IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 11 January 2024, accepted 30 January 2024, date of publication 6 February 2024, date of current version 13 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3362794

RESEARCH ARTICLE

Weighted Feature Selection for Machine Learning Based Accurate Intrusion Detection in Communication Networks

GAURAV TRIPATHI[1], VISHAL KRISHNA SINGH[2], VARUN SHARMA[1], AND MAJITHIA VIVEK VINODBHAI[3]

[1]Department of Computer Science, Indian Institute of Information Technology, Lucknow, Uttar Pradesh 226002, India
[2]School of Computer Science and Electronics Engineering, University of Essex, CO4 3SQ Colchester, U.K.
[3]TCS Innovation Laboratories, Infocity, Gandhinagar, Gujarat 382421, India

Corresponding author: Vishal Krishna Singh (v.k.singh@essex.ac.uk)

ABSTRACT Network intrusion detection systems work on huge data sets, with large feature sets dominated by noisy data and irrelevant features, resulting in steep degradation in detection accuracy and a steep proliferation in model training and computation time. This work presents a novel method to optimize the feature selection process in machine learning algorithms for accurate detection of intrusion attacks in communication networks. The proposed method targets features with a high impact on the target variable to optimize feature selection and reduction. The CICIDS-2017 data set is used to test the performance of the proposed approach. Results prove the dexterity of the proposed method as it is able to achieve an almost 51% reduction in irrelevant features and increases the detection accuracy of the tuned random forest classifier to 99.9% with an almost 50% reduced model computation time.

INDEX TERMS Communication networks, machine learning, random forest, intrusion detection, network attacks.

I. INTRODUCTION

Intrusion Detection System (IDS) is an effective technique for prevention against cyber-attacks in communication networks and for preventing intrusion in communication systems [1]. Emerging technologies such as Big Data, Internet of Things (IoT), Edge Computing, Cloud Computing, Wirleless Sensor Networks (WSNs), etc., [2], [3], [4], [5], [6], [7], [8] generate a large amount of multidimensional data with numerous features that must be reduced to create an efficient IDS. Consequently, optimization techniques are applied to these heterogeneous massive data sets, not only because of the increasing number of tuples but also because of the exceedingly high number of irrelevant features in each tuple, which may result in high false positives, redundant observations, and high computational complexity.

An important aspect of the degrading performance of existing IDS has been poor feature extraction, which is a major focus of the proposed work. Feature selection is imperative to reduce the non-contributing features in the classification, which have a significant impact on the efficiency and accuracy of the IDS. Therefore, if the data set is divided into components that are less redundant and have more accurate logs, the testing component will have better model performance. As a result, for optimal IDS, choosing a less redundant data set for testing and training components of the system model is one of the most important steps to improve the accuracy and performance of the communication networks.

Notably, recent research has proved that the methods of machine learning (ML) can considerably enhance the performance of network IDS by making them less complex, more proactive, and less costly, by reducing the time spent on repeated tasks, and by allowing businesses to employ their resources more efficiently, given that they are supplemented by data that completely represents the environment. ML approaches have been widely used to successfully detect network intrusions or abnormal behavior

The associate editor coordinating the review of this manuscript and approving it for publication was Ghufran Ahmed.

owing to their capability of learning from historical data and statistical analysis. For instance, the authors in [9] investigated random search as well as grid search approaches by fine-tuning the hyper-parameters of the Support Vector Machine (SVM) classifier. The findings conclude that the predictive capabilities of the SVM classifier combined with random search deliver acceptable accuracy. In yet another application of ML for IDS, the uniform distribution-based balancing method called UDBB is proposed to manage the unbalanced distribution of minority class instances in the CICIDS-2017 data set [10]. All of the data files from the CICIDS-2017 data set were combined into a single file to compare the imbalanced scenario (with CICIDS-2017's original distribution) to the balanced case (after applying the UDDB technique). The authors in [1] conducted a review of network IDS (NIDS) technologies, reviewing their types as well as their benefits and drawbacks. A comprehensive comparison of various previously known research outcomes is evaluated and compared on the most common public data sets, such as NSL KDD, CICIDS, and others. An empirical study, proposed in [11], demonstrates that the algorithm does indeed live up to its claims of mining the medical data set by presenting a fuzzy system using genetic algorithm (GA) combined with SVM. The results prove that the models built with fewer features have a lower miscalculation rate and a higher diagnosis rate. The results of [12] revealed that the proposed model performed well in terms of accuracy, precision, recall, *f1* score with reduced selected features, and understanding ability of the learning process. An empirical comparison of the consistent feature selection measures with the wrapper method reveals that the former is significantly more consistent than the latter. A careful review of the existing works reveals that most of these methods consider an accuracy-time trade-off and hardly consider the model's computational time. Considering the huge volume of data passing through the IDS, the performance can be significantly optimized by addressing these issues.

The organisation of the paper is as follows: Section II presents a comprehensive review of the recent related works in IDS, followed by Section III, where the problem statement is defined. Section IV presents the data set description, and Section V presents the proposed solution. The results and detailed discussions are presented in Section VI, with the concluding comments and future directions in Section VII.

II. RELATED WORK

In one of the significant works on IDS, the authors in [13] suggested that a good intrusion detection system must have essential functionality to give it's attack classification results that are more accurate and efficient. The authors trained multiple functions to produce highly accurate results, which were verified on the NSL-KDD data set. For this study, the data set was divided into five sections based on the attack categories (DoS, Probe, R2L, U2R, and Normal). The final accuracy for attack classification achieved in the experiments was reported to be 98.20% for *Normal*, 99.60% for *Probe*, 99.70% for *R2L*, 97.2% for *DoS*, and 92.50% for *U2R*. After feature extraction, the execution time was reduced from 1.73 ms for 41 features to 0.3 ms for a total of 12 relevant features. Since then, many approaches have been proposed to optimize the accuracy of such IDS without imposing any extra computational overhead. The work in [14], is one such method where a new multi-objective teaching learning-based algorithm (NTLBO) is proposed to provide feature subset selection in NIDS. The method considers 22 relevant features selected out of 79 features and obtains 97.50% accuracy for the CICIDS-2017 data set. The authors are able to achieve a significant improvement in the overall accuracy as compared to [15] and [16], but at the cost of high computational time. The recent methods proposed in [17], [18], [19], and [20], are some examples of the use of wrapper-based, SGM, encoder-based, and ensemble-based methods, respectively. Thus, targeting the lack of an optimal feature reduction strategy and the trade-off with the model computation time in these methods, the proposed work aims to address the issues of accuracy, feature reduction, and model computation time. The state-of-the-art *chi-square* statistical feature selection method is optimized to obtain statistical features for accurate classification. The proposed method is analysed and compared against a set of established methods, for which detailed results are presented. The novel contributions of this work are summarised as follows:

