% for both training and test sets, with model training and testing times of 1.2298 seconds and 0.0127 seconds, respectively. Initially, the DT model without feature selection or dimensionality reduction provided 99.99% and 75.31% accuracy for training and testing datasets, with model running times of 0.6950 seconds and 0.0408 seconds. Upon comparing these results, our proposed hybrid approach demonstrates high performance on both the NSL-KDD and KDDCup datasets regarding accuracy and time complexity.

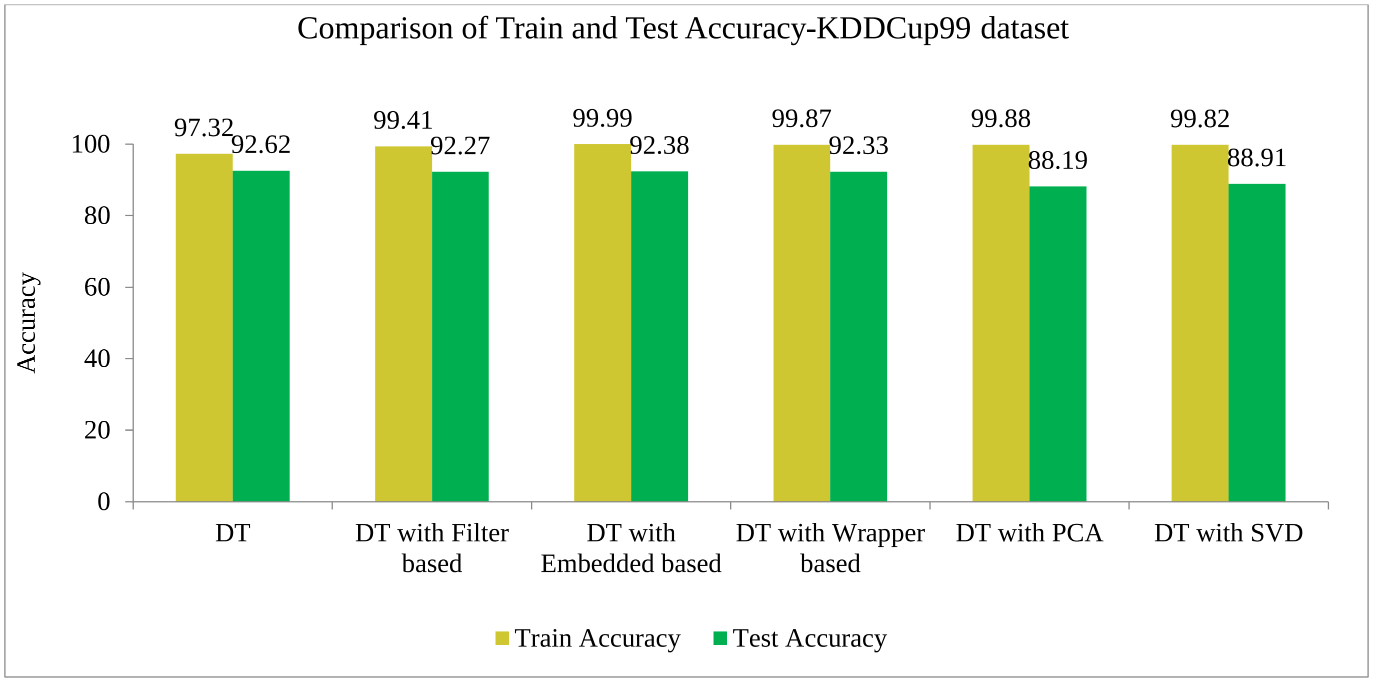


Fig. 7 Overall comparison using the KDD Cup1999 dataset

