International Journal of Computational Intelligence Systems (2023) 16:31
https://doi.org/10.1007/s44196-023-00205-w

RESEARCH ARTICLE

A Multi-level Random Forest Model-Based Intrusion Detection Using Fuzzy Inference System for Internet of Things Networks

Joseph Bamidele Awotunde[1] · Femi Emmanuel Ayo[2] · Ranjit Panigrahi[3] · Amik Garg[4] · Akash Kumar Bhoi[4,5,6,7] · Paolo Barsocchi[6]

Received: 19 October 2022 / Accepted: 20 February 2023 / Published online: 12 March 2023

Abstract
Intrusion detection (ID) methods are security frameworks designed to safeguard network information systems. The strength of an intrusion detection method is dependent on the robustness of the feature selection method. This study developed a multi-level random forest algorithm for intrusion detection using a fuzzy inference system. The strengths of the filter and wrapper approaches are combined in this work to create a more advanced multi-level feature selection technique, which strengthens network security. The first stage of the multi-level feature selection is the filter method using a correlation-based feature selection to select essential features based on the multi-collinearity in the data. The correlation-based feature selection used a genetic search method to choose the best features from the feature set. The genetic search algorithm assesses the merits of each attribute, which then delivers the characteristics with the highest fitness values for selection. A rule assessment has also been used to determine whether two feature subsets have the same fitness value, which ultimately returns the feature subset with the fewest features. The second stage is a wrapper method based on the sequential forward selection method to further select top features based on the accuracy of the baseline classifier. The selected top features serve as input into the random forest algorithm for detecting intrusions. Finally, fuzzy logic was used to classify intrusions as either normal, low, medium, or high to reduce misclassification. When the developed intrusion method was compared to other existing models using the same dataset, the results revealed a higher accuracy, precision, sensitivity, specificity, and F1-score of 99.46%, 99.46%, 99.46%, 93.86%, and 99.46%, respectively. The classification of attacks using the fuzzy inference system also indicates that the developed method can correctly classify attacks with reduced misclassification. The use of a multi-level feature selection method to leverage the advantages of filter and wrapper feature selection methods and fuzzy logic for intrusion classification makes this study unique.

Keywords Intrusion detection · Multi-level feature selection · Random forest · Fuzzy inference system · NSL-KDD dataset

✉ Ranjit Panigrahi
ranjit.panigrahi@gmail.com

✉ Akash Kumar Bhoi
akashkrbhoi@gmail.com

✉ Paolo Barsocchi
paolo.barsocchi@isti.cnr.it

Joseph Bamidele Awotunde
awotunde.jb@unilorin.edu.ng

Femi Emmanuel Ayo
ayo.femi@oouagoiwoye.edu.ng

Amik Garg
amikkgarg@gmail.com

1 Department of Computer Science, Faculty of Information and Communication Sciences, University of Ilorin, Ilorin 240003, Nigeria

2 Department of Mathematical Sciences, Olabisi Onabanjo University, Ago-Iwoye, Ogun State, Nigeria

3 Department of Computer Applications, Sikkim Manipal Institute of Technology, Sikkim Manipal University, Majitar, Sikkim 737136, India

4 KIET Group of Institutions, Ghaziabad, Delhi-NCR 201206, India

5 Directorate of Research, Sikkim Manipal University, Gangtok, Sikkim 737102, India

6 Institute of Information Science and Technologies, National Research Council, 56124 Pisa, Italy

7 eSupport for Research, Burla, Sambalpur 768018, India

Abbreviations

AIDS	Anomaly-based intrusion detection system
BIDS	Behavior-based intrusion detection system
CFS-SFS	Feature selection and sequential forward selection
DDoS	Distributed denial-of-service
DM	Data mining
DoS	Denial of service
FCBF	Fast-based correlation features
FCBFiP	Fast-based correlation features in pieces
FNs	False negatives
FOCUS	Fog computing-based security
FPs	False positives
GA	Genetic search
GSA	Genetic search algorithm
HIDS	Host-based intrusion detection system
ID	Intrusion detection
IDSs	Intrusion detection systems
IoT	Internet of things
KIDS	Knowledge-based intrusion detection system
MCFS	Multi-cluster feature selection
MIDS	Misuse-based intrusion detection system
MITM	Man in the middle
ML	Machine learning
NIDS	Network-based intrusion detection system
Pr	Probing attack
R2L	Remote-to-local
RF	Random forest
SFS	Sequential forward selection
SIDS	Signature-based intrusion detection system
U2R	User-to-root
VPN	Virtual private network

1 Introduction

The internet has formed an integral part of people's daily lives, and a great deal of data must be protected from cybercrimes [1]. The importance of the internet to daily human activities has caused attackers to perpetuate many cybercrime activities. The attackers are always looking for new methods to steal users' confidential data by exploiting a vulnerability in the computer networks. The main goal of data security is to develop security models to ensure data confidentiality, integrity, and availability on networks [2, 3]. The motivation to prevent security breaches against information systems on networks has prompted researchers to develop security models capable of detecting intrusions. Developing intrusion detection systems (IDSs) aims to distinguish between intrusion and normal attacks.

IDSs are security frameworks designed to protect information systems on networks. IDS can be classified based on their environments [4] and detection mechanisms [5–7]. IDSs are further classified based on their environment as host-based IDS (HIDS) and network-based IDS (NIDS) [8]. Host-based IDS can be defined as the IDS designed to detect attacks and vulnerabilities on the host computer. On the other hand, network-based IDS monitors the whole network boundary to identify intrusive traffic on the network before penetrating the host computers. Based on their detection mechanisms, IDSs are further classified as signature-based IDS (SIDS) and anomaly-based IDS (AIDS). Signature-based IDS, also known as knowledge-based intrusion detection system **(**KIDS**)** or misuse-based intrusion detection system **(**MIDS**),** compare incoming patterns with known attack database to detect deviations from known attack patterns. The false alarm rate for SIDS is extremely low but cannot detect new and unknown attacks. Therefore, researchers are focusing more on anomaly detection. Anomaly-based IDS, sometimes known as behavior-based intrusion detection systems **(**BIDS**),** is a better method for detecting unknown attacks by recognizing deviations of incoming patterns with a normal profile behavior. The advantage of AIDS is the ability to detect new attacks from any deviations from normal patterns, but one major drawback is the possibility of a high false alarm rate [9].

Several IDS which have been developed are still susceptible to attacks [10, 11]. Although many machine learning (ML) algorithms have been deployed for IDS to increase detection accuracy, existing IDS methods continue to struggle to achieve good results [9]. Researchers have affirmed the importance of feature selection methods in IDSs to increase detection accuracy. Feature selection is a method that can be used during the preprocessing stage to improve the accuracy of the base classifiers. The filter method is a more popular method for feature selection based on the dataset analysis for choosing the most important features without considering the base classifier performance. On the other hand, the wrapper method considers the base classifier performance for choosing the best feature subsets to increase classifier accuracy. The embedded method is considered relatively close to the wrapper method because it uses essential functions and considers the classifier performance for the choice of a better feature subset. The filter method has a faster processing time than the other feature selection methods with slow processing time.

Data mining algorithms have been frequently applied in implementing IDSs [12]. However, many developed data mining methods are either single or hybrid. This study developed a multi-level random forest algorithm for intrusion detection using a fuzzy inference system. The developed IDS method uses correlation-based feature selection and sequential forward selection (CFS-SFS) for the multi-level selection of features. The first phase of the multi-level features selection used a method called correlation-based feature selection to filter out irrelevant features.

The output of the filtered features serves as input to the sequential forward feature selection, a wrapper method for selecting the most relevant features. The selected relevant features finally serve as input to the random forest classifier for intrusions detection. To know the severity level of a detected intrusion and prevent misclassification, fuzzy logic was used to classify an intrusion as either small, medium-small, medium-large, or large.

1.1 Motivations

Three forms of feature selection methods are available they are (i) wrapper, (ii) filtering, and (iii) embedding [13]. Techniques like filtering and embedding are found in different selection methods. While the latter employs both strategies, the first two are distinct. The wrapper technique relies on the accuracy of a predetermined learning algorithm in resolving a specific issue. The chosen features are evaluated based on their performance. There are two steps in the wrapper approach. It first looks for a subset of features, and then, in a subsequent phase, it uses the learning algorithm, which functions as a black box, to evaluate the features that have been chosen. These stages are repeated iteratively until a predetermined stopping criterion is satisfied. The wrapper technique has a problem because the search space for any n features is 2^n, it poses a problem for datasets with enormous dimensions. Different approaches have been developed to address the high-dimensional issue, such as best-first search, hill-climbing, and branch-and-bound search. Genetic algorithms can also improve the locally optimal training performance. The feature selection filtering techniques are separate from the learning algorithms and are more effective than wrapper approaches. However, the chosen features may not be the best due to a lack of a specific learning algorithm. Therefore, the filter methods are divided into two steps. A set of ranking criteria is used to rank the features first. Each feature can be ranked separately using a univariate feature ranking algorithm, or it can be multivariate, with a batch ranking of multiple features. The second stage extracts the characteristics using the aforementioned rating criteria [14].

This study addresses the issue of selecting a set of unique attributes that can improve an IDS's classification accuracy. When a class label is present, the feature selection technique is simplified by calculating each feature's impact on the class label's predictions. Even when class labels are present, feature selection can be done unsupervised (supervised learning). This strategy ensures that the feature subset of the original attributes contains the optimal number of features and is more accurate at detection. The most critical stage is choosing the best representative features using ML-based algorithms. By eliminating redundant and irrelevant characteristics, dimensionality reduction is used to minimize the number of features. However, the calculation cost is higher when determining the ideal number of features from data with many features. This work employs a genetic search algorithm (GSA) to find the best features to improve classification performance. This task involved fine-tuning parameters while the GSA for the feature optimization problem was being implemented. The current approach also creates a novel fitness function for the task at hand. The GSA quickly converges and offers the best features with higher prediction accuracy when the settings are fine tuned.

1.2 Contributions

The study key contributions are as follows:

(a) The design of a multi-level feature selection method to combine the advantages of the filter and wrapper feature selection methods, using the best features chosen, created the hybrid GSA model to train ML classifiers.

(b) The use of a random forest classifier to improve detection accuracy

(c) The design of a fuzzy logic model for intrusion classification reduces the likelihood of misclassification.

(d) The proposed model's effectiveness is compared with cutting-edge intrusion detection systems and traditional feature selection approaches.

2 Related Work

In cybersecurity, ML is critical for detecting malicious and intrusive traffic. In other words, ML algorithms are frequently used in Internet of Things (IoT) risk management to identify IoT traffic. However, due to poor feature selection, ML approaches misclassify a wide range of malicious traffic in a secure IoT network. Therefore, selecting a feature set with enough data to identify smart IoT anomalies accurately and intrusion traffic is critical to solving the problem. This section discusses a few studies on IoT anomalies and intrusion attacks. In addition, several studies have demonstrated the effectiveness of feature selection techniques in the field of network security.