- An accurate ML-based feature extraction mechanism for intrusion detection in communication networks is proposed to address the issues of accuracy and computation time.
- To the best of our knowledge, the proposed method is the first ever step to address the accuracy-time trade-off and is based on the *chi-square* statistical method for feature selection, where k best features with the highest feature score are selected.
- Results on the CICIDS-2017 data set prove that the proposed method is able to reduce irrelevant features, improve accuracy, and decrease computation time.
- A comprehensive comparative analysis of the proposed method with various ML-based approaches, namely Linear-SVM classifiers, Naive Bayes classifiers, Decision Tree classifiers, and Random Forest classifiers, is presented. For a fair comparative analysis, the experimental environment and data set are replicated in all the possible cases. The analysis is validated by reducing irrelevant features and performing classification on *binary classification*, *multi-class classification*, and *all attack labels*.

The table 1 highlights the *Main Approaches*, *Advantage*, and *Drawbacks* of the major methods used for performance analysis in this work.

III. PROBLEM DESCRIPTION

In most cases, the raw TCP dump format is used to gather and store network traffic data. It is possible to pre-process this data and turn it into connection records later. A connection is

TABLE 1. Recent related works with CICIDS-2017 data set.

Related Work	Main Approach	Advantage	Drawback
[14]	Improved TLBO, Multi-Class Classification, Linear Regression (LR) and SVM	Accuracy	Computation Time
[15]	SMOTE, Binary Classification and Adaboost	Feature Reduction	Accuracy, Computation Time
[16]	BMCD, Multi-Class Classification, Random Forest (RF), MLP and NB	Accuracy	Feature Reduction
[17]	Wrapper Method, Multi-Class Classification and SVM	Accuracy, Computation Time	Feature Reduction
[18]	SGM, Multi-Class Classification, RF and MLP	Accuracy	Computation Time, Feature Reduction
[19]	Auto Encoder, Multi-Class Classification and RF	Accuracy	Feature Reduction, Computation Time
[20]	Ensemble, Multi-Class Classification and NN	Accuracy	Feature Reduction, Computation Time

nothing more than a series of TCP packets with well-defined beginning and ending times. Each connection record contains 100 bytes of data and is designated as either 'Normal' or an 'Attack' with a specific attack type. There is a vector for every connection record, and it is defined as follows:

$$V = \{f_1, f_2, f_3 \ldots . f_n, C_{lbl}\} \tag{1}$$

where f stands for features with length n, each f's value $\in$ D, and C_{lbl} stands for a class label.

However, existing ML based algorithms for IDS, deliver unacceptable accuracy with data sets having large feature sets. Understandably, large number of features result in irrelevant features mitigating the efficiency of the models. Additionally, large feature sets lead to high computation and training time for ML models.

IV. DATA SET DESCRIPTION AND MULTI-MODAL DATA

In this study, the CICIDS-2017 [21] data set, is used to validate the efficiency of the proposed method and for performance comparison with the state-of-the-art methods. The CICIDS-2017 data set is comprised of benign and most of the popular common network attacks. A novel B-Profile system [22] was used to profile the abstract behavior of human interactions in order to generate naturalistic benign background traffic. The data set is created from 25 users, based on the HTTP, HTTPS, FTP, SSH, and email protocols. The data capturing period started at 9 a.m., Monday, July 3, 2017 and ended at 5 p.m. on Friday July 7, 2017, for a total of 5 days. The implemented attacks include Brute Force FTP, Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet, and DDoS. The attacks are executed in both the morning and afternoon on Tuesday, Wednesday, Thursday, and Friday. The CICIDS-2017 data set has 2, 827, 876 entries and was built using actual traces of benign and fourteen types of attacks retrieved from network traffic data. There are 2, 271, 320 recordings for innocuous traffic and 556, 556 records for attacks. The data set is split into two partitions: *the training set* and *the testing set*, using the train test split method from the sci-kit-learns module in a ratio of 7:3. The CICIDS-2017 data set contains 79 attributes in total. The distribution of every class label from the CICIDS-2017 data set is shown in table 2.

TABLE 2. Attack label counts of CICIDS-2017 dataset.

Attack Labels	Attack Count
Benign	2,271,320
DoS Hulk	230,124
Port Scan	158,804
DDoS	128,025
DoS Golden Eye	10,293
FTP Patator	7,935
DoS Slowloris	5,796
SSH Patator	5,987
DoS Slowhttptest	5,499
Bot	1956
Web attack: Brute Force	1507
Web attack: XSS	652
Infiltration	36
Web attack: SQL Injection	21
Heartbleed	11

V. PROPOSED SOLUTION

The proposed methodology is divided into four steps, which are summarised as follows:

1) **Data Pre-processing:** Data pre-processing involves data cleaning for noise reduction, normalization for preserving the information values, variable encoding to extract binary features from categorical features, and attack labelling into seven categories.
2) **Feature Selection:** Feature selection is used to determine the features that contribute the most to establishing the best-class prediction for the goal variable.
3) **Model Training and Optimization:** Train the classifier model on the training data by categorising the data set into one of three groups: *binary classification*, *multi-class classification*, and *all attack labels*. After training the model on the training data, its performance is evaluated and compared using multiple performance metrics.
4) **Performance Evaluation:** The testing data set is used to forecast and evaluate the proposed method using a variety of performance indicators such as accuracy, precision, recall, and *f1* score. The number of accurate and inaccurate predictions made by the model in relation to the data's actual results (target value) is represented by a confusion matrix, as shown in figure 1.