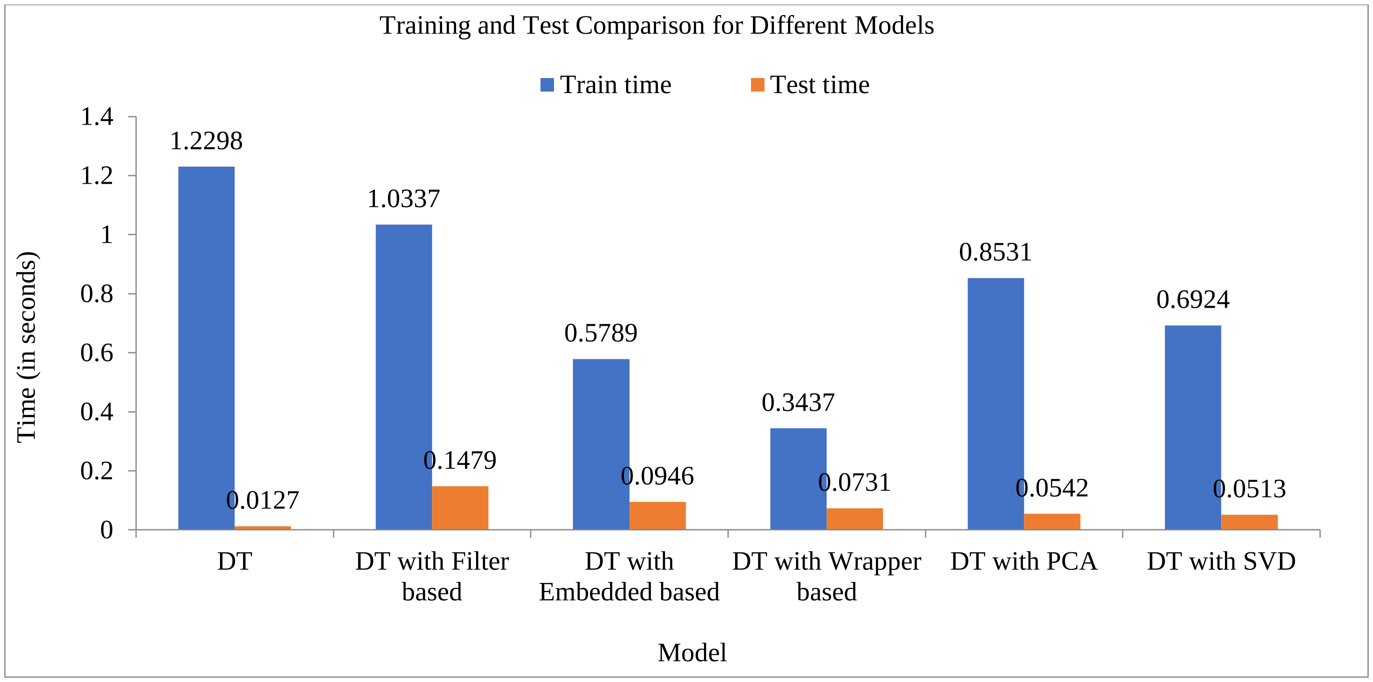


Fig. 8 Overall model time complexity comparison using the KDD Cup1999 dataset

Table 15. Feature selection and dimensionality reduction comparison analysis on the NSL-KDD dataset

Model	No. of Features	Train Acc$_y$ (%)	Train Time (Sec)	Test Acc$_y$ (%)	Test Time (Sec)
DT	41	99.99	0.6950	75.31	0.0408
DT with filter-based Approach	30	99.97	0.5027	71.89	0.0401
DT with Embedded-based Approach	16	99.97	0.2936	72.03	0.0340
DT with Wrapper-based Approach	8	98.72	0.1359	69.49	0.0189
DT with PCA	6	97.32	1.0555	70.41	0.0214
DT with SVD	6	97.32	1.2298	70.61	0.0127