Anomaly and intrusion detection in IoT networks have received a lot of attention in recent years, and experts are working hard to find a solution [11]. Various types of cybersecurity solutions are suggested and used in an IoT network platform to protect computers and IoT applications from attacks and unauthorized access [15–19]. In 2017, for example, IoT distributed denial-of-service (DDoS) attacks increased by up to 172% [20, 21]. Similarly, when compared to 2013, the number of malicious attacks in 2017 has increased several times, with the vast majority of their attacks, such as Botnet attacks and others, being quite dangerous, according to a Kaspersky lab study [22]. Anderson proposed the first intrusion detection system in 1980 to combat the issue of cyberattacks [23]. The authors in [24] then presented a real-time intrusion detection expert systems paradigm, which was able to identify breaches, intrusions, leaks, Trojan horses, and other threats. However, their model employed assumptions to find malicious network attacks. Additionally, their analysis placed a particular emphasis on user activity to detect irregular processes. The man-in-the-middle (MITM) vulnerabilities have suddenly worsened thanks to DDoS [25]; however, these pose a significant danger to the IoT, and other researchers work hard to precisely identify, detect, and implement a plan to safeguard IoT networks against such dangerous intrusions.

Similarly, a novel approach called fog computing-based security (FOCUS) was unveiled in 2018 by authors in [26]. This technique is mainly employed to protect the IoT network against malware-based intrusions. The virtual private network (VPN), in their suggested concept, is employed to protect IoT communication pathways and channels [27, 28]. Additionally, in an IoT network context, their proposed security system can transmit notifications throughout DDoS attacks [29, 30]. Their study validated proof of concept for results evaluation, and they experimented on the proposed model to test the system's effectiveness. However, their experimental findings demonstrated that the suggested approach effectively percolates harmful attempts with a slight reduction in response time and bandwidth utilization.

The feature selection method is crucial and indispensable during data processing. However, feature selection entails choosing useful features from many attributes and eliminating unnecessary ones that do not offer identification-related information. In this regard, in [31], the authors reviewed some effective feature selection techniques based on correlation measurement techniques. They created a new method for the fast-based correlation features (FCBF) algorithm's functions to improve industrial IoT network capabilities. However, they convert the FCBF technique into the fast-based correlation features in pieces (FCBFiP) method for their experiments. The main goal was to partition the feature space into equal-sized segments. They suggested this strategy and enhanced the correlation and ML models running on each node. However, their suggested model performs better regarding model accuracy and throughput. Authors in [32] created a novel technique for identifying attacks coming from IoT devices, suggested an anomaly identification approach that extracts system performance, evaluated it experimentally, and used autoencoders to identify unusual network traffic coming from IoT devices. However, they utilized two well-known IoT-based botnet assaults to evaluate the suggested strategy, and some business devices in the IoT network were compromised by Mirai as well. Their suggested method can detect attacks on IoT devices, according to experimental findings. Similarly, authors in [33] proposed a features selection strategy to improve the functionality of IoT anomaly detection hardware. The data correlation variation between the IoT sensors was monitored in real time to detect identical deployed sensors, and the sensors with the highest correlation variances were selected as the features for anomaly classification. They investigated the window size for data calculation and clustering using curve alignment. Multi-cluster feature selection (MCFS) was then used to select the online feature selection scenario. They demonstrated that the proposed method effectively reduces the false negative (FN) rate of detecting IoT infrastructure anomalies.

In addition to the previously mentioned security technologies, for instance, attack [34, 35], and crucial management [36], the management of evidence [35] can be utilized for IoT security as well. However, in the literature review above, it was clear that finding a reliable and consistent feature set for anomaly and intrusion detection to classify IoT network data is crucial. The attributes selection method's main notion entails four critical steps. The subset generation, which produces a feature set; evaluation of the subset, in which the features are assessed through analysis; decision-making process, where decisions are made to either approve or reject a feature according to specific guidelines and subset validation.

2.1 Feature Selection

Feature selection methods are variable reduction techniques that can convert features from a high-dimensional space to a low-dimensional space while maintaining the classification algorithms' efficiency [37]. In another word, feature selection is the extraction of the best features required for the development of a classifier with high detection accuracy and low false alarm. The goal of feature selection methods is to remove uncorrelated variables from the set of features while keeping the data useful to the classification model. Handling Big Data for ML is required in most fields today, including cybersecurity. Security data is proliferating, and intelligent and efficient management is required [25]. Data mining (DM) and machine learning (ML) techniques for high-dimensional datasets focus on generating relevant insights by minimizing dataset features. The dimensionality constraint is the primary issue that must be addressed to implement DM and ML techniques [25] A “dimensionality constraint” is data that is dispersed in high-dimensional space. This harms low-dimensional space learning methods [38]. Another issue is overfitting, which reduces the accuracy of the ML model when the data contains a large number of characteristics. A high feature count also generates a higher memory and computational cost [39]. The best solution to the high dimensionality problem is to reduce the dimensions of a given dataset using state-of-the-art feature reduction techniques. Feature selection can reduce dimensions [40, 41]. This technique converts numerous features of big data into a new, low-dimensional feature space. Feature selection refers to selecting the most appropriate feature subset from the provided input feature vector to assist the ML model train effectively.

In the real world, the datasets have noise that adds unnecessary and redundant characteristics. Eliminating the noise from the data helps accelerate the learning process, thus improving the classifier’s classification performance while lowering the false positives (FPs) and FNs [42]. There are two types of feature selection techniques: supervised and unsupervised. Supervised feature selection methods are typically created to solve classification or regression issues [43]**.** These strategies are used to separate the feature subset from the original features provided to estimate the targets in a regression analysis or to be able to discriminate between the classes of data that are accessible [44].

2.1.1 Genetic Search

A search method based on a genetic algorithm is called a genetic search (GA). Using computers to imitate the process of natural evolution served as the inspiration for GA. The GA was initially proposed as an ML algorithm by authors in [45]. The algorithm is an iterative one that normally starts with an initial population of random individual programs. The evaluation of their fitness measures determines the best individual programs in the population. Every iteration results in the next population of the fittest individuals, thanks to computerized genetic recombination and mixing.

2.1.2 Correlation-Based Feature Selection

Correlation-based feature selection (CFS) is a filter-based feature selection method that selects features based on their correlation with the class. The feature–class relationship is evaluated and the relationship with the highest correlation is chosen for selection. Based on this feature evaluation, the GSA assesses each characteristic’s attributes and creates elements with the best fitness value. If two feature subsets have the same fitness values, the genetic search additionally employs rule evaluation to return the feature subset with the fewest number of subgroups.

2.1.3 Sequential Forward Selection

Sequential forward selection (SFS) is a wrapper method that performs a bottom-to-up search process. The SFS starts from an empty set and sequentially adds features from the full feature set with already selected features that result in the highest classifier accuracy.

2.2 Random Forest Algorithm

Random forest (RF) is an ensemble method of decision trees. The algorithm works by producing a different number of decision trees from different samples and takes their majority vote for the classification decision. The benefit of RF is that increased precision can be attained without the risk of overfitting.

2.3 Fuzzy Logic

Fuzzy logic is described as a multi-valued algebra in which the truth values are all intermediate values between 0 and 1, inclusive [46]. Fuzzy logic has become a popular application for problems relating to uncertainty and classification [47–50]. The advantage of using fuzzy logic for intrusion detection is that one can capture the overlapping severity grades of intrusions.

3 Methodology

This study developed a multi-level random forest algorithm for intrusion detection using a fuzzy inference system (ML-RFID-FIS). The developed ML-RFID-FIS is divided into four major phases: dataset preprocessing, feature selection, detection, and classification. The dataset preprocessing phase involves feature encoding and normalization to make the data interpretable and easily understood by the ML model. Training and testing sets were created from the dataset. 80% of the data are in the training set, while the remaining 20% are for the testing set. The feature selection phase used a multi-level feature selection approach to blend the advantages of the filter and wrapper methods. The first stage (filter method) of the multi-level feature selection used correlation-based feature selection to select essential features based on the

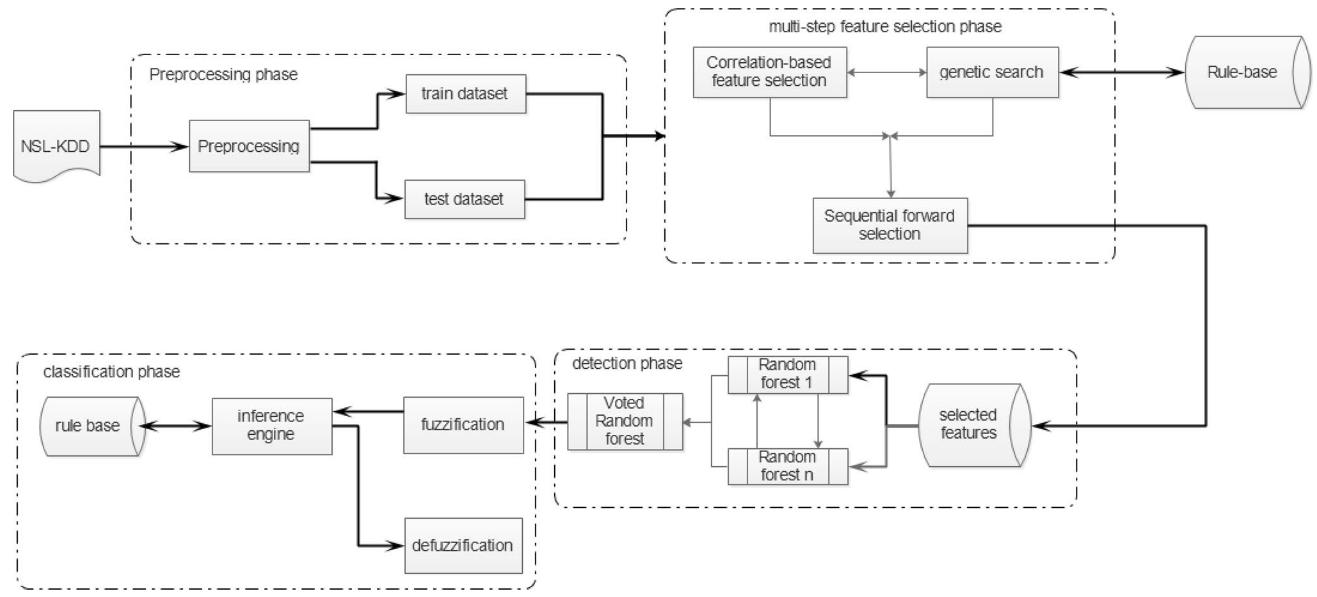


Fig. 1 Architecture for the multi-level random forest for intrusion detection using fuzzy inference system

multi-collinearity in the data. The second stage (wrapper method) used a sequential forward selection method to further select top features based on the accuracy of the baseline classifier. This is because the filter methods are not affected by the classifier, hence the wrapper method. The random forest technique is then used to detect intrusions using the chosen top features. Fuzzy logic was used to classify intrusions as either normal, low, medium, or high to reduce misclassification. Python programming language was used for the implementation. Figure 1 describes the architecture of the developed ML-RFID-FIS.

3.1 Dataset Description

The NSL-KDD dataset was adopted for implementation due to its effectiveness in intrusion detection [51]. NSL-KDD is a dataset that has been proposed as a solution to some of the issues in the existing IDS datasets. The dataset is a perfect representation of real networks. It can be used as a standard benchmark dataset for the design of IDS relating to Internet traffics. The 41 attributes in the dataset are classified as either normal or anomalous. The characteristics can be broken down into three categories: time based (19 features), connection based (9 features), and content based (13 features). Four types of attacks may be distinguished from the dataset: probing, denial of service (DoS), user-to-root (U2R), and remote-to-local (R2L).