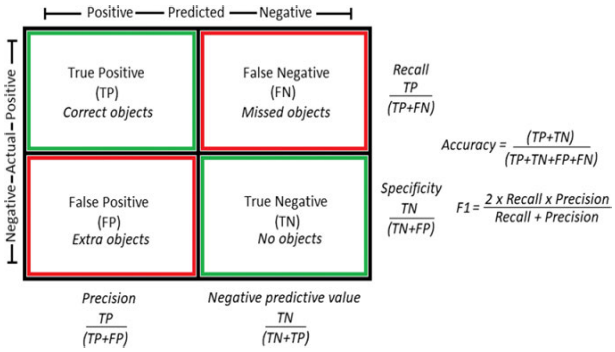


FIGURE 1. Performance evaluation: confusion matrix.

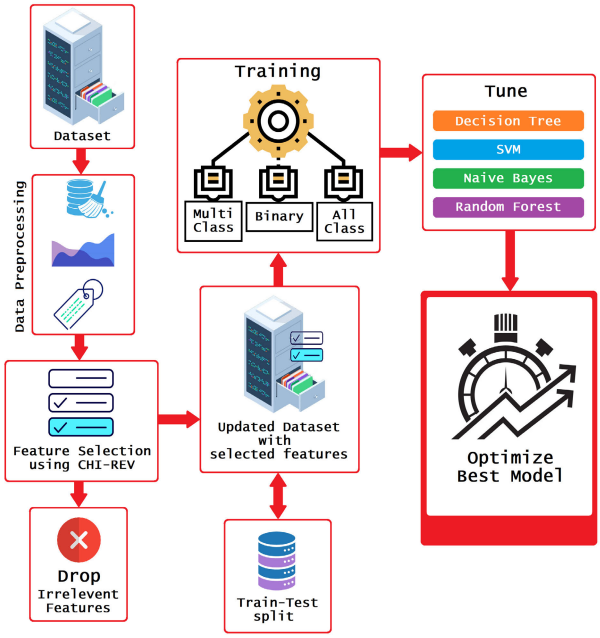


FIGURE 2. End-to-end proposed model.

- The accuracy is determined by dividing accurate forecasts by the total number of predictions and computing the percentage of accurate forecasts.
- Precision, Recall, and *f1* score calculation is shown in the figure 1.

The steps of the end-to-end proposed model, implemented to achieve the desired objectives, are shown in figure 2.

A. DATA PRE-PROCESSING

Data pre-processing consists of data cleaning, encoding, labelling, splitting, and normalization. More specifically, in this work, the data pre-processing involves removing the redundant values and filling in the missing entries by using the binning method to obtain 70 features. All the values marked as, *'Null'*, *'NaN'*, or *'infinite'* are replaced by *'NaN'*. Only unique value features are considered for normalization, resulting in the deletion of eight more attributes. Each attack's *label counts* are then examined, and the outcome is presented in table 2. Further, in order to confine the numerical values of data in a range, typically $0 - 1$, without affecting the range discrepancies of the actual values or losing information, *min-max* normalization is used for computation optimization and error mitigation. The equation 2 shows the *min-max* normalization as used on the CICIDS-2017 data set.

$$X' = a + \frac{(x - min(x))(b - a)}{max(x) - min(x)} \tag{2}$$

where $X =$ feature value, $a =$ lower boundary range of feature value, $b =$ upper boundary range of feature value and $min(x)$ and $max(x)$ represent the minimum and maximum values, respectively, of every tuple in that feature attribute.

Finally, the data set is checked for the data type of all the attributes present in the CICIDS-2017 data set. Attributes with integer and float data types, remain unchanged, however, all the categorical values are encoded.

B. FEATURE SELECTION

Typically, Chi-square is helpful for reducing irrelevant features and selecting relevant features in statistics. It is utilized to determine the independence of two events by calculating the Chi-square co-relation between each feature [23]. Chi square scores are computed as:

$$\chi^2 = \sum \frac{(Observed\ Frequency - Expected\ Frequency)^2}{Expected\ Frequency} \tag{3}$$

While using chi-square, weightage of the positive difference between *'observed'* and *'expected frequency'* is considered to be the same. However, in typical large heterogeneous data sets, if *'observed'* value is higher than *'expected'* value, then the *'weightage'* of this positive difference should be considered higher because the actual (observed) value might be impacting target variable. At the same time, negative difference cannot be neglected because it may have some dependence on the target variable. To address this issue, the novel *chi-rev* method, is introduced to differentiate between the negative and the positive difference between *'feature'* and *'target variables'*. A *'scaling factor'* given by β, is used to allocate more weightage to the *'positive difference'* of the *'observed'* frequency and the *'expected'* frequency. Positive and negative differences between features and target variables, are considered separately and different weights are assigned using the scaling factor, to reduce the negative difference of low observed frequency features. The steps of the proposed optimization are as follows:

- **Step 1:** Initialize the parameters.
- **Step 2:** if *Observed Frequency* is greater than *Expected Frequency* then calculate the positive difference by using the formula:
  $\chi^2(+ve) \leftarrow \chi^2(+ve) + \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$
- **Step 3:** if *Observed Frequency* is less than *Expected Frequency* then calculate the negative difference by using the formula:
  $\chi^2(-ve) \leftarrow \chi^2(-ve) + \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$

- **Step 4:** Return 0 if *Observed Frequency* is equal to the *Expected Frequency*.
- **Step 5:** Repeat from step 2 to Step 4 for all Unique Tuple Values and all Class Labels.
- **Step 6:** Calculate Chi-square value by using scaling factor 'β', using the following formula:
  $\chi^2 \leftarrow \beta\ X\ \chi^2(+ve) + (1-\beta)\ X\ \chi^2(-ve)$
- **Step 7:** Return Chi-square value.

