Computer Standards & Interfaces 87 (2024) 103779

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

FDF-HybridFS: Towards design of a failure detection framework using hybrid feature selection method for IP core networks that connect 5G core in NFV-based test environment

Anjali Rajak *, Rakesh Tripathi

Department of Information Technology, National Institute of Technology Raipur, 492010, India

ARTICLE INFO

Keywords:
Failure detection
Core network
Route information
Features ranking
Feature selection

ABSTRACT

With the advancement of recent technologies, high-speed and secure internet connectivity is required for 5th-generation mobile networks. A massive number of smart devices connected to the networks generate a large amount of data. Consequently, increasing traffic has generated more strain on networks and devices. Therefore, to handle the huge amount of data generated by the various devices and to detect network and device failures, automation is expected to play an important role. Hence, these requirements have accelerated the necessity of automation. This study presents a novel hybrid feature-selection technique for detecting network and device failure in IP core networks that connect 5 G core networks for internet connectivity. This approach first performs feature ranking using the MI (Mutual information), CC (Correlation coefficient), and GI (Gini index) to get the three different feature sets. Subsequently, in order to acquire a single optimized feature set, features are combined using a union operation. This optimized feature set is fed to the wrapper-based Boruta feature selection algorithm. In addition, to handle the data imbalance problem, we have applied the SMOTETomek method. Finally, obtained optimized feature set is fed to the three most popular ensemble approaches, Random forest, LGBM (Light gradient boosting machine), and xGBoost for the detection of failure. The proposed detection system's performance is evaluated using the KDDI dataset provided by KDDI Corporation. The effectiveness of the proposed FDF-HybridFS is tested and compared with several recent state-of-the-art methods cited in the related work in terms of detection rate, precision, accuracy, and F1 score. An analysis of the proposed failure detection framework's performance on the KDDI dataset shows that it outperforms state-of-the-art methods in terms of both detection rate and detection accuracy.

1. Introduction

The recent advancement of network technologies such as cloud-native functions, 5G, and network function virtualization has dramatically transformed the telecommunications industry and brought faster speeds experience to the end users. According to [1] the market for 5G technologies is expected to grow from $30.11 billion in 2022 to $99.76 billion by 2026. The growth of 5G technologies is expected to be driven by the increase in government initiatives to promote and roll out emerging 5G technology. The rapid development of 5G services such as gaming AR/VR, smart transportation, and a large number of connected IoT devices. These huge numbers of network-connected devices not only increase the network infrastructure but also cause critical network issues. According to [2], downtime on cloud services has negative economic consequences, such as a loss of more than $30,000 to $6700, 00 for a service provider such as YouTube. As a result, network management and failure detection play a significant role.

The internet connection required for 5G networks must be stable, high-speed, and secure, but when it breaks down, the consequences can be disastrous. Additionally, since the internet is operated mutually amongst internet service providers (ISP), even if a breakdown happens in a certain internet service provider's domain, the failure quickly spreads across the entire network. Therefore, only experienced ISPs can handle a network failure that affects the entire network. The ability to quickly and automatically detect anomalies is important. Border gateway protocol is used to connect the IP backbone network of different ISPs. When receiving internal or external route information, a border gateway protocol router must continuously update the route

* Corresponding author.
E-mail address: arajak.phd2021.it@nitrr.ac.in (A. Rajak).

https://doi.org/10.1016/j.csi.2023.103779
Received 11 January 2023; Received in revised form 12 May 2023; Accepted 1 August 2023
Available online 2 August 2023

information and respond appropriately [3]. Thus, these routers are essential to 5G services, and it is required to immediately identify software & hardware flaws and malfunctions in order to sustain a specific level of service. Moreover, data-based optimization faces difficulties due to increasing network traffic. Recent cutting-edge AI technologies provide novel approaches that can change our work focus from fault handling to fault prediction, which allows operators to take preventative measures in advance. On the other hand, due to the COVID-19 pandemic, implementing social restrictions increased internet traffic, particularly that remote working, online education, and meetings. For example, Netflix has faced an increase in subscribers, with approximately 16 million individuals signing in accounts in the first quarter of 2020 [4–6]. Such increasing traffic has caused more device and network failures. Hence, automatically detecting failure is an important problem.