(a) Probing attack (Pr): This is the accumulation of system information by testing it to discover vulnerabilities that can be used to compromise it later. Some of the probing attacks are ipsweep, portsweep, nmap, and satan.
(b) Denial of service (DOS): An attack in which the host system is flooded by unwanted messages by the attacker preventing authorized users from gaining access to resources or services. Some examples of DoS attacks are Neptune, Smurf, tear drop, pod, and mail bomb.
(c) User-to-root attack (U2R): The attacker begins the attack using a regular user account to gain entry to the system. The attack then exploits the system flaws to gain access to resources that should normally be unavailable to them. Examples of U2R are buffer overflow, loadmodule, and Perl.
(d) Remote-to-local attack (R2L): This is an intrusion committed by an attacker with authorization to send packets to a machine connected to the network with no identity on that machine. The attacker exploits some system flaws to attain remote access to the machine as a user. Examples are Guess_passwd, imap, and spy.

Table 1 Summary of the dataset

Dataset	Number of instances
TR1	6175 (65% of TR)
TR2	3325 (35% of TR)
TE	4500

3.2 Dataset Preprocessing Phase

The training data (TR) for the implementation consists of 9500 randomly chosen records from the NSL-KDD training file, while the testing data (TE) comprises 4500 randomly chosen records from the NSL-KDD test file. Additionally, numeric encoding is used for symbolic properties (such as protocol type, service, flag, and class). The TR is split into two partitions: TR1 and TR2 of 65% and 35%, respectively. Table 1 shows the summary of the dataset.

Data normalization was done to scale the data because of the variation in the units and magnitude of the data. The MinMaxScaler method was used to reduce the data between 0 and 1 as shown in Eq. 1:

$$\text{Xnorm} = \frac{\text{X} - \text{Xmin}}{\text{Xmax} - \text{Xmin}} \tag{1}$$

where Xnorm is the normalized score; Xmax and Xmin are the highest and lowest values of the feature in the data, respectively.

3.3 Feature Selection Phase

The feature selection phase used a multi-level feature selection approach to blend the advantages of the filter and wrapper methods. The first stage (filter method) of the multi-level feature selection used correlation-based feature selection to select essential features based on the multi-collinearity in the data. The association between each attribute and the class is computed as part of the correlation-based feature selection. A feature V_i is said to be relevant to the class if and only if there exists some v_i and c for which $p\left(V_i = v_i\right) > 0$ as described in Eq. 2:

$$p\left(C = c | V_i = v_i\right) \neq p(C = c) \tag{2}$$

where C denotes a given class, c denotes class subsets, V_i denotes a candidate feature, and v_i denotes the feature subsets.

The correlation between each attribute and class can be predicted as in Eq. 3:

$$r_{zc} = \frac{k\overline{r_{zi}}}{\sqrt{k + k(k-1)\overline{r_{ii}}}} \tag{3}$$

where k is the number of components and r_{zc} is the correlation between the summed components and the outside variable. The average correlation between the components and the external variable is denoted by r_{zi}, and the average correlation between the components is denoted by r_{ii}.

Each attribute-class association was evaluated using the genetic search technique, and it returns the chosen characteristics with the highest fitness value (Eq. 4). A rule-based strategy was designed to return the feature subset with the least number of features having the same fitness as those feature subsets having more features. In other words, if the fitness value of two feature subsets is equal, the feature subset with the fewest features is what a rule evaluator returns.

$$\mathit{fitness}(X) = \frac{3}{4}A + \frac{1}{4} = \left(1 - \frac{S + F}{2}\right) \tag{4}$$

where X is a feature subset, A is the average cross-validation accuracy of the baseline classifier, S is the number of instances or training samples, and F is the number of subset features.

The second stage (wrapper method) of the multi-level feature selection used the sequential forward selection method to further select top features based on the accuracy of the baseline classifier. This is because the filter methods are not affected by the classifier, hence the wrapper method. The sequential forward selection method starts from the empty set (Eq. 5) and sequentially selects the next final best feature X^+ that results in the highest accuracy J(Y_k+X^+) of the baseline classifier when combined with the features Y_k that have already been selected (Eq. 6).

$$Y_o = \{\phi\} \tag{5}$$

where Y_o denotes the empty feature set.

$$X^+ = \underset{X \notin Y_k}{\operatorname{argmax}}\left[J\left(Y_k + X\right)\right] \tag{6}$$

where X^+ denotes the next best feature, Y_k denotes already selected features, X denotes added features, and argmax denotes the highest accuracy.

3.4 Detection Phase

The proposed network intrusion detection phase used a random forest algorithm as the baseline classifier. The selected final features were used as the training dataset for the random forest algorithm. The training dataset was partitioned into TR1 and TR2 of 65% and 35%, respectively. The random forest algorithm builds decision trees on the TR1 and TR2 partitions and takes their majority vote for intrusion detection classification. The algorithms that describe the methodology are as follows:

Algorithm 1 first used the multi-step feature selection to reduce the features into the most optimal feature set. Then the selected optimal features serve as input into algorithm 2.

The detector is a C4.5 decision tree that builds a *tree of rules* to appropriately separate the dataset into its respective classes (intrusion, normal) during the training phase. The input features are categorized as incursion or normal throughout the testing phase using the tree of rules. The C4.5 decision tree was constructed using algorithm 2. Recursively choosing the optimal attribute to divide the data is how the algorithm operates (Step 7) and then growing the tree's leaf nodes (Steps 11 and 12) until the stopping requirement is satisfied (Step 1).

The decision tree is expanded by adding a new node using the *createNode()* function. A decision tree node either has a test condition or class label expressed as a *node. label,* or a test condition, denoted as a *node.test_cond*. For each tree in the forest (step 16), $T_{(i)}$ refers to the *ith* bootstrap, and the procedure chooses a bootstrap sample from T. Then using a decision tree learning algorithm, a decision tree is learned. A randomly chosen subset of the features $f \subseteq F$, where F is the set of features, was chosen at each tree node. The node then divides based on f's best feature rather than F's. F is significantly smaller in practice than f. Sometimes the most computationally demanding part of decision tree learning is choosing which feature to split. Limiting the available functionalities and features, thus, significantly decreasing the computational complexity component of decision tree learning, accelerating the tree's learning process in process.

Algorithm 1: *Multi-step FeatureSelection*

```
Input  $Training\ Dataset\ X\{x_1, x_2, \ldots, x_k \mid xi \in C\}$
Output – $X$ //reduced features
Process begin
    01  Start by creating an initial population P at random
    02  Create a probability distribution p over the P members using
        $p(x)/f(x)$ as the central variable
    03  $x$ and $y$ should be chosen as population members concerning $p$.
    04  Apply crossover to $x$ and $y$ to produce new population members $x'$ and $y'$.
    05  Apply mutation to $x'$ and $y'$
    06  Insert $x'$ and $y'$ into P' (the next generation).
    07  If $|P'| < |P|$, goto 2 & 3.
    08  Let $P \leftarrow P'$
    09  Go to generation 6 if there are further generations to process.
    10  Return the value of $x \in P$ where $f(x)$ is highest.
    11  If the fitness value of two feature subsets is the same
    12  //select the feature subset with the fewest number of feature subsets.
    13  $Y \leftarrow X(x_1, x_2, \ldots, x_k)$
    14  $Y_o = \{\emptyset\}$
    15  $X^+ = \underset{X \notin Y_k}{\operatorname{argmax}}[J(Y_k + X)]$
    16  $Y_{k+1} = Y_k + X^+, k = k + 1$ //update
    17  return – $X$
    end
```

Algorithm 2: BuildClassifier

```
Input    A training set $T := (x_1, y_1), \ldots, (x_n, y_n)$, reduced feature $F$,
         and number of trees in forest $R$
Output   The learned tree $H$
Process  begin
         01 if stopping_cond($X, F$) = true then
         02    leaf = createNode()
         03    leaf.label = Classify($X$)
         04    return leaf
         05 else
         06    root = createNode()
         07    root.test_cond = find_best_split($X, F$)
         08    let $V = \{v | v$ is a possible outcome of root.test_cond$\}$
         09    for each $v \in V$ do
         10        $Xv = \{x$ | root.test_cond($x$) = $v$ and $x \in X\}$
         11        child = buildClassifier($X, F$)
         12        add child as descendant of root and label the edge as $v$
         13    end for
         14 end if
         15 return root
         16 function RandomForest(root, $F$)
         17    $H \leftarrow \emptyset$
         18    for $i \in 1, \ldots, R$ do
         19        $T^{(i)} \leftarrow$ A bootstrap sample from root
         20        $h_i \leftarrow$ RandomizedTreeLearn($T^{(i)}, F$)
         21        $H \leftarrow H \cup \{h_i\}$
         22    end for
         23    return The learned tree
         24 end function
         25 function RandomizedTreeLearn(root, $F$)
         26    At each node:
         27    $f \leftarrow$ very small subset of $F$
         28    Split on best feature in $f$
         29    return $H$
         30 end function
         end
```

3.5 Classification Phase

The following concepts and definitions of fuzzy logic are used in the fuzzy extension to intrusion classification:

3.5.1 Fuzzy Set

The attacks represented by the dataset and categorized into four types make up the fuzzy set (Eq. 7) for the intrusion categorization: probing, denial of service (DoS), user-to-root (U2R), and remote-to-local (R2L). The defined fuzzy set's members are present in varying degrees in 0 and 1:

$$A = \{\text{Probing}, \text{DoS}, \text{U2R}, \text{R2L}\} \tag{7}$$

where A denotes the fuzzy membership set and Probing, DoS, U2R, andR2L denote the fuzzy inputs.

Table 2 Fuzzy value range

Linguistic value	Value range
Normal	$0.1 \le x < 0.3$
Low	$0.3 \le x < 0.6$
Medium	$0.6 \le x < 0.8$
High	$0.8 \le x \le 1.0$

Table 3 Sample rule base

#No	Probing	DoS	U2R	R2L	Intrusion classification (conclude)	Non-zero min no
1	0.25	0.25	0.25	0.25	Normal	0.25
2	0.25	0.5	0.5	0.5	Low	0.25
3	0.25	0.75	0.75	0.75	Medium	0.25
4	0.25	0.9	0.9	0.9	Medium	0.25
5	0.5	0.25	0.25	0.25	Normal	0.25
6	0.5	0.5	0.5	0.5	Low	0.5
7	0.5	0.75	0.75	0.75	Medium	0.5
8	0.5	0.9	0.9	0.9	High	0.5
9	0.75	0.25	0.25	0.25	Normal	0.25
10	0.75	0.5	0.5	0.5	Low	0.5
11	0.75	0.75	0.75	0.75	Medium	0.75
12	0.75	0.9	0.9	0.9	High	0.75
13	0.9	0.25	0.25	0.25	Normal	0.25
14	0.9	0.5	0.5	0.5	Low	0.5
15	0.9	0.75	0.75	0.75	Medium	0.75
16	0.9	0.9	0.9	0.9	High	0.9

3.5.2 Linguistic Variables

According to Eq. 8, the linguistic variables represent the membership level for the specified fuzzy set A. It is employed to demonstrate the level of categorization for a specific class attribute value.

$$m_A(x) = \{normal, low, medium, high\} \tag{8}$$

where $m_A(x)$ denotes the degree of membership for membership set A and normal, low, medium, andhigh denotes the linguistic variables.