Algorithm 1 Proposed *CHI-REV*(k)

Require: k = Select k best features
Ensure: χ^2 = Chi-square value of feature with respect to target variable

function CHI-REVk
$m > 0, n > 0$
$i \leftarrow 0, j \leftarrow 0$
$m \leftarrow UniqueTupleValues$
$n \leftarrow ClassLabels$
$O_{ij} \leftarrow ObservedFrequency$
$E_{ij} \leftarrow ExpectedFrequency$
$\chi^2(+ve) \leftarrow 0, \chi^2(-ve) \leftarrow 0$
$\beta \leftarrow 0.75$
while (i <**m**) **do**
  while (j <**n**) **do**
    if (O_{ij} - E_{ij}) > 0 **then**
      $\chi^2(+ve) \leftarrow \chi^2(+ve) + \frac{(O_{ij}-E_{ij})^2}{E_{ij}}$ ▷ **Positive difference**
    else if (O_{ij} - E_{ij}) < 0 **then**
      $\chi^2(-ve) \leftarrow \chi^2(-ve) + \frac{(O_{ij}-E_{ij})^2}{E_{ij}}$ ▷ **Negative difference**
    else if (O_{ij} - E_{ij}) == **0 then return 0**
    end if
  end while
end while
$\chi^2 \leftarrow \beta\ X\ \chi^2(+ve) + (1-\beta)\ X\ \chi^2(-ve)$ ▷ **Scaling factor**
return χ^2
end function

The figure 3 shows a graph plotted after calculating cumulative feature scores using the described method in algorithm 1. A careful observation of the cumulative feature scores, as presented in figure 3, shows that just around 40 features, out of all the features, are more significant and can provide nearly 99% of the information. Thus, it is recommended to select only those features that are empirical to decrease the model's training time. The proposed *chi-rev* is used to select 40 most relevant characteristics for model development and evaluation.

C. DATA LABELING AND SPLITTING FOR CLASSIFICATION

Three distinct classification techniques are considered for feature classification. Each ML method considered in this work (Linear-SVM, Naive Bayes, Random Forest, and Decision Tree), is tested with these classification techniques. The classification techniques and their corresponding output attributes are presented below:

- **Binary Classification:** The output attribute, corresponding to binary label classification, is shown in table 3.
- **Multi-class Classification:** The output attribute, corresponding to multi-class classification, is shown in table 4.
- **All Attack Label Classification:** The output attribute, corresponding to all attack label classification, is shown in figure 4.

TABLE 3. Attack classification using binary classification.

Attack Categories	Attack Labels
Normal	Benign
Attack	DoS Golden eye, DoS Hulk, DoS Slowhttpptest, DDoS, DoS Slowloris, SSH Patator, Port scan, FTP Patator, Web attack: Brute Force, Bot Web attack: XSS

TABLE 4. Attack classification using multi-class classification.

Attack Categories	Attack Labels
Benign	Benign
DoS	DoS Golden eye, DoS Slowloris, DoS Slowhttpptest, DoS Hulk
Probe	Port scan
DDoS	DDoS
Brute force	SSH Patator, FTP patator
Botnet	Bot
Web attack	Web attack: XSS, Web attack: Brute Force

D. MODEL TRAINING WITH ML ALGORITHMS

The Linear-SVM, Naive Bayes, Decision Tree, and Random Forest classifiers are applied and trained on each type of classification stated above. After pre-processing and optimizing the feature data set based on feature scores, each ML algorithm is applied to the following three categories:

1) Binary Classification: There are two labels in this category: *normal* and *attack*, as shown in table 3.
2) Multi-class Classification: There are seven separate attack categories as shown in table 4.
3) All Attack Labels Classification: There are a total of twelve attack labels in this category, as shown in figure 4.

VI. RESULTS AND ANALYSIS

A. EXPERIMENTAL ENVIRONMENT

For a fair analysis of the proposed IDS, all the methods used for comparative analysis were applied to a common data set, i.e., the CICIDS-2017. The comparative analysis is presented for the results obtained for an identified scenario with the 2, 827, 876 entries of the CICIDS-2017 dataset. A total of 2, 271, 320 recordings for innocuous traffic and 556, 556 records for attacks were considered, and a ratio of 7:3 was considered for training and testing sets. It is important to note that, due to the variable nature of approaches,

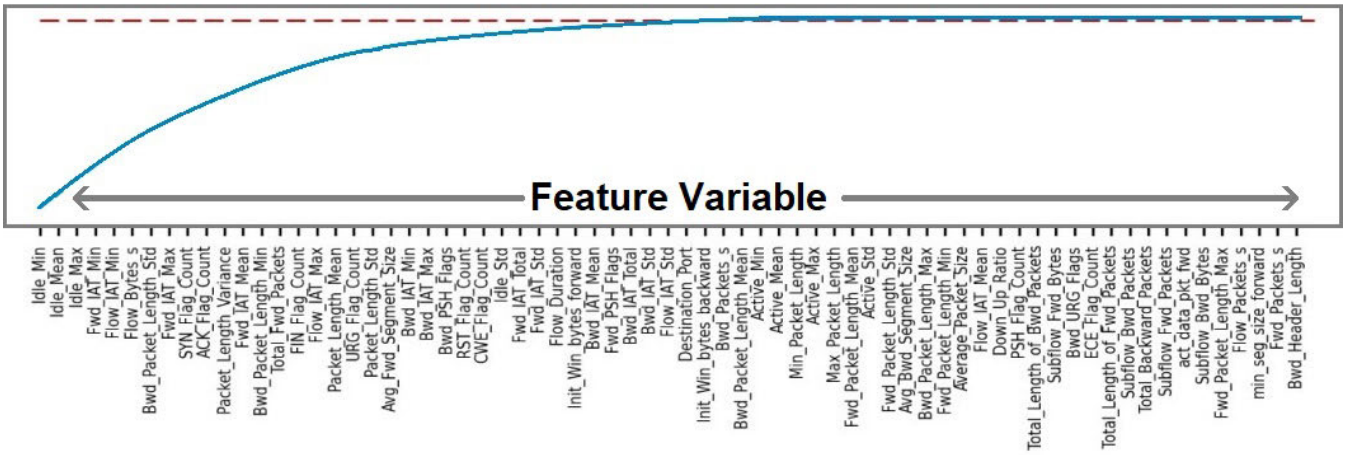


FIGURE 3. Cumulative feature scores using proposed *CHI-REV*.