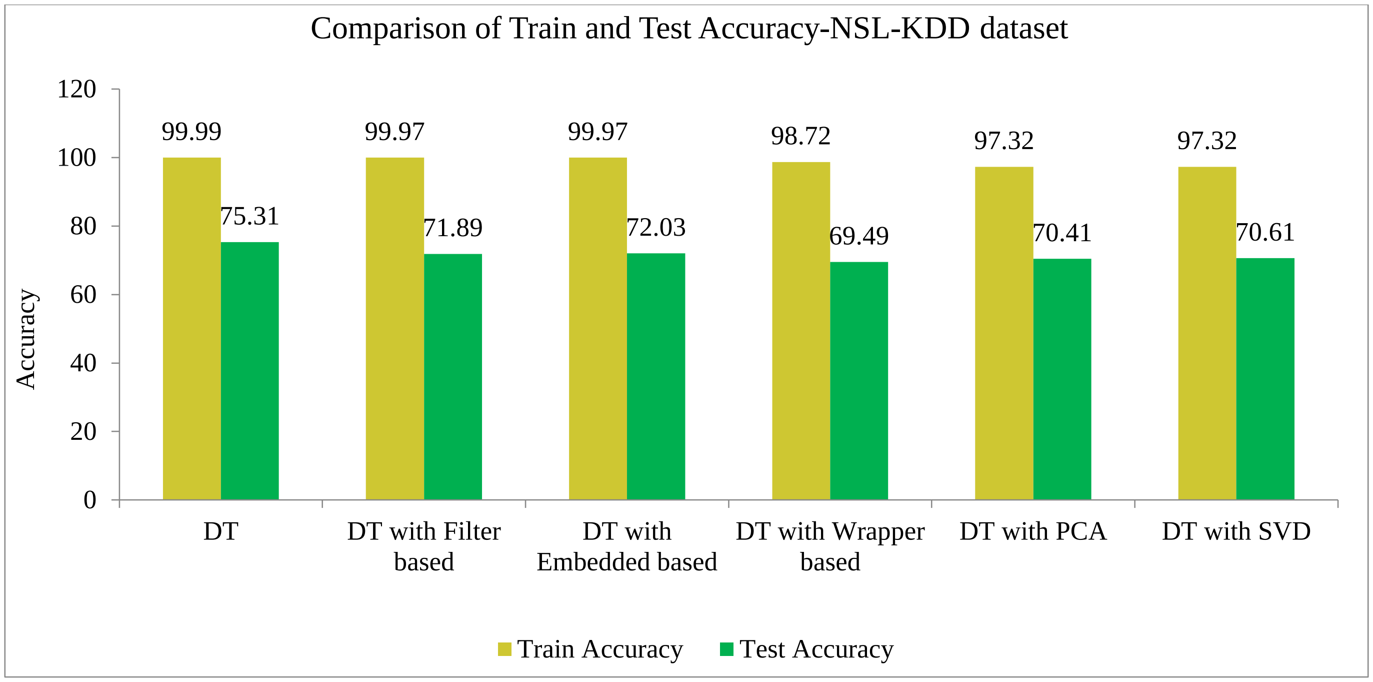


Fig. 9 Overall comparison using the KDD Cup1999 dataset

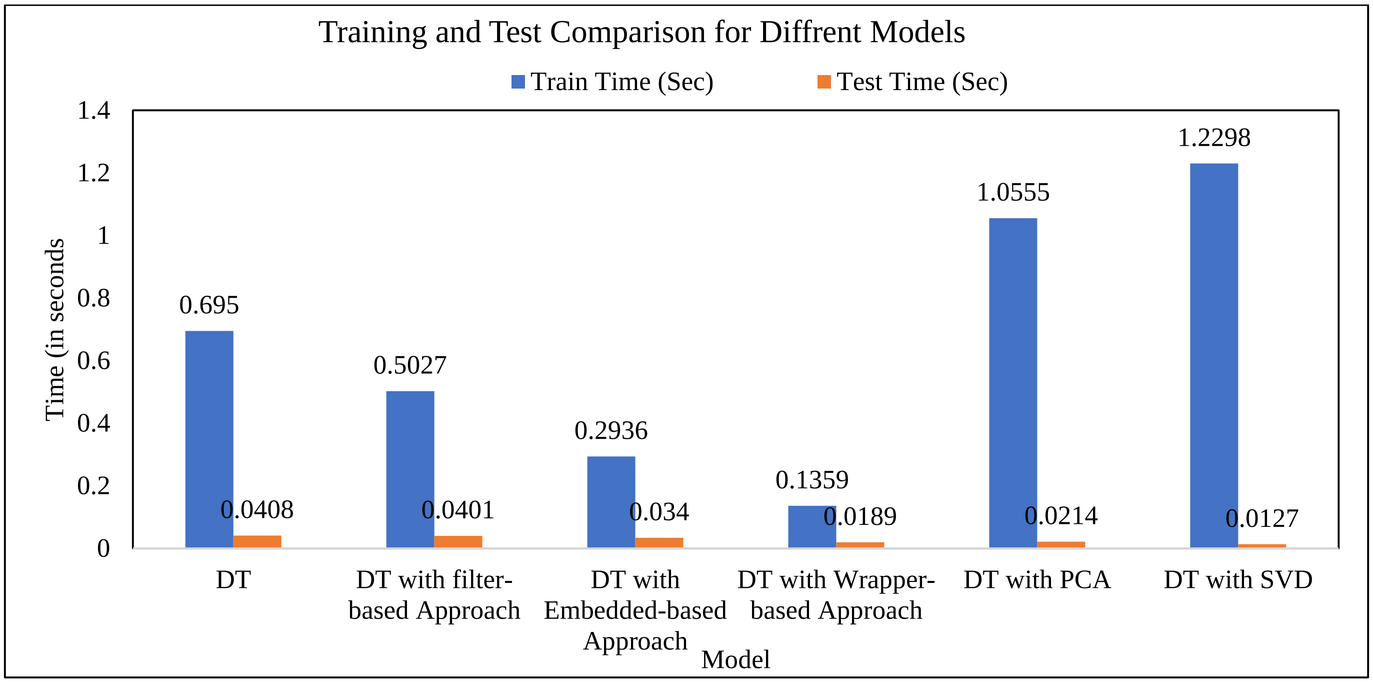


Fig. 10 Overall model time complexity comparison using the KDD Cup1999 dataset

Table 15 depicts the NSL-KDD dataset performance for both FS and DR methods, and our proposed approach hybrid feature selection dimensionality reduction approach achieved 97.31%,70.61% for training and test set accuracy 1.0555 and 0.0214 Seconds for model training and testing performance using DT with PCA model. The SVD with DT model provides 97.32% and 70.61% accuracy for both training and test set accuracy and 1.2298 and 0.0127 Seconds for model training and testing performance. Initially, the DT model provided 99.99% and 75.31% accuracy for training and testing datasets and 0.6950 and 0.0408 Seconds for model running time for both training and testing datasets. Our comparison results are shown as individual methods' performance and time complexity. Based on this result, our proposed hybrid approach provides better performance on both the NSL-KDD and KDDCup99 datasets.

5. Conclusion

This study emphasizes the importance of Feature Selection (FS) and Dimensionality Reduction (DR) strategies in optimizing Intrusion Detection Systems (IDS) for reliable performance in identifying network anomalies. The findings show that different FS methods, such as filter-based, embedded-based, and wrapper-based, have distinct merits in balancing accuracy and computing efficiency. Furthermore, dimensionality reduction approaches such as Principal Component Analysis (PCA) and Singular Value Decomposition (SVD) have successfully lowered feature dimensions while maintaining IDS accuracy. Future research could focus on improving hybrid FS and DR approaches for IDS across diverse datasets and investigating emerging machine learning and data preprocessing techniques to improve detection capabilities and scalability in real-world network security applications.