3.5.3 Fuzzification

Since the linguistic variables are divided into four grades, Eq. 9 triangle membership function was modified to fit the situation. The crisp values were transformed into fuzzy values by fuzzification. Table 2 displays the fuzzy range of values for the fuzzification procedure.

$$\mu_A(x;[a,b,c]) = \begin{cases} 0, if\ x = a \\ \frac{x-a}{c-a}, if\ x \in [a,c] \\ \\ \frac{b-x}{c-b}, if\ x \in [b,c] \\ 0, if\ x \geq c \end{cases} \tag{9}$$

where x represents the x-coordinate of real values and a, b, c represents the y-coordinate between 0 and 1.

3.5.4 Fuzzy Rules

The rule of thumb defined a total of 16 rules. Given that four linguistic factors were employed, we have $2^4 = 16$ rules. Table 3 displays the rules established with the assistance of subject-matter experts. The modified fuzzy logic evaluated its rules using the AND function by taking the minimum values.

3.5.5 Inference Engine

The idea of the fuzzy rules established on the membership set for intrusion classification is used by the fuzzy inference engine. These fuzzy rules are intended to predict the

Table 4 Attack labeling

Attacks	Data labeling
Normal	0
Denial of service (DoS)	1
Probing	2
Remote-to-local (R2L)	3
User-to-root (U2R)	4

Table 5 List of features

No.	Name of feature	No.	Name of feature	No.	Name of feature
1	Duration	15	Su_attempted	29	Same_srv_rate
2	Protocol_type	16	Num_root	30	Diff_srv_rate
3	Service	17	Num_file_creations	31	Srv_diff_host_rate
4	Flag	18	Num_shells	32	Dst_host_count
5	Src_bytes	19	Num_access_files	33	Dst_host_srv_count
6	Dst_bytes	20	Num_outbound_cmds	34	Dst_host_same_srv_rate
7	Land	21	Is_host_login	35	Dst_host_diff_srv_rate
8	Wrong_fragment	22	Is_guest_login	36	Dst_host_same_src_port_rate
9	Urgent	23	Count	37	Dst_host_srv_diff_host_rate
10	Hot	24	Srv_count	38	Dst_host_serror_rate
11	Num_failed_logins	25	Serror_rate	39	Dst_host_srv_serror_rate
12	Logged_in	26	Srv_serror_rate	40	Dst_host_rerror_rate
13	Num_compromised	27	Rerror_rate	41	Dst_host_srv_rerror_rate
14	Root_shell	28	Srv_rerror_rate	42	Label

Table 6 Category of attack types

Attack types	Category
Back, Land, Neptune, Pod, Smurf, Teardrop, Mailbomb, Apache2, Processtable, Udpstorm, Worm	DoS
Ipsweep, Nmap, Portsweep, Satan, Mscan, Saint	Probing
Ftp_Write, Guess_Passwd, Imap, Multihop, Phf, Spy, Warezclient, Warezmaster, Warezmaster, Sendmail, Named, Snmpgetattack, Snmpguess, Xlock, Xsnoop, Httptunnel	R2L
Buffer_Overflow, Loadmodule, Perl, Rootkit, Ps, Sqlattack, Xterm	U2R

Table 7 Features selected after correlation-based feature selection

1	Dst_Host_Srv_Serror_Rate	7	Is_Guest_Login
2	Same_Srv_Rate	8	Dst_Host_Srv_Diff_Host_Rate
3	Dst_Host_Same_Src_Port_Rate	9	Num_Failed_Logins
4	Count	10	Dst_Host_Serror_Rate
5	Srv_Serror_Rate	11	Serror_Rate
6	Dst_Host_Diff_Srv_Rate	12	Wrong_Fragment

Table 8 Features selected after sequential feature selection

1	'Duration'	7	'dst_host_count'
2	'Service'	8	'dst_host_diff_srv_rate'
3	'Flag'	9	'dst_host_same_src_port_rate'
4	'src_bytes'	10	'dst_host_srv_diff_host_rate'
5	'srv_count',		
6	'diff_srv_rate'		

Table 9 Confusion matrix before feature selection

		Normal	DoS	Probe	U2R	R2L
	Labels	0	1	2	3	4
Normal	0	5565	1766	127	1	0
DoS	1	94	9399	217	0	1
Probe	2	201	953	1266	1	0
U2R	3	1	2878	6	0	0
R2L	4	0	66	0	1	0

severity grade for a particular intrusion class. The fuzzy inference technique used root mean square (RMS) to support its conclusions. The RMS was used to combine the different possibilities of rules that lead to the same conclusion. It calculates the center of gravity by adding all the results from the same firing rules.

Table 10 Rates of attacks before feature selection

	Labels	TP	TN	FP	FN
Normal	0	5565	14788	296	1894
DoS	1	9399	7169	5663	312
Probe	2	1266	19772	344	1155
U2R	3	0	19655	3	2885
R2L	4	0	2247	1	67

The RMS equation is given in Eq. 10:

$$\sqrt{\sum R^2} = \sqrt{R_1^2 + R_2^2 + R_3^2 + \cdots + R_n^2} \tag{10}$$

where $R_1^2 + R_2^2 + R_3^2 + \cdots + R_n^2$ denotes values of several rules in the fuzzy rule base, all leading to the same result.

The classification steps are summarized in Algorithm 3.

Table 11 Evaluation metrics before feature selection of the random forest

	Metrics	Accuracy (%)	Precision (%)	Sensitivity (%)	Specificity (%)	F1-score (%)
Overall	Labels	72.00	72.01	72.00	93.01	72.01
Normal	0	90.29	94.95	74.61	98.04	83.56
DoS	1	73.50	62.40	96.79	55.87	75.88
Probe	2	93.35	78.63	52.29	98.29	62.81
U2R	3	87.19	0.00	0.00	99.98	0.00
R2L	4	99.70	0.00	0.00	100.00	0.00

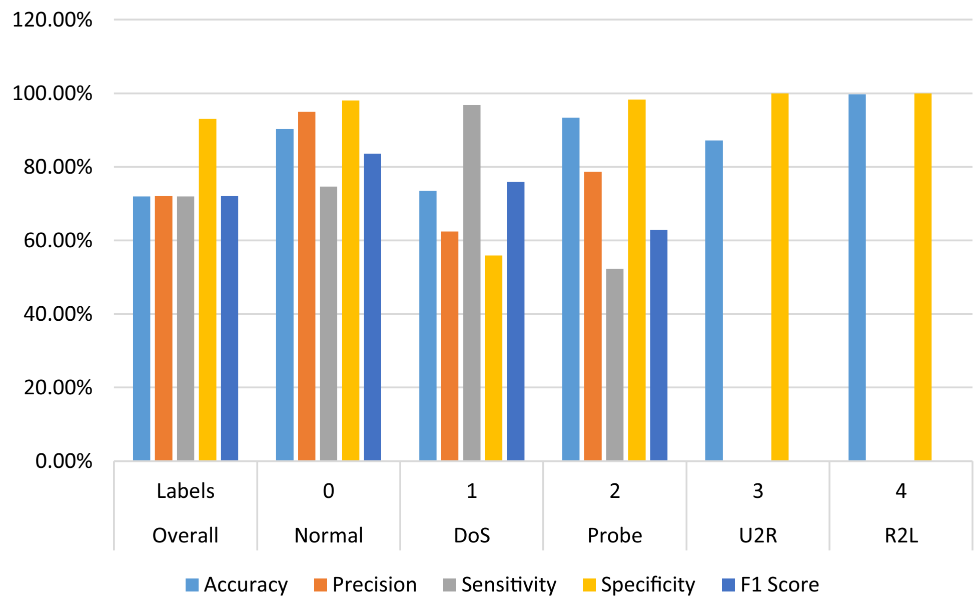


Fig. 2 Model performance on the attack types before feature selection

Table 12 Confusion matrix after feature selection

		Normal	DoS	Probe	U2R	R2L
	Labels	0	1	2	3	4
Normal	0	15389	4	13	29	3
DoS	1	16	10635	1	0	0
Probe	2	27	5	2767	0	0
U2R	3	44	0	0	745	0
R2L	4	15	0	0	4	5

***Algorithm 3:** Classification Algorithm*

```
Input    Intrusion class $D_i$; Fuzzy inputs Probing, DoS, U2R, R2L
Output   Severity level $C_i$
Process  begin
         01  Accept $D_i$ as input
         02  for $i := 1$ to $|D_i|$, where I represent Probing, DoS, U2R or R2L
         03      if $(0.1 \le C_i < 0.3)$ then
         04          $C_i \leftarrow normal$
         05      elseif$(0.3 \le C_i < 0.6)$ then
         06          $C_i \leftarrow low$
         07      elseif $(0.6 \le C_i < 0.8)$ then
         08          $C_i \leftarrow medium$
         09      elseif $(0.8 \le C_i \ge 1)$ then
         10          $C_i \leftarrow high$
         11      end if
         12  end for
         13  return $C_i$
         end
```

4 Experimental Implementation of the Proposed System

4.1 Implementation

The experiment was carried out on the NSL-KDD dataset. The configurations used in carrying out the implementation are intel(R) Core(TM) i5-3230 M CPU @ 2.60 GHz, 2601 MHz, 2 Core(s), 4 Logical Processor(s) @ 2.60 GHz with 4 GB RAM running on 64-bit Windows 10 operating system and Python 3.8. Numpy, Scikit-Learn, and Pandas libraries are among the python packages used. The experimentation for the developed ML-RFID-FIS method was implemented with correlation-based feature selection, sequential forward selection, and a random forest classifier. The NSL-KDD dataset was loaded using panda's library.

Table 13 Rate of attacks after feature selection

	Labels	TP	TN	FP	FN
Normal	0	15389	14,162	102	49
DoS	1	10635	19,041	9	17
Probe	2	2767	26889	14	32
U2R	3	745	28880	33	44
R2L	4	5	29675	3	19

4.2 Dataset Preprocessing

The data were preprocessed and cleaned by checking for null data, invalid values, and transformations like normalization and changing of categorical variables to numerical variables (Table 4). The data had 41 columns, excluding the target column (Table 5). Table 6 shows the attack types and their categories.

4.3 Feature Selection

The correlation-based feature selection, a filter method, was used to extract the first set of relevant features based on their

Table 14 Evaluation metrics after feature selection of the ML-RFID-FIS

	Metrics	Accuracy (%)	Precision (%)	Sensitivity (%)	Specificity (%)	F1-score (%)
Overall	Labels	99.46	99.46	99.46	93.86	99.46
Normal	0	99.49	99.34	99.68	99.28	99.51
DoS	1	99.91	99.92	99.84	99.95	99.88
Probe	2	93.85	99.50	98.86	99.95	99.18
U2R	3	99.74	95.76	94.42	99.89	95.09
R2L	4	99.93	62.50	20.83	99.99	31.25

Table 15 Overall performance comparison of the ML-RFID-FIS

Algorithm	Accuracy (%)	Precision (%)	Sensitivity (%)	Specificity (%)	F1-score (%)
C4.5 tree	92.97	91.34	90.84	93.93	91.14
Decision table	88.22	85.64	85.23	94.85	85.13
Bagging	93.97	93.71	90.83	97.84	92.25
KNN	90.76	88.62	87.87	90.67	88.22
Logistic	92.44	91.81	88.62	97.13	90.24
Naïve Bayes	79.27	66.62	95.13	93.65	78.42
Bayesian Logistic Regression	81.36	91.63	86.51	84.12	89.74
Naïve Bayes Multinomial	79.07	74.13	72.23	84.82	73.17
Multilayer Perceptron	90.64	96.54	79.25	97.87	87.88
Random forest	72.00	72.01	72.00	93.01	72.01
ML-RFID-FIS	99.46	99.46	99.46	93.86	99.46

metrics. The correlation-based feature selection reduced the number of features from 41 to 29, discarding the rest. The features that were dropped are shown in Table 7. Next, the sequential feature selection received the 29 features and selected the relevant features. Finally, the sequential feature selection reduced the 29 features to 10 features, and the features were used for the development of the model. The features selected are shown in Table 8.