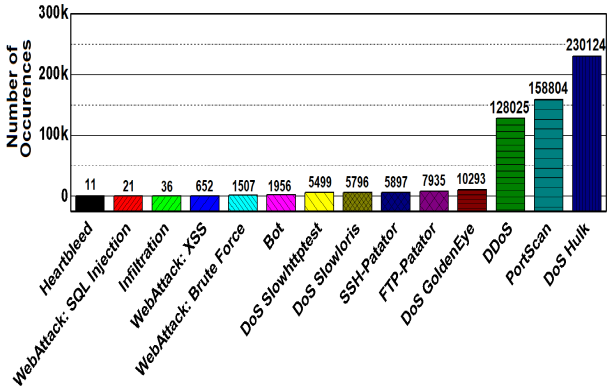


FIGURE 4. All attack labels with number of occurrences.

parameters, and methods used in different ML-based network IDS, the results for a common set of parameters for the identified data set were not available. Hence, the proposed as well as the approaches used for comparative analysis were tested on a set of common parameters and are mentioned in the table 5.

TABLE 5. Experimental environment.

Parameter	Value
Total Entries	2, 827, 876
Total Attack Type	14
Total Number of Attacks	556, 556
Total Attributes	79
Innocuous Traffic Recordings	2,271,320
Processor	13^{th} Gen, Intel(R) Core(TM) i7-1365U

B. COMPARATIVE ANALYSIS

The table 6 shows a detailed comparison of the proposed random forest classifier using *chi-rev* with existing state-of-the-art methods. A careful observation of the table 6 presents the following facts:

1) The proposed *chi-rev* is able to achieve the highest accuracy of 99.90%, 99.89% and 99.86% on binary, multi-class, and all-label data when tuned together with a random forest classifier.
2) The proposed scheme has the best feature reduction at 50.63% and is able to outperform all the compared approaches.
3) A clear explanation for the efficiency of the proposed method is its ability to optimize the feature selection process by using weighted values for the positive and negative differences between expected and observed frequencies so as to reduce the model computation time and number of required features.

C. LINEAR-SVM

The proposed *chi-rev*, when applied together with Linear-SVM, has the following output:

- *Binary Classification:* The Linear-SVM classifier training time is around 129 seconds for the binary classification model, whereas the testing time is reported as 0.04 seconds. The accuracy of the Linear-SVM model is observed to be 92.73%. The confusion matrix for the binary classification using Linear-SVM is shown in figure 5-(a).
- *Multi-class Classification:* The Linear-SVM classifier training time is around 269 seconds for multi-class classification, whereas the testing time is reported to be 0.12 seconds. The accuracy of the Linear-SVM model is observed to be 94.32%. The figure 5-(b) shows the confusion matrix for the multi-class classification using Linear-SVM.
- *All Attack Labels Classification:* Linear-SVM classifier training time is around 361 seconds, whereas testing time is 0.18 seconds. The accuracy of the SVM model is 94.39%. The Linear-SVM confusion matrix for all label classifications is shown in figure 5-(c).

D. DECISION TREE

The proposed *chi-rev* when applied together with a decision tree, has the following output:

TABLE 6. Comparison of the proposed model with existing methods on CICIDS-2017 data set.

Existing Method	Method	Classification	Model	Feature Reduction	Accuracy	Model Computation (CPU) Time
Mohammad Aljanabi et al (2021)[14]	Improved TLBO	Multi-class	LR SVM	72.50%	97.50% 93.00%	29 seconds 5484 seconds
Arif Yulianto et al (2019)[15]	SMOTE	Binary	Adaboost	68.35%	81.83%	Not mentioned
Amer Abdulrehman et al (2019)[16]	BMCD	Multi-class	RF MLP NB	87.50%	99.32% 94.82% 75.35%	Not mentioned
S.U. Jan et al (2019)[17]	Wrapper based feature selection	Multi-class	SVM	90.25%	98.03%	208 seconds
Hongpo Zhang et al (2020)[18]	SGM	Multi-class	RF MLP	Not mentioned	93.80% 99.60%	Not mentioned
Hassan Musafer et al (2020)[19]	Auto encoder	Multi-class	Random Forest	85%	99.50%	2034 seconds
Saikat Das et al (2021)[20]	Ensemble based	Multi-class	Ensemble Neural Network	72.50%	98.90%	Not mentioned
Proposed work	**CHI-REV**	**Binary Multi-class All labels**	**Random Forest**	**50.63%**	**99.90% 99.89% 99.86%**	**495 seconds**

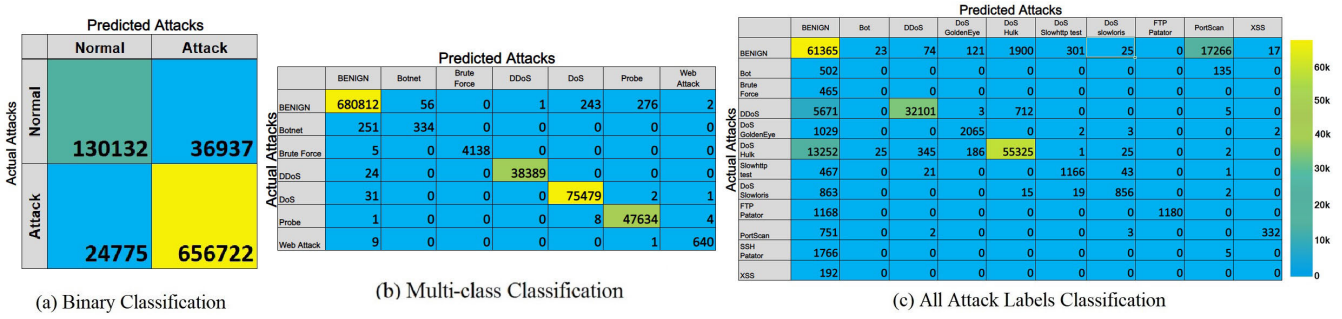


FIGURE 5. Confusion matrix of linear-SVM.