FS (Feature selection) is the technique of recognizing the input variable, feature, or attributes that are most significant for the failure detection systems [7,8]. The features that do not contribute to the process must be eliminated, as their inclusion increases the misclassification rate and decreases accuracy. In general, computational time gradually increases due to the high dimensionality of internet traffic. Moreover, if an irrelevant feature is identified in advance, then developing an efficient detection or classification system for a 5G network would become a trivial task. Hence, the overall performance in terms of classification accuracy and failure detection rate would increase, and the training and testing time of the detection system could be reduced [9]. FS processes are broadly divided into two techniques: the filter method and the wrapper method. Filter techniques are not dependant on any learning methodology as they focus on the fundamental characteristics of data. Furthermore, a filter-based method is not computationally expensive and has a high potential for generalization due to its self-determination from the induction algorithm. Though, if the selection criteria are different from the learning criteria used in a training phase, the right feature subset may not be chosen. Because most filters only evaluate one feature at a time, they often find a difficulty to identify a function that will improve the criteria as a whole. This is another downside of the filter method. On the other hand, the wrapper method relies on a greedy search algorithm, as it tests all possible feature combinations and chooses the combination that provides the best result for a particular ML (Machine learning) task. The wrapper method's downside is that it turns computationally expensive if the feature set is particularly large. Hence, we have proposed the HybridFS method (Hybrid filter-wrapper feature selection) that exploits the advantage of both methods.

In general, ML algorithms aim to identify patterns in the training dataset before developing a model for classifying incoming input based on the discovered patterns. In the context of 5G network security applying ML algorithms, it is crucial that the training dataset includes all essential data that illustrate real-world failures as these datasets are the foundation for acquiring system learning. Thus, it is crucial to choose a significant dataset that includes failure scenarios that can be used to evaluate how efficiently the detection system is performing.

The major challenges are listed below:

1 Selecting an appropriate FS method that can efficiently evaluate the importance of features is a challenging task.
2 Automatically managing huge amounts of data is a challenging task, when a huge amount of data has been generated by a huge number of smart devices, which has induced more strain on networks, cloud data centers, and devices.
3 How to rapidly and automatically detect device & network failures is a challenging problem for the network operator?

To address the above challenges, we have presented a new hybrid-FS technique that can efficiently detect abnormal instances for the devices & network in large-scale NFV-based test environments. Feature preprocessing steps are used in the initial phase, which includes three important steps that is feature mapping, imputing, and normalizing the features. The second phase includes a new hybrid FS method that is based on the filter and wrapper FS methods. We have divided the second phase into the following two steps: first, we perform the filter-based FS techniques using the Gini index, followed by mutual information and the correlation coefficient. Through ranking features in accordance with the ranking function of the relevant feature selection technique, three feature sets are generated as a consequence. By performing union set operation on the ranked feature set, we have kept those features that were selected by the filter techniques. In the second phase, we used the wrapper-based Boruta feature selection techniques to further improve these feature sets and generate a single optimized feature set. In the third phase, we applied the iForest method to eliminate the outliers. Subsequently, the SMOTETomek method is then applied to handle the data imbalance problem. The selected feature set is then provided to three conventional ensemble learning approaches -LGBM, xGBoost, and RF, in the final phase. The proposed system is evaluated using a dataset provided by KDDI Corporation that includes device and network failure types.

The proposed FDF- HybridFS main contributions include the following:

(1) With the help of the correlation coefficient, mutual information, Gini index, and Boruta feature selection techniques, this study proposed a novel hybrid filter-wrapper-based FS method, to obtain a significant feature set for developing an efficient network and device failure detection system for the IP networks that connect 5G core networks.
(2) The proposed model eliminates outliers by using iForest (isolation forest) outlier detection methods. In addition, in order to handle the data imbalance issue and improve the accuracy of the KDDI dataset, we apply the SMOTETomek (SMOTE with Tomek link) data balancing technique to three ML algorithms, such as xGBoost, LGBM, and RF, using analytical tools.
(3) To illustrate the performance of our proposed FDF-HybridFS in terms of accuracy, f1 score, precision, and the detection rate on the KDDI dataset that contains the failure types. In addition, our proposed FDF-HybridFS is compared to some recent state-of-the-art methods, and the classifier is validated using a 10-fold CV (cross-validation) method.
(4) The experimental results demonstrate that our proposed approach significantly outperforms existing approaches for different types of failure in terms of accuracy and failure detection rate.

This work is structured as follows: A brief discussion of the relevant study on network failure analysis is presented in Section 2. The architecture of the proposed novel detection system is presented in Section 3. In Section 4, the experimental results and effectiveness of the proposed failure detection system are evaluated using the KDDI dataset, and then experimental outcomes are compared with some relevant existing methods for the KDDI dataset. Finally, in Section 5 we conclude this work with some directions for the future.

2. Related work

Network fault analysis using artificial intelligence has been the subject of numerous studies. The majority of approaches rely on thresholds, predefined rules, and expert experiments. This section presents an overview of related work on failure detection.