4.4 Results and Discussion

To better understand the classification error of the attack types in the dataset, Table 9 displays the confusion matrix before feature selection. Table 9 for the NSL-KDD dataset makes it clear that the constructed model can distinguish between the various assault types correctly. Only 1766 out of the total samples categorized as normal have the wrong classification of DoS, 127 of the total samples categorized as normal had been incorrectly labeled as Probe, and only 1 sample out of the whole number of normal samples is mistakenly labeled as U2R. Similar to this, only 94 out of all samples from a DoS assault are incorrectly labeled as normal, the classification of 217 of the total samples used in the DoS attack as Probe, and one of the total samples from the DoS assault is incorrectly categorized as R2L. Just 201 out of the total samples from the Probe attack have been incorrectly classed as normal, only 953 out of the total samples from the Probe assault have the incorrect DoS classification, and only 1 out of the total samples from the Probe attack is incorrectly categorized as U2R. Only 1 of the total samples from the U2R attack is incorrectly categorized as normal, only 2878 out of the total samples from the U2R attack have the incorrect DoS classification, and just 6 out of the total samples used in the U2R attack are mistakenly labeled as probes. Only 66 out of the total samples from the R2L attack have been incorrectly labeled as DoS, and only 1 of the total samples from the R2L attack is incorrectly categorized as a U2R. It might be said that most mistakes result from misclassifying U2R attacks as DoS attacks.

Table 10 shows the rates of attacks before feature selection. It can be observed that the DoS attack has the highest number of correct classifications of 9,399 compared to the

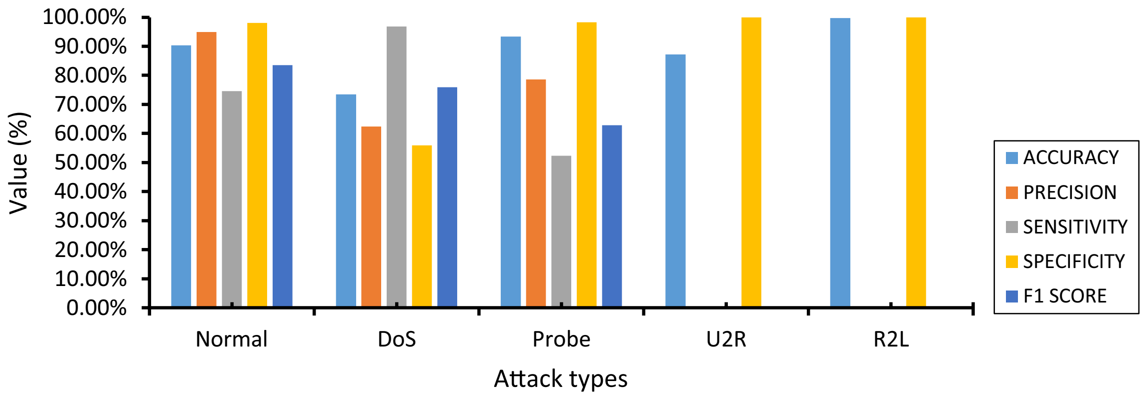


Fig. 3 Graphical representation of model performance on the attack types before feature selection

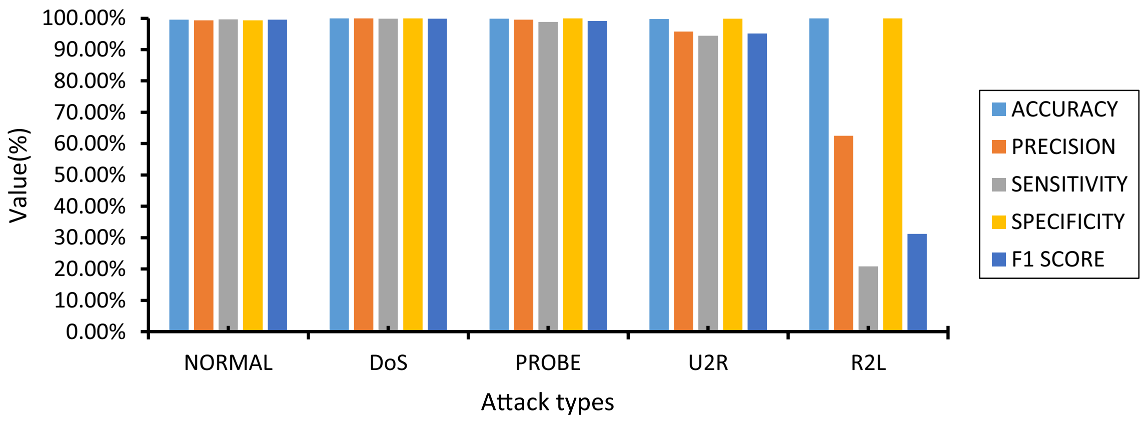


Fig. 4 Graphical representation of model performance on the attack types after feature selection

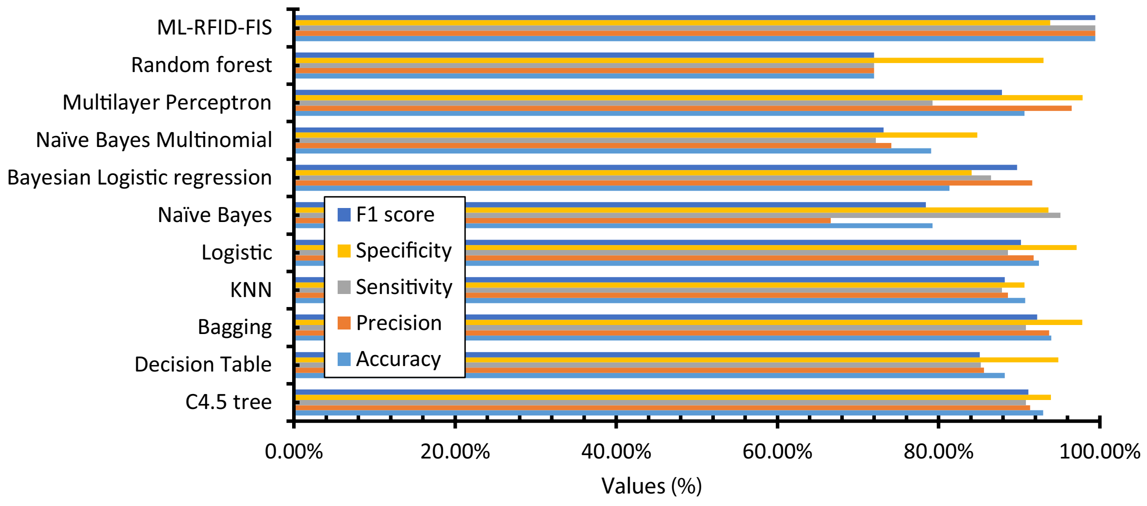
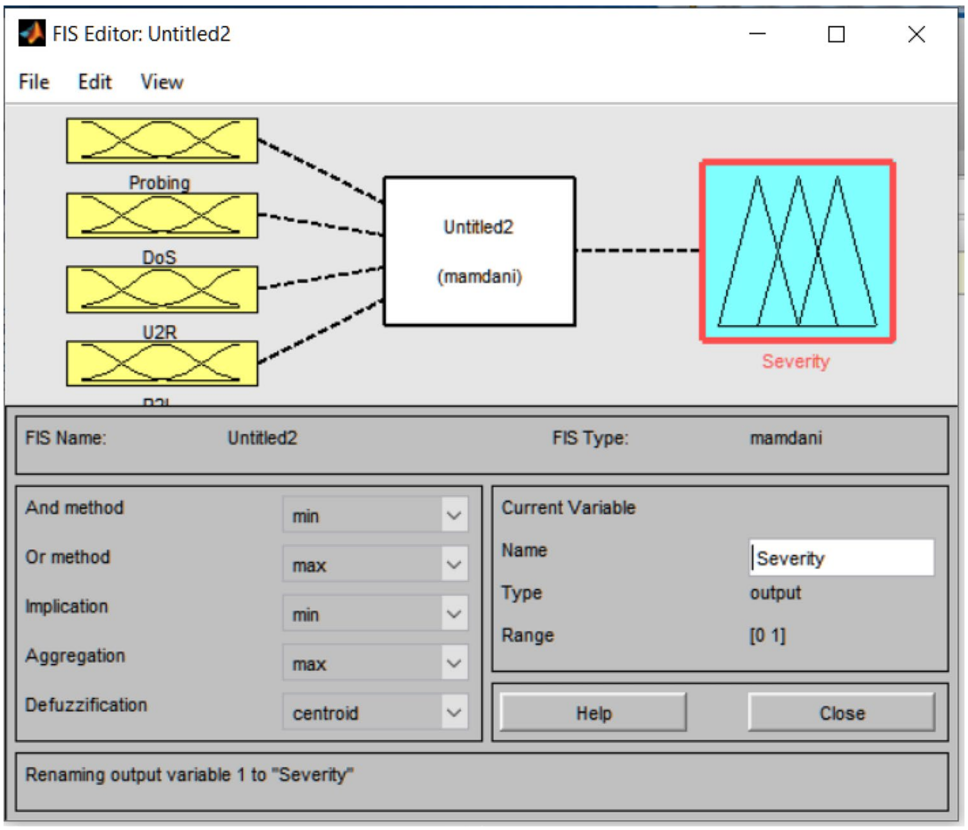