- *Binary Classification:* The decision tree classifier's training time is reported to be around 82 seconds for the binary classification model, whereas the testing time is observed to be 0.16 seconds at a reported accuracy of 99.88%. The figure 6-(a) shows the confusion matrix for the decision tree classifier for binary labels.
- *Multi-class Classification:* The decision tree classifier's training time is around 81 seconds for the multi-class classification model using a decision tree. The testing time for the same is reported to be 0.17 seconds at an accuracy of 99.87%. The figure 6-(b) shows the confusion matrix for the multi-class classification using a decision tree.
- *All Attack Labels Classification:* The training time for this model using a decision tree is observed to be approximately 82 seconds, while the testing time is reported to be 0.18 seconds at an accuracy of 99.83%. The corresponding confusion matrix is presented in figure 6-(c)

E. NAIVE BAYES

The proposed *chi-rev* when applied together with Naive Bayes, has the following output:

- *Binary Classification:* The Naive Bayes classifier training time is observed to be approximately 20 seconds for the binary classification model, while the testing time is 0.11 seconds at a reported accuracy of 87.06%. The figure 7-(a) shows the obtained confusion matrix of the Naive Bayes method for binary classification.

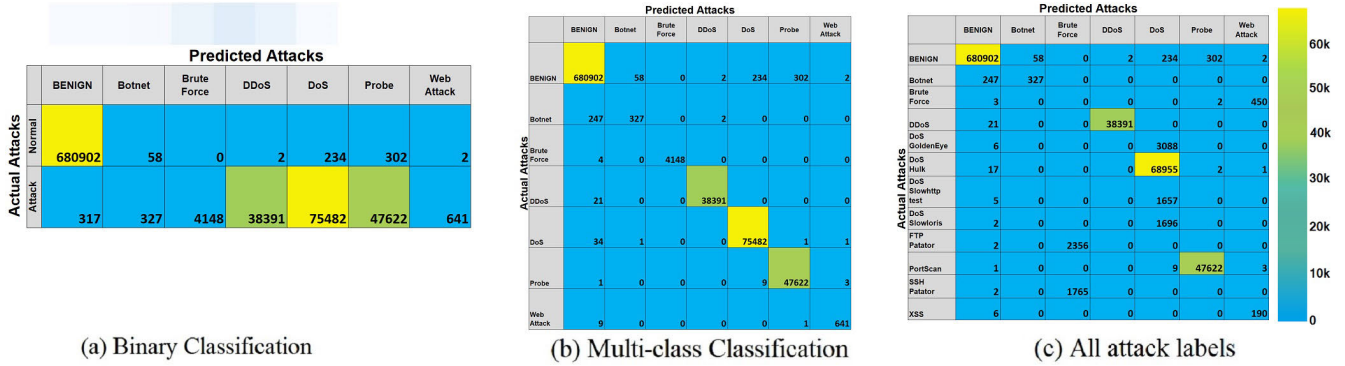


FIGURE 6. Confusion matrix of decision tree.

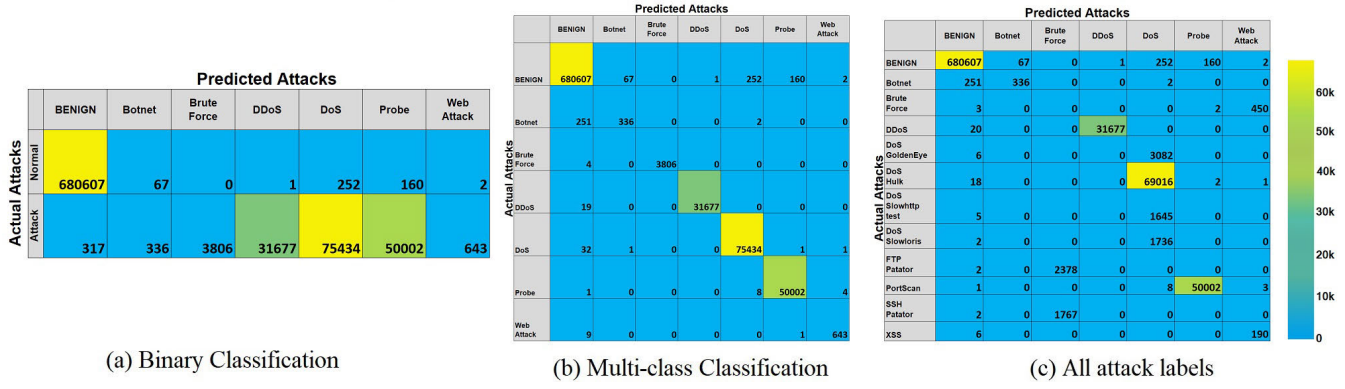


FIGURE 7. Confusion matrix of naive bayes.

- *Multi-class Classification:* The Naive Bayes classifier's training time on the multi-class model is reported to be around 7 seconds, with a testing time of 0.13 seconds at the reported accuracy of 86.53%. The confusion matrix for the corresponding scenario is presented in figure 7-(b).
- *All Attack labels Classification:* The Naive Bayes classifier's training time on all attack labels is reported to be 9 seconds, with a testing time of 0.19 seconds at an accuracy of 86.39%. The corresponding confusion matrix is shown in figure 7-(c).

F. RANDOM FOREST

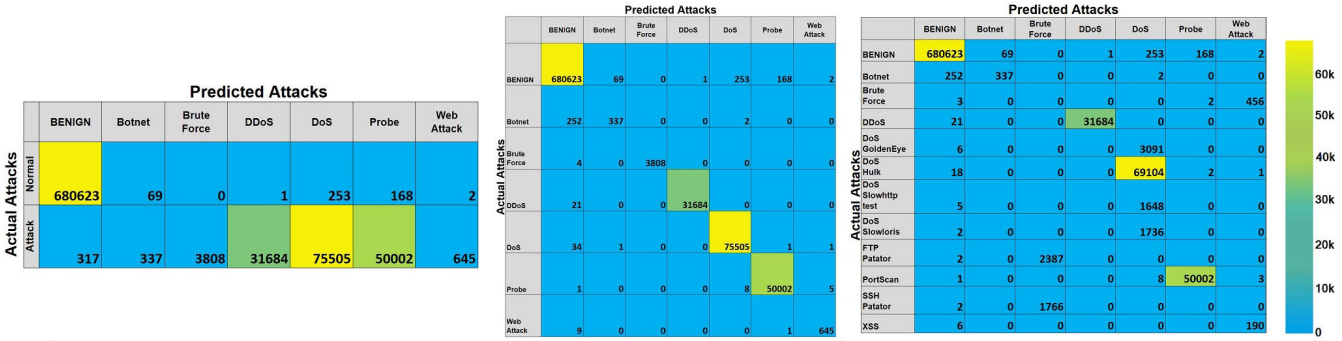
The proposed *chi-rev* when applied together with Random Forest, has the following output:

- *Binary Classification:* The random forest classifier training time is reported to be 572 seconds for the binary classification model, whereas the testing time is observed to be 10 seconds. The accuracy of the random forest model is reported at 99.90% and is shown through a confusion matrix in figure 8-(a).
- *Multi-class Classification:* For this classification, the random forest classifier's training time is found to be 495 seconds, whereas the testing time is 12 seconds at an accuracy of 99.89% and is shown in a confusion matrix in figure 8-(b).
- *All Attack Labels Classification:* The random forest classifier's training time is measured to be approximately around 508 seconds for all label classification models, with a reported testing time of 14 seconds at an accuracy of 99.86%. The corresponding output is presented in figure 8-(c).

G. ANALYSIS

Table 7 shows results achieved by various ML models using the proposed *chi-rev* feature selection method on different labels. It is evident from the presented results that, as compared to all the considered methods, the random forest algorithm is able to outperform the other classification algorithms with high accuracy as well as high precision, recall, and *f1* score. As shown in the table 7, the random forest algorithm achieves accuracy as high as 99.9%, with maximum values of *Precision*, *Recall*, and *f1 Score* values being reported at 99.8%, 99.8%, and 99.85%, respectively.

The table 8 shows a comparative analysis of the training time of ML models before the application of the proposed *chi-rev* (shown under *B*) and after the application of the proposed *chi-rev* (shown under *A*) and can be evidently noted to be reduced up to 50% with the proposed method. The results



(a) Binary Classification (b) Multi-class Classification (c) All attack labels

FIGURE 8. Confusion matrix of random forest.

TABLE 7. Result analysis of various ml models on different labels.

Class	SVM	Decision Tree	Naïve Bayes	Random Forest	Evaluation Criteria
All Label	**94.39%**	**99.83%**	**86.49%**	**99.86%**	**1.Accuracy**
	59.88%	90.63%	34.66%	91.37%	2.Precision
	49.57%	91.08%	21.01%	89.38%	3.Recall
	53.14%	90.85%	22.60%	90.21%	4.F1-score
Binary	**92.73%**	**99.88%**	**87.06%**	**99.90%**	**1.Accuracy**
	89.34%	99.80%	84.89%	99.80%	2.Precision
	87.12%	99.84%	70.71%	99.89%	3.Recall
	88.17%	99.82%	74.74%	99.85%	4.F1-score
Multi-class	**94.32%**	**99.87%**	**86.53%**	**99.89%**	**1.Accuracy**
	65.84%	96.90%	35.04%	97.66%	2.Precision
	55.02%	97.79%	29.29%	93.62%	3.Recall
	57.18%	97.34%	31.23%	95.23%	4.F1-score

TABLE 8. Model training time (in seconds) before (B) and after (A) removing ir-relevant features using *CHI-REV*.

Classification	SVM		Decision Tree		Naive Bayes		Random Forest	
	B	A	B	A	B	A	B	A
All Label	576	361	100	82	10	9	980	508
Binary	248	129	96	82	24	20	915	572
Multi-class	431	269	97	81	7	7	928	495

and comparative analysis prove that the proposed *chi-rev* method with random forest model outperforms all other ML models and achieves a milestone of 99.9% accuracy in each area of attack for binary and multi-class classification. The proposed method is able to achieve a reduced feature set size of 40 features, resulting in almost 51% feature reduction and up-to 50% reduction in the training time.

VII. CONCLUSION AND FUTURE DIRECTIONS

In this work, a novel *chi-rev* method is proposed for feature reduction and is tuned with a random forest classifier to optimize the classifier's performance. Different ML models are tested and evaluated using the CICIDS-2017 data set. The proposed model, which utilizes the random forest classifier and the proposed *chi-rev*, is able to outperform most of the state-of-the-art methods in terms of accurate detection of network attacks in communication networks. Comparative analysis proves the dexterity of the proposed method, which is able to achieve an accuracy of 99.90%, combined with an almost 51% feature reduction and a 50% reduction in training time as compared to the state-of-the-art methods. While the proposed method achieves significantly improved performance, we continue our research to improve feature reduction and optimize computational time. With accuracy and computational overhead as the focus, we aim to explore transformer-based solutions to optimise the performance of the proposed method in the future.

REFERENCES

[1] 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.

[2] K. Peng, V. C. M. Leung, L. Zheng, S. Wang, C. Huang, and T. Lin, "Intrusion detection system based on decision tree over big data in fog environment," *Wireless Commun. Mobile Comput.*, vol. 2018, pp. 1–10, Mar. 2018, doi: 10.1155/2018/4680867.

[3] P. Pirozmand, M. A. Ghafary, S. Siadat, and J. Ren, "Intrusion detection into cloud-fog-based IoT networks using game theory," *Wireless Commun. Mobile Comput.*, vol. 2020, pp. 1–9, Nov. 2020, doi: 10.1155/2020/8819545.

[4] V. K. Singh, B. Nathani, and M. Kumar, "WEED-MC: Wavelet transform for energy efficient data gathering and matrix completion," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 5, pp. 1066–1073, May 2020.

[5] V. K. Singh, M. Kumar, and S. Verma, "Node scheduling and compressed sampling for event reporting in WSNs," *IEEE Trans. Netw. Sci. Eng.*, vol. 6, no. 3, pp. 418–431, Jul. 2019.

[6] V. K. Singh, M. Kumar, and S. Verma, "Accurate detection of important events in WSNs," *IEEE Syst. J.*, vol. 13, no. 1, pp. 248–257, Mar. 2019.

[7] A. Shivhare, V. K. Singh, and M. Kumar, "Anticomplementary triangles for efficient coverage in sensor network-based IoT," *IEEE Syst. J.*, vol. 14, no. 4, pp. 4854–4863, Dec. 2020.