References

[1] S. Al-Helli, and A. Akbas, “Guided Feature Selection and Dimensionality Reduction Method for IDS Improvement in DDoS Attacks,” *International Conference on Engineering Technologies*, Konya, Turkey, pp. 75-80, 2020. [Google Scholar] [Publisher Link]

[2] Ankit Thakkar, and Ritika Lohiya, “Attack Classification Using Feature Selection Techniques: A Comparative Study,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, pp. 1249-1266, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[3] Arshid Ali et al., “Network Intrusion Detection Leveraging Machine Learning and Feature Selection,” *IEEE 17th International Conference on Smart Communities: Improving Quality of Life Using ICT, IoT and AI (HONET)*, Charlotte, NC, USA, pp. 49-53, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[4] Raghava Satya SaiKrishna Dittakavi, “Dimensionality Reduction Based Intrusion Detection System in Cloud Computing Environment Using Machine Learning,” *International Journal of Information and Cybersecurity*, vol. 6, no. 1, pp. 62-81, 2022. [Google Scholar] [Publisher Link]

[5] Fadi Salo, Ali Bou Nassif, and Aleksander Essex, “Dimensionality Reduction with IG-PCA and Ensemble Classifier for Network Intrusion Detection,” *Computer Networks*, vol. 148, pp. 164-175, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[6] M. Di Mauro et al., “Supervised Feature Selection Techniques in Network Intrusion Detection: A Critical Review,” *Engineering Applications of Artificial Intelligence*, vol. 101, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[7] Ghanshyam Prasad Dubey, and Rakesh Kumar Bhujade, “Optimal Feature Selection for Machine Learning Based Intrusion Detection System by Exploiting Attribute Dependence,” *Materials Today: Proceedings*, vol. 47, no. 17, pp. 6325-6331, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[8] Gulab Sah, and Subhasish Banerjee, “Feature Reduction and Classifications Techniques for Intrusion Detection System,” *International Conference on Communication and Signal Processing*, India, pp. 1543-1547, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[9] Majdi Maabreh et al., “Towards Data-Driven Network Intrusion Detection Systems: Features Dimensionality Reduction and Machine Learning,” *International Journal of Interactive Mobile Technologies*, vol. 16, no. 14, pp. 1-13, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[10] Majid Torabi et al., “A Review on Feature Selection and Ensemble Techniques for Intrusion Detection System,” *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 5, pp. 1-16, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[11] Muhammad Naveed et al., “A Deep Learning-Based Framework for Feature Extraction and Classification of Intrusion Detection in Networks,” *Wireless Communication and Mobile Computing*, vol. 2022, no. 1, pp. 1-11, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[12] Razan Abdulhammed et al., “Features Dimensionality Reduction Approaches for Machine Learning Based Network Intrusion Detection,” *Electronics*, vol. 8, no. 3, pp. 1-27, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[13] Razan Abdulhammed et al., “Features Dimensionality Reduction Approaches for Machine Learning Based Network Intrusion Detection,” *Electronics*, vol. 8, no. 3, pp. 1-27, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[14] Rami Ahmad et al., “Feature-Selection and Mutual-Clustering Approaches to Improve DoS Detection and Maintain WSNs’ Lifetime,” *Sensors*, vol. 21, no. 14, pp. 1-25, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[15] Surendra Yadav et al., “Mitigation of Attacks via Improved Network Security in IOT Network Environment Using RNN,” *Measurement: Sensors*, vol. 32, pp. 1-8, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[16] Thi-Thu-Huong Le, Yongsu Kim, and Howon Kim, “Network Intrusion Detection Based on Novel Feature Selection Model and Various Recurrent Neural Networks,” *Applied Sciences*, vol. 9, no. 7, pp. 1-29, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[17] B. Venkatesh, and J. Anuradha, “A Review of Feature Selection and Its Methods,” *Cybernetics and Information Technologies*, vol. 19, no. 1, pp. 1-24, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[18] Qingfeng Li et al., “An Intrusion Detection Model Based on Feature Selection and Improved One-Dimensional Convolutional Neural Network,” *International Journal of Distributed Sensor Network*, vol. 2023, no. 1, pp. 1-12, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[19] Yuyang Zhou et al., “Building an Efficient Intrusion Detection System Based on Feature Selection and Ensemble Classifier,” *Computer Networks*, vol. 174, 2020. [CrossRef] [Google Scholar] [Publisher Link]