Fig. 5 Overall performance comparison of the ML-RFID-FIS

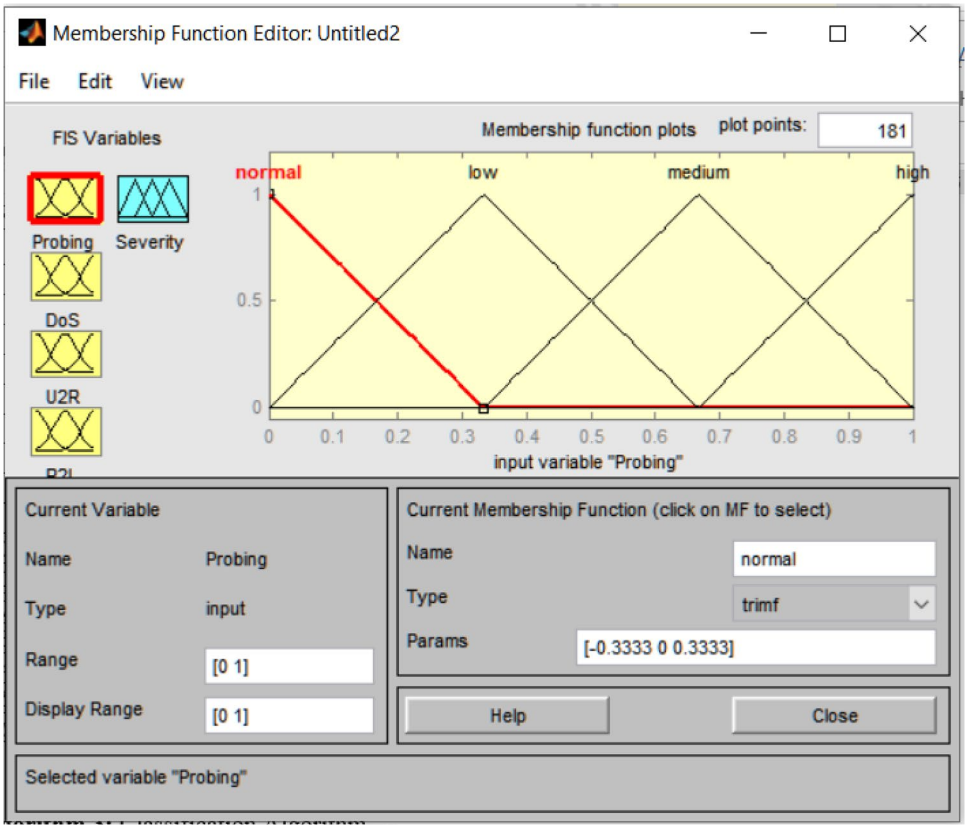
other attack types while R2L has the lowest false alarm rate compared to the other attack types. Table 11 and Fig. 2 show the evaluation metrics before the feature selection of the random forest model. The random forest model has an overall accuracy, precision, sensitivity, specificity, and F1-score of 72.00%, 72.01%, 72.00%, 93.01%, and 72.01%, respectively. These results showed that the accuracy value for intrusion detection is low for the baseline classifier before the feature selection method. The results in Table 11 indicate that an ordinary random forest without the developed multi-level feature selection is unable to correctly differentiate the different attack types in the dataset. The ordinary random forest could not differentiate U2R and R2L attacks with 0.00% and 0.00% F1-score, respectively. The results also suggest poor classification results for Normal, DoS, and Probe attacks with F1-score of 83.56%, 75.88%, and 62.81%, respectively. Therefore, the overall results in Table 11 are a pointer to the need for the developed multi-level feature selection method to enhance the random forest algorithm.

To better understand the classification error of the attack types in the dataset, Table 12 displays the confusion matrix after feature selection. The proposed model can correctly distinguish between the various attack kinds. Only 4 out of the total samples categorized as normal are DoS, 13 of the total samples classified as normal are mistakenly labeled as Probe, only 29 out of the total samples that belong to normal are incorrectly labeled as U2R, and just 3 out of the total samples that correspond to normal are incorrectly labeled as R2L. Similarly to this, just 16 out of all the samples from the DoS assault are incorrectly labeled as normal, and only 1 of the total samples from the DoS attack is mistakenly labeled as a probe. Only 27 out of the total samples from the Probe attack have the incorrect classification, and just 5 out of the total samples from the Probe assault are incorrectly labeled



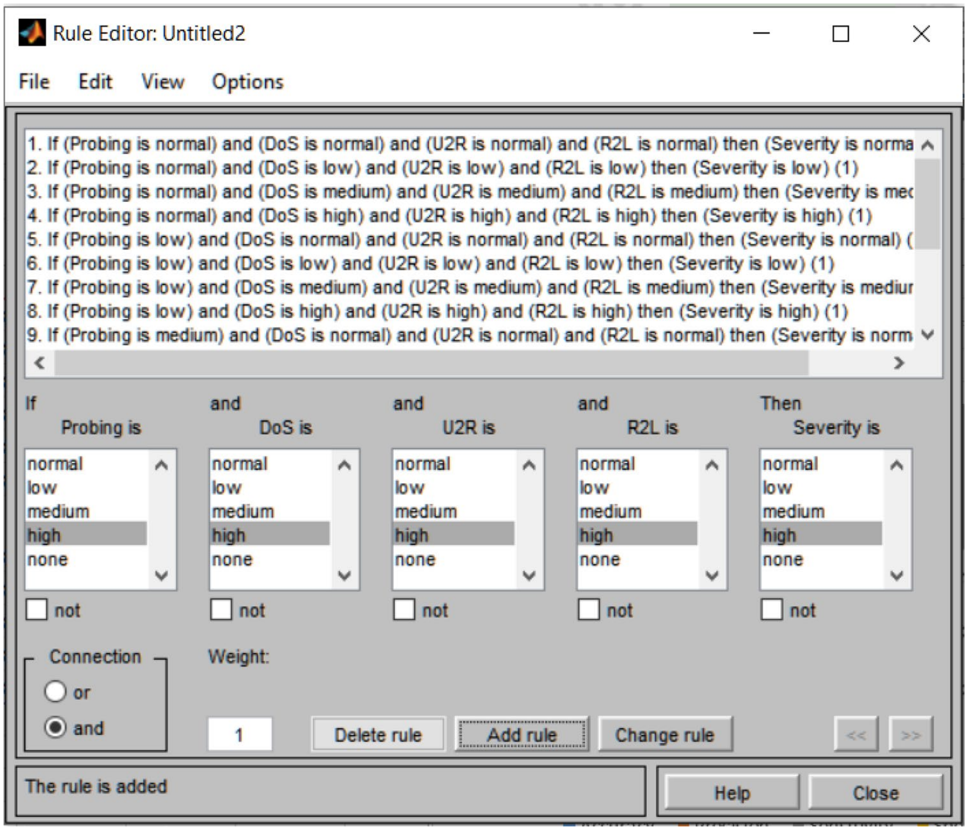
FIS Name: Untitled2
FIS Type: mamdani
And method: min
Or method: max
Implication: min
Aggregation: max
Defuzzification: centroid
Current Variable
Name: Severity
Type: output
Range: [0 1]
Renaming output variable 1 to "Severity"

Fig. 6 Fuzzy inference system editor for intrusion classification



plot points: 181
Current Variable
Name: Probing
Type: input
Range: [0 1]
Display Range: [0 1]
Current Membership Function (click on MF to select)
Name: normal
Type: trimf
Params: [-0.3333 0 0.3333]
Selected variable "Probing"

Fig. 7 Membership function editor for each of the fuzzy variables


Rule Editor: Untitled2

1. If (Probing is normal) and (DoS is normal) and (U2R is normal) and (R2L is normal) then (Severity is norma
2. If (Probing is normal) and (DoS is low) and (U2R is low) and (R2L is low) then (Severity is low) (1)
3. If (Probing is normal) and (DoS is medium) and (U2R is medium) and (R2L is medium) then (Severity is mec
4. If (Probing is normal) and (DoS is high) and (U2R is high) and (R2L is high) then (Severity is high) (1)
5. If (Probing is low) and (DoS is normal) and (U2R is normal) and (R2L is normal) then (Severity is normal) (
6. If (Probing is low) and (DoS is low) and (U2R is low) and (R2L is low) then (Severity is low) (1)
7. If (Probing is low) and (DoS is medium) and (U2R is medium) and (R2L is medium) then (Severity is mediur
8. If (Probing is low) and (DoS is high) and (U2R is high) and (R2L is high) then (Severity is high) (1)
9. If (Probing is medium) and (DoS is normal) and (U2R is normal) and (R2L is normal) then (Severity is norm

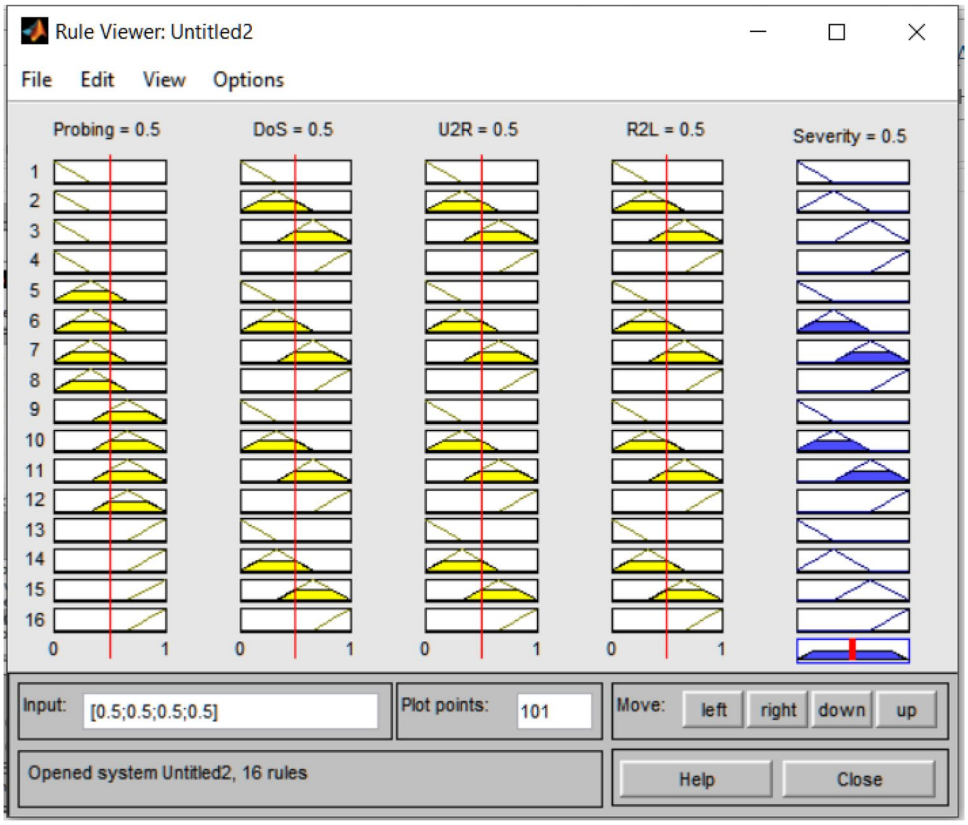
If Probing is	and DoS is	and U2R is	and R2L is	Then Severity is
normal	normal	normal	normal	normal
low	low	low	low	low
medium	medium	medium	medium	medium
high	high	high	high	high
none	none	none	none	none
not	not	not	not	not

Connection: or [✓ and]

Weight: 1

The rule is added

Fig. 8 The rule editor


Rule Viewer: Untitled2

Probing = 0.5 DoS = 0.5 U2R = 0.5 R2L = 0.5 Severity = 0.5

Input: [0.5;0.5;0.5;0.5] Plot points: 101

Opened system Untitled2, 16 rules

Fig. 9 The rule viewer

Table 16 Rule viewer adjustment for intrusion classification

No	Probing	DoS	U2R	R2L	Intrusion severity level
1	0.5	0.657	0.882	0.827	Medium
2	0.918	0.898	0.973	0.827	High
3	0.5	0.5	0.5	0.5	Low
4	0.882	0.935	0.882	0.936	High
5	0.336	0.343	0.3	0.864	Low
6	0.591	0.0833	0.627	0.882	Low
7	0.136	0.0833	0.209	0.118	Normal
8	0.3	0.806	0.645	0.445	Low
9	0.9	0.9	0.9	0.9	High
10	0.75	0.9	0.9	0.9	High

as DoS. Only 44 of the total samples from the U2R attack have the wrong classification, and only 745 of the total samples from the U2R attack have been incorrectly identified as U2R. Only 15 out of the total samples from the R2L attack have the wrong classification, only 4 out of the total samples from the R2L attack have been incorrectly labeled as U2R, and just 5 out of the total samples that are part of the R2L attack are incorrectly labeled as R2L. It can be said that most mistakes result from mistaking a U2R attack for a distinctive and normal attack.