[8] V. K. Singh, C. Singh, and H. Raza, "Event classification and intensity discrimination for forest fire inference with IoT," *IEEE Sensors J.*, vol. 22, no. 9, pp. 8869–8880, May 2022.

[9] R. G. Mantovani, A. L. D. Rossi, J. Vanschoren, B. Bischl, and A. C. P. L. F. de Carvalho, "Effectiveness of random search in SVM hyper-parameter tuning," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2015, pp. 1–8.

[10] R. Abdulhammed, H. Musafer, A. Alessa, M. Faezipour, and A. Abuzneid, "Features dimensionality reduction approaches for machine learning based network intrusion detection," *Electronics*, vol. 8, no. 3, p. 322, Mar. 2019, doi: 10.3390/electronics8030322.

[11] R. A. Welikala, M. M. Fraz, J. Dehmeshki, A. Hoppe, V. Tah, S. Mann, T. H. Williamson, and S. A. Barman, "Genetic algorithm based feature selection combined with dual classification for the automated detection of proliferative diabetic retinopathy," *Computerized Med. Imag. Graph.*, vol. 43, pp. 64–77, Jul. 2015, doi: 10.1016/j.compmedimag.2015.03.003.

[12] W. Lian, G. Nie, B. Jia, D. Shi, Q. Fan, and Y. Liang, "An intrusion detection method based on decision tree-recursive feature elimination in ensemble learning," *Math. Problems Eng.*, vol. 2020, pp. 1–15, Nov. 2020, doi: 10.1155/2020/2835023.
[13] A. Alazab, M. Hobbs, J. Abawajy, and M. Alazab, "Using feature selection for intrusion detection system," in *Proc. Int. Symp. Commun. Inf. Technol. (ISCIT)*, Piscataway, NJ, USA, Oct. 2012, pp. 296–301, doi: 10.1109/ISCIT.2012.6380910.
[14] M. Aljanabi and M. A. Ismail, "Improved intrusion detection algorithm based on TLBO and GA algorithms," *Int. Arab J. Inf. Technol.*, vol. 18, pp. 170–179, Mar. 2021, doi: 10.34028/iajit/18/2/5.
[15] A. Yulianto, P. Sukarno, and N. A. Suwastika, "Improving AdaBoost-based intrusion detection system (IDS) performance on CIC IDS 2017 dataset," *J. Phys., Conf. Ser.*, vol. 1192, Mar. 2019, Art. no. 012018.
[16] A. A. Abdulrahman and M. K. Ibrahem, "Evaluation of DDoS attacks detection in a new intrusion dataset based on classification algorithms," *Iraqi J. Inf. Commun. Technol.*, vol. 1, no. 3, pp. 49–55, Feb. 2019.
[17] S. U. Jan, S. Ahmed, V. Shakhov, and I. Koo, "Toward a lightweight intrusion detection system for the Internet of Things," *IEEE Access*, vol. 7, pp. 42450–42471, 2019.
[18] H. Zhang, L. Huang, C. Q. Wu, and Z. Li, "An effective convolutional neural network based on SMOTE and Gaussian mixture model for intrusion detection in imbalanced dataset," *Comput. Netw.*, vol. 177, Aug. 2020, Art. no. 107315.
[19] H. Musafer, A. Abuzneid, M. Faezipour, and A. Mahmood, "An enhanced design of sparse autoencoder for latent features extraction based on trigonometric simplexes for network intrusion detection systems," *Electronics*, vol. 9, no. 2, p. 259, Feb. 2020.
[20] S. Das, S. Saha, A. T. Priyoti, E. K. Roy, F. T. Sheldon, A. Haque, and S. Shiva, "Network intrusion detection and comparative analysis using ensemble machine learning and feature selection," *IEEE Trans. Netw. Service Manage.*, vol. 19, no. 4, pp. 4821–4833, Dec. 2022.
[21] (2017). *CICIDS Dataset Description*. [Online]. Available: https://www.unb.ca/cic/datasets/ids-2017.html
[22] A. Gharib, I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "An evaluation framework for intrusion detection dataset," in *Proc. Int. Conf. Inf. Sci. Secur. (ICISS)*, Dec. 2016, pp. 1–6.
[23] S. Thaseen and A. K. Cherukuri, "Intrusion detection model using chi square feature selection and modified Naïve Bayes classifier," in *Proc. 3rd Int. Symp. Big Data Cloud Comput. Challenges (ISBCC) Smart Innov., Syst. Technol.*, vol. 49, 2016, pp. 81–91.

GAURAV TRIPATHI received the bachelor's degree in information technology and the master's degree in computer science and engineering from SLIET, Longowal, India, in 2013 and 2018, respectively. He is currently pursuing the Ph.D. degree with the Indian Institute of Information Technology, Lucknow, India. His research interests include machine learning, the Internet of Things, and data analytics.

VISHAL KRISHNA SINGH received the bachelor's degree in information technology, the master's degree in computer technology and application, and the Ph.D. degree in information technology from the Indian Institute of Information Technology, Allahabad, India, in 2010, 2013, and 2018, respectively. He is currently a Lecturer and is associated with the Networks and Communications Research Group, School of Computer Science and Electronics Engineering, University of Essex, Colchester, U.K. His research interests include the Internet of Things, wireless sensor networks, in-network inference, machine learning, and data analytics.

VARUN SHARMA received the bachelor's and master's degrees in commerce and the Ph.D. degree in economic and financial management from Rajasthan University, Jaipur, in 2008, 2010, and 2019, respectively. He is currently an Assistant Professor with the Indian Institute of Information Technology, Lucknow, India. His research interests include data mining, data analytics, machine learning, algotrading, financial management, SMEs, and FinTech. He has qualified UGC-NET, in 2012. He received the Gold Medal during the master's study.

MAJITHIA VIVEK VINODBHAI received the bachelor's and master's degrees in computer engineering from the Indian Institute of Information Technology, Lucknow, in 2018 and 2022, respectively. He is currently a Systems Engineer with TCS Innovation Laboratories. His research interests include machine learning, data science, and deep learning.