Table 13 shows the rates of attacks after feature selection. It can be observed that the normal class has the highest number of correct classifications at 15,389 compared to the other attack types while R2L has the lowest false alarm rate compared to the other attack types. Table 14 shows the evaluation metrics after the feature selection of the developed ML-RFID-FIS model. The ML-RFID-FIS model has an overall accuracy, precision, sensitivity, specificity, and F1-score of 99.46%, 99.46%, 99.46%, 93.86%, and 99.46%, respectively. The results in Table 14 indicate that a random forest with the developed multi-level feature selection can correctly differentiate the different attack types in the dataset. The developed method could differentiate Normal, DoS, Probe, U2R, and R2L attacks with 99.51%, 99.88%, 99.18%, 95.09%, and 31.25%, respectively. The results also suggest poor classification results for R2L attacks with an F1-score of 31.25%. Therefore, the overall results in Table 14 are a pointer to the need for the developed multi-level feature selection method to enhance the random forest algorithm. These results showed that the accuracy value for intrusion detection is high for the developed ML-RFID-FIS model after the application of the multi-level feature selection method. The justification for the high accuracy can be attributed to the efficacy of the multi-level feature selection method and the ability of the random forest to deal with overfitting without affecting the overall classification accuracy.

Table 15 shows the comparison of the developed model with other existing ML algorithms using the same dataset based on reduced features for intrusion detection. Most of the ML algorithms on intrusion detection obtained an F1-score of at least 72% classification rates. The F1-score of ML-RFID-FIS is better at 99.46% compared to bagging with the closest score of 92.25%. The results suggest random forest as the least ML algorithm for the classification of intrusions with an F1-score of 72.01%. The outcomes demonstrated that the optimum IDS method is ML-RFID-FIS while the worst ID method is random forest across the evaluation metrics. The results showed that the developed ML-RFID-FIS is an improvement over the standalone random forest with accuracy, precision, sensitivity, specificity, and F1-score of 99.46%, 99.46%, 99.46%, 93.86%, and 99.46%, respectively, of ML-RFID-FIS compared to 72.00%, 72.01%, 72.00%, 93.01%, and 72.01% of random forest. These results are a justification that the developed method can provide better results than the traditional ML algorithms. Overall, the results demonstrated that the various ML methods on the same dataset performed in a well-balanced manner.

Table 17 The comparison of the proposed model with other existing models using the same dataset

Authors	Model	Accuracy	Precision	Sensitivity	Specificity	F1-score
Su et al. (2020) [52]	BAT-MC	84.3	84.7	85	85.1	84.9
Raghuvanshi et al. (2022) [53]	SVM	98.0	97.4	96.7	–	96.9
Rawat et al. (2022) [54]	PCA + DNN	76.0	76.4	75.7	75.9	76.0
Rastogi et al. (2022) [55]	RF	98.7	98.9	99.0	97.5	98.5
	K-NN	98.3	98.9	98.0	98.7	98.3
Sherin et al. (2022) [56]	Stacked Model	90.0	90	89.7	–	–
Esmaeili et al. (2022) [57]	BiLSTM	82.3	83.0	82.9	83.0	82.4
Ahanger et al. (2021) [58]	SVM	99.1	98.7	96.0	99.5	99.0
Ahmadi et al. [59]	CART + Chi-square	80.6	96.0	69.0	–	80.0
Proposed Model	ML-RFID-FIS	99.46%	99.46%	99.46%	93.86%	99.46%

The graphical representation of the performance rate of each class label of the developed ML-RFID-FIS model before the feature selection is shown in Fig. 3. Similarly, Fig. 4 shows the graphical representation of the performance rate of each class label of the developed ML-RFID-FIS model after the selection of features. Figure 5 shows the overall performance comparison for the developed ML-RFID-FIS. The graph showed that the developed model exhibited the highest value of performance compared to the other models.

Figure 6 shows the fuzzy inference system editor where the fuzzy input variables for intrusion classification are added between the 0 and 1 intervals. The fuzzy input variables include Probing, DoS, U2R, and R2L. Figure 7 shows the membership function editor for each of the fuzzy variables. This is the editor that enables the definition of linguistic variables for the various fuzzy variables within a specified fuzzy range of values indicated by normal, low, medium, and high. These linguistic variables allow fuzzification of the fuzzy variables within the specified range of values. Figure 8 shows the rule editor for the defined fuzzy variables and linguistic variables. The rule editor is where rules are defined and added based on expert knowledge. The rule of thumb defined a total of 16 rules. Figure 9 shows the rule viewer for variables combination and adjustments. The various variables are combined through the adjustment of their fuzzy values to produce a unique decision output for intrusion classification.

Table 16 shows the rule viewer adjustments for intrusion classification. The result of the rule adjustment showed the severity level of an intrusion. The result showed that an intrusion is classified as a medium when the four attack variables in order are low, medium, high, and high, respectively (rule #1). As another result, an intrusion is classified as high when all four attack variables are high (rule #2). Similarly, an intrusion is classified as low when all the input variables are medium (rule #3). An intrusion is classified as high if all four attack variables are high (rule #4). The other results are interpreted similarly. These results showed that on average, an intrusion will produce a low severity level. On some other occasions, an intrusion will produce a high severity level.

4.5 Comparative Analysis with Existing Models

To unbiasedly evaluate the reliability and distinction of the suggested model network, a comparison of the proposed model with a few current cutting-edge models was carried out. Table 17 compares the ML-RFID-FIS performances with some of the aforementioned previous state-of-the-art techniques while utilizing the same NSL-KDD dataset. According to the table, the ML-RFID-FIS model outperforms all other models in terms of performance measures. With its ML-based design, the ML-RFID-FIS model is capable of extracting and selecting its features. The developed ML-RFID-FIS model performed better with accuracy, precision, sensitivity, specificity, and F1-score of 99.46%, 99.46%, 99.46%, 93.86%, and 99.46%, respectively, when compared to the other models. The models under comparison are recent techniques developed to classify network traffic using the same NSL-KDD dataset but the developed model outperforms the recent techniques. The methods under comparison are not only recent efforts on intrusion detection but are very relevant with remarkable accuracies. Table 17 displays the comparative findings between these studies and the developed method using the same NSL-KDD dataset.

In terms of accuracy, the developed feature selection method performs better than the existing models. The developed feature selection method performs better in terms of accuracy than the alternatives. Our developed feature selection strategy produced a feature set that provides excellent classification accuracy, precision, and recall with minimal computing complexity. Using 10 of the 41 features, the model had an accuracy of 99.46%, which was its highest. The importance of the developed model lies in reducing the overfitting effect by removing unnecessary features based on the developed multi-level feature selection technique.

5 Conclusion and Future Work

This research developed a multi-level random forest algorithm for intrusion detection using a fuzzy inference system. The multi-level feature selection method combines the advantages of the filter and wrapper methods. The first stage of the multi-level feature selection is the filter method using a correlation-based feature selection to select essential features based on the multi-collinearity in the data. Next, the top features from the feature set were chosen using a genetic search strategy in correlation-based feature selection. The GSA assesses each attribute's merits, which then delivers the characteristics with the highest fitness values for selection. Additionally, it employed a rule evaluation to determine whether two feature subsets had the same fitness value, yielding the feature subset that contains the fewest features. The second stage is a wrapper method based on the sequential forward selection method to further select top features based on the accuracy of the baseline classifier. The selected top features serve as input into the random forest algorithm for detecting intrusions. Fuzzy logic was used to classify intrusions as either normal, low, medium, or high to reduce misclassification. When the developed intrusion method was compared to other existing models using the same dataset, the results revealed a higher accuracy, precision, sensitivity, specificity, and F1-score of 99.46%, 99.46%, 99.46%, 93.86%, and 99.46%, respectively. The classification of attacks using the fuzzy inference system also indicates that the developed method can correctly classify attacks with reduced misclassification. Future research could focus on developing deep learning architectures that use cutting-edge optimization techniques. The fuzzy logic linguistic variables might be expanded to include more severity classes for improved intrusion classification.

Author Contributions All the authors have designed the study, developed the methodology, performed the analysis, and written the manuscript. All authors have contributed equally.

Funding The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Data Availability Authors utilized publicly available datasets, i.e., the NSL-KDD dataset (Tavallaee et al. 2009) [51].

Declarations

Conflict of Interest The authors have no relevant financial or non-financial interests to disclose. On behalf of all authors, the corresponding author states that there is no conflict of interest.

Ethical Approval Not applicable as the study did not require ethical approval.

References

1. Waskle, S., Parashar, L., Singh, U.: Intrusion detection system using PCA with random forest approach. In 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC) (pp. 803–808). IEEE (2020)
2. Ganapathy, S., Kulothungan, K., Muthurajkumar, S., Vijayalakshmi, M., Yogesh, P., Kannan, A.: Intelligent feature selection and classification techniques for intrusion detection in networks: a survey. EURASIP J. Wirel. Commun. Netw. **2013**(1), 1–16 (2013)
3. Patel, R., Thakkar, A., Ganatra, A.: A survey and comparative analysis of data mining techniques for network intrusion detection systems. Int. J. Soft Comput. Eng. (IJSCE) **2**(1), 265–260 (2012)
4. Kajal, A., Nandal, S.K.: A hybrid approach for cyber security: improved intrusion detection system using Ann-Svm. Indian J. Comput. Sci Eng **11**(4), 412–425 (2020)
5. Amiri, F., Yousefi, M.R., Lucas, C., Shakery, A., Yazdani, N.: Mutual information-based feature selection for intrusion detection systems. J. Netw. Comput. Appl. **34**(4), 1184–1199 (2011)
6. Bour, H., Abolhasan, M., Jafarizadeh, S., Lipman, J., Makhdoom, I.: A multi-layered intrusion detection system for software-defined networking. Comput. Electr. Eng. **101**, 108042 (2022)
7. Roy, S., Li, J., Choi, B.J., Bai, Y.: A lightweight supervised intrusion detection mechanism for IoT networks. Futur. Gener. Comput. Syst. **127**, 276–285 (2022)
8. Panigrahi, R., Borah, S., Bhoi, A.K., Mallick, P.K.: Intrusion detection systems (IDS)—an overview with a generalized framework. In: Cognitive informatics and soft computing, pp. 107–117. Springer, Singapore (2020)
9. Zhou, Y., Cheng, G., Jiang, S., Dai, M.: Building an efficient intrusion detection system based on feature selection and ensemble classifier. Comput. Netw. **174**, 107247 (2020)
10. Ayo, F.E., Folorunso, S.O., Abayomi-Alli, A.A., Adekunle, A.O., Awotunde, J.B.: Network intrusion detection based on deep learning model optimized with rule-based hybrid feature selection. Infor. Secur. J. **29**(6), 267–283 (2020)
11. Saba, T., Rehman, A., Sadad, T., Kolivand, H., Bahaj, S.A.: Anomaly-based intrusion detection system for IoT networks through deep learning model. Comput. Electr. Eng. **99**, 107810 (2022)
12. Saeed, M.M.: A real-time adaptive network intrusion detection for streaming data: a hybrid approach. Neural Comput. Appl. **34**(8), 6227–6240 (2022)
13. Alhenawi, E.A., Al-Sayyed, R., Hudaib, A., Mirjalili, S.: Feature selection methods on gene expression microarray data for cancer classification: a systematic review. Comput. Biol. Med. **140**, 105051 (2022)
14. Awotunde, J. B., Abiodun, K. M., Adeniyi, E. A., Folorunso, S. O., Jimoh, R. G.: A deep learning-based intrusion detection technique for a secured IoMT system. In International Conference on Informatics and Intelligent Applications. Springer, Cham. (pp. 50–62) (2021)
15. Alzahrani, M.Y., Bamhdi, A.M.: Hybrid deep-learning model to detect botnet attacks over internet of things environments. Soft Comput. **26**, 1–15 (2022)
16. Kalinin, M.O., Krundyshev, V.M., Sinyapkin, B.G.: Development of the intrusion detection system for the internet of things based on a sequence alignment algorithm. Autom. Control. Comput. Sci. **54**(8), 993–1000 (2020)
17. Panigrahi, R., Borah, S., Pramanik, M., Bhoi, A.K., Barsocchi, P., Nayak, S.R., Alnumay, W.: Intrusion detection in cyber–physical environment using hybrid naïve bayes—decision table and multi-objective evolutionary feature selection. Comput. Commun. **188**, 133–144 (2022)
18. Sharma, S., Verma, V.K.: AIEMLA: artificial intelligence enabled machine learning approach for routing attacks on internet of things. J. Supercomput. **77**(12), 13757–13787 (2021)
19. Verma, A., Ranga, V.: Evaluation of network intrusion detection systems for RPL based 6LoWPAN networks in IoT. Wirel. Pers. Commun. **108**(3), 1571–1594 (2019)
20. Li, M., Sun, Y., Lu, H., Maharjan, S., Tian, Z.: Deep reinforcement learning for partially observable data poisoning attack in crowdsensing systems. IEEE Internet Things J. **7**(7), 6266–6278 (2019)
21. Soe, Y. N., Feng, Y., Santosa, P. I., Hartanto, R., Sakurai, K.: Implementing lightweight iot-ids on raspberry pi using

correlation-based feature selection and its performance evaluation. In International Conference on Advanced Information Networking and Applications. Springer, Cham. (pp. 458–469) (2019)

22. Shafiq, M., Tian, Z., Bashir, A.K., Du, X., Guizani, M.: IoT malicious traffic identification using wrapper-based feature selection mechanisms. Comput. Secur. **94**, 101863 (2020)
23. Anderson, J. P.: Computer security threat monitoring and surveillance. Technical Report, James P. Anderson Company,1, (1980)
24. Denning, D.E.: An intrusion-detection model. IEEE Trans. Software Eng. **2**, 222–232 (1987)
25. Awotunde, J.B., Chakraborty, C., Adeniyi, A.E.: Intrusion detection in industrial internet of things network-based on deep learning model with rule-based feature selection. Wirel. Commun. Mobile Comput. **2021**, 1 (2021)
26. Alharbi S, Rodriguez P, Maharaja R, Iyer P, Bose N, Ye Z.: FOCUS: A fog computing-based security system for the Internet of Things. In2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC) Jan 12. IEEE (pp. 1–5). (2018)
27. Awotunde, J.B., Jimoh, R.G., Folorunso, S.O., Adeniyi, E.A., Abiodun, K.M., Banjo, O.O.: Privacy and security concerns in IoT-based healthcare systems. In: The Fusion of internet of things artificial intelligence and cloud computing in health care, pp. 105–134. Springer, Cham (2021)
28. Xiao, L., Li, Y., Huang, X., Du, X.: Cloud-based malware detection game for mobile devices with offloading. IEEE Trans. Mob. Comput. **16**(10), 2742–2750 (2017)
29. Awotunde, J.B., Misra, S.: Feature extraction and artificial intelligence-based intrusion detection model for a secure internet of things networks. In: Illumination of artificial intelligence in cybersecurity and forensics, pp. 21–44. Springer, Cham (2022)
30. Vinayakumar, R., Alazab, M., Jolfaei, A., Soman, K. P., Poornachandran, P.: Ransomware triage using deep learning: twitter as a case study. In 2019 Cybersecurity and Cyberforensics Conference (CCC). IEEE. (pp. 67–73) (2019)
31. Egea, S., Mañez, A.R., Carro, B., Sánchez-Esguevillas, A., Lloret, J.: Intelligent IoT traffic classification using novel search strategy for fast-based-correlation feature selection in industrial environments. IEEE Internet Things J. **5**(3), 1616–1624 (2017)
32. Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Breitenbacher, D., Shabtai, A., Elovici'N-BaIoT, Y.: Networkbased detection of IoT botnet attacks using deep autoencoders'. IEEE Pervasive Comput. Special Issue Securing IoT **17**(3), 12–22 (2018)
33. Su, S., Sun, Y., Gao, X., Qiu, J., Tian, Z.: A correlation-change based feature selection method for IoT equipment anomaly detection. Appl. Sci. **9**(3), 437 (2019)
34. Tan, Q., Gao, Y., Shi, J., Wang, X., Fang, B., Tian, Z.: Toward a comprehensive insight into the eclipse attacks of tor hidden services. IEEE Internet Things J. **6**(2), 1584–1593 (2018)
35. Tian, Z., Shi, W., Wang, Y., Zhu, C., Du, X., Su, S., Guizani, N.: Real-time lateral movement detection based on evidence reasoning network for edge computing environment. IEEE Trans. Ind. Inf. **15**(7), 4285–4294 (2019)
36. Resul, D.A.S., Gündüz, M.Z.: Analysis of cyber-attacks in IoT-based critical infrastructures. Int. J. Inf Secur Sci **8**(4), 122–133 (2020)
37. Adewole, K.S., Salau-Ibrahim, T.T., Imoize, A.L., Oladipo, I.D., AbdulRaheem, M., Awotunde, J.B., Aro, T.O.: Empirical analysis of data streaming and batch learning models for network intrusion detection. Electronics **11**(19), 3109 (2022)
38. Liu, X., Tang, J.: Mass classification in mammograms using selected geometry and texture features, and a new SVM-based feature selection method. IEEE Syst. J. **8**(3), 910–920 (2013)
39. Ogundokun, R.O., Awotunde, J.B., Sadiku, P., Adeniyi, E.A., Abiodun, M., Dauda, O.I.: An enhanced intrusion detection system using particle swarm optimization feature extraction technique. Procedia Comput. Sci. **193**, 504–512 (2021)
40. Xue, Y., Tang, Y., Xu, X., Liang, J., Neri, F.: Multi-objective feature selection with missing data in classification. IEEE Trans. Emerg. Topics Comput. Intell. **6**(2), 355–364 (2021)
41. Xue, Y., Xue, B., Zhang, M.: Self-adaptive particle swarm optimization for large-scale feature selection in classification. ACM Trans. Knowl. Discov. Data (TKDD) **13**(5), 1–27 (2019)
42. Ma, J., Gao, X.: Designing genetic programming classifiers with feature selection and feature construction. Appl. Soft Comput. **97**, 106826 (2020)
43. Rostami, M., Berahmand, K., Forouzandeh, S.: A novel community detection based genetic algorithm for feature selection. J. Big Data **8**(1), 1–27 (2021)
44. Lualdi, M., Fasano, M.: Statistical analysis of proteomics data: a review on feature selection. J. Proteom. **198**, 18–26 (2019)
45. Alcalá, R., Gacto, M.J., Herrera, F., Alcalá-Fdez, J.: A multi-objective genetic algorithm for tuning and rule selection to obtain accurate and compact linguistic fuzzy rule-based systems. Internat. J. Uncertain. Fuzziness Knowl. Based Syst. **5**(05), 539–557 (2007)
46. Zadeh, L.A., Klir, G.J., Yuan, B.: Fuzzy sets, fuzzy logic, and fuzzy systems: selected papers, vol. 6. World Scientific, Singapore (1996)
47. Abu Arqub, O.: Adaptation of reproducing kernel algorithm for solving fuzzy Fredholm-Volterra integrodifferential equations. Neural Comput. Appl. **28**(7), 1591–1610 (2017)
48. Alshammari, M., Al-Smadi, M., Arqub, O.A., Hashim, I., Alias, M.A.: Residual series representation algorithm for solving fuzzy duffing oscillator equations. Symmetry **12**(4), 572 (2020)
49. Abu Arqub, O., Singh, J., Maayah, B., Alhodaly, M.: Reproducing kernel approach for numerical solutions of fuzzy fractional initial value problems under the Mittag-Leffler kernel differential operator. Math. Methods Appl. Sci. (2021). https://doi.org/10.1002/mma.7305
50. Abu Arqub, O., Singh, J., Alhodaly, M.: Adaptation of kernel functions-based approach with Atangana–Baleanu–Caputo distributed order derivative for solutions of fuzzy fractional Volterra and Fredholm integrodifferential equations. Math. Methods Appl. Sci. (2021). https://doi.org/10.1002/mma.7228
51. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A. A.: A detailed analysis of the KDD CUP 99 data set. In 2009 IEEE symposium on computational intelligence for security and defense applications. IEEE. (pp. 1–6) (2009)
52. Su, T., Sun, H., Zhu, J., Wang, S., Li, Y.: BAT: Deep learning methods on network intrusion detection using NSL-KDD dataset. IEEE Access **8**, 29575–29585 (2020)
53. Raghuvanshi, A., Singh, U.K., Sajja, G.S., Pallathadka, H., Asenso, E., Kamal, M., Phasinam, K.: Intrusion detection using machine learning for risk mitigation in IoT-enabled smart irrigation in smart farming. J. Food Qual. **2022**, 1 (2022)
54. Rawat, S., Srinivasan, A., Ravi, V., Ghosh, U.: Intrusion detection systems using classical machine learning techniques vs integrated unsupervised feature learning and deep neural network. Internet Technol. Lett. **5**(1), e232 (2022)
55. Rastogi, S., Shrotriya, A., Singh, M.K., Potukuchi, R.V.: An analysis of intrusion detection classification using supervised machine learning algorithms on NSL-KDD dataset. J. Comput. Res. Innov. (JCRINN) **7**(1), 124–137 (2022)
56. Sherin, V. I. J., Radhika, N.: Stacked ensemble-IDS using NSL-KDD dataset. J. Pharm. Negative Results. **13**, 351–356 (2022)

57. Esmaeili, M., Goki, S.H., Masjidi, B.H.K., Sameh, M., Gharagozlou, H., Mohammed, A.S.: ML-DDoSnet: IoT intrusion detection based on denial-of-service attacks using machine learning methods and NSL-KDD. Wirel. Commun. Mob. Comput. **2022**, 1 (2022)
58. Ahanger, A. S., Khan, S. M., Masoodi, F.: An effective intrusion detection system using supervised machine learning techniques. In 2021 5th International Conference on Computing Methodologies and Communication (ICCMC). IEEE. (pp. 1639–1644) (2021)
59. Ahmadi, S. S., Rashad, S., Elgazzar, H.: Efficient feature selection for intrusion detection systems. In 2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON). IEEE. (pp. 1029–1034) (2019)