In order to examine the root cause of failures in the NFV-based environment authors in [5], have presented an ML-based fault classification. This approach collects 41 attributes from the network environment and uses ML algorithms to analyse these features in order to find three kinds of root causes including CPU overload, node down, and interface down. The data set generator, machine learning-based fault

classifier, preprocessor, and evaluator are its four functional building elements. The work of ML-based fault classification can make use of failure data that is periodically generated by the data set generator. They train and test their models using three algorithms: MLP (Multilayer Perceptron), RF, and SVM. The outcome demonstrates that RF performs well even with a small data set, while SVM could perform better by expanding training data, feature reduction, or adjusting the mix of normal/abnormal samples. However, their study does not include the feature ranking and feature selection technique. These techniques are an important phase in achieving better performance of a machine learning method. Our proposed model can perform better in all cases, especially for a huge volume of network log data, by extracting relevant information from a huge amount of data. In [4], authors have presented an efficient system to predict network failure from huge volumes of unstructured log files in a real-time scenario. Their proposed approach uses three steps: feature extraction, feature refinement, and feature reduction. They tested their work using six ML algorithms: DT (Decision Tree), xGBoost, LGBM, SVM, MLP, and RF. DT has achieved an accuracy of 80.95 percent, xGBoost has achieved an accuracy of 93.69 percent, LightGBM has achieved an accuracy of 93.33 percent, SVM has achieved an accuracy of 79.05 percent, MLP has achieved an accuracy of 81.31 percent and RF has shown an accuracy of 92.74 percent. Results demonstrated that xGBoost outperforms others in detecting failure. However, their study only focuses on extraction, refinement, and reduction of the features they have not included any approach to select the optimized feature set. In [6], authors have proposed a root cause analysis method using ML and time-series network parameters to identify the root cause of problems in the software-defined network environment. There are different ML models, so they have considered the balance between the time complexity and accuracy to choose the suitable algorithm. In addition, they contribute to the troubleshooting datasets to detect three root causes: buffer overload, switch failure, and link failure. The experimental finding shows that the f1-score, precision, and recall of the proposal are approximately 94 percent in dynamic networks and 97 percent in static networks. Although they employed wrapper-based FS methods, their study does not include a details discussion about the feature selection process. In [10], authors have presented a fault detection system for local area networks. The detection system is based on a set of rules defined by network administrators' experience and the data obtained throughout the network monitoring procedure. Even though these approaches can be implemented automatically using scripts, they often have low levels of inefficiency and high costs associated with human effort. As a result, methods including probabilistic approaches and finite state machines have also been studied [11]. The authors of [12] have developed a model based on finite state machines and achieved fault detection of partially recorded data sequences. In [13], authors utilize finite state machines, a probabilistic method is used to select the synchronization of situations and design adaptive strategies in the most efficient manner. However, these conventional techniques can hardly adapt to the frequent and dynamic changes in the network topology. In contrast, the volume of data received from regulated entities is increasingly huge in the era of the 5th generation, and data-driven fault detection approaches can gain enormous benefits. With the spread of the adoption of ML technology in multiple areas, numerous new investigations have been proposed on network failure analysis using machine learning. This promising technology can also be advantageous for networking itself. In [14], authors have proposed a Bayesian approach for detecting and monitoring faults that might arise in the Internet route. In a network campus area, they extract data using edge-switching devices and then classify using the Bayesian approach. It has been discovered that the classification results are over 90 percent. In [13], to localize and recognize the most probable cause of failures impacting a given service, authors have presented a probabilistic failure localization method based on BN (Bayesian Networks). Time series monitoring data that was taken from various light paths are used by the authors. A trained Bayesian network is used by an algorithm to localize and specify the optical layer's most likely cause of the errors when a service identifies extreme errors. In [15], authors have suggested root cause localization and anomaly detection for VNF (Virtual network function) using supervised machine learning. Based on monitoring data, this method identifies SLA violations. It can recognize the root anomalous virtual network function virtual machine causing Service level agreement (SLA) violations and obtain a low false alarm rate, high recall, and high precision. Their studies in [14–16] demonstrate that the suggested methodology can provide high accuracy of fault classification. Authors do not, however, compare their approach to other training scenarios or different ML algorithms. In [17], the authors compared the link fault detection capabilities of three machine learning models, including SVM, random forest, and MLP. The measures taken from the conventional traffic flow, including E2E delay, packet loss, and aggregate flow rate, are analysed by the authors in order to build a three-stage ML-LFIL (ML for Link Fault Identification and Localization). In [17,18], compare the outcomes of various machine learning algorithms and examine how training conditions affect performance. However, their ML model's objective is to predict service metrics and fault detection, and it does not sufficiently address fault classification. The performance of faults classification using EM (Expectation maximization), Fuzzy-C-Means, and K- Means have been compared by authors in [19]. They create a prototype to illustrate how to classify the network traffic problem under various scenarios using data sets attained from a network with light and heavy traffic scenarios in the routers and servers. The results demonstrate that FCM is more accurate than K-Means and expectation maximization. However, processing data requires extra time. Only data related to the physical interface is the authors' primary concern. As a consequence, there is no sufficient study on fault classification in NFV (Network function virtualization) environment. In Ref. [20], authors have discussed the vulnerability of IoT networks to cyber-attacks and the importance of intrusion detection systems in preventing them. However, the high dimensionality of existing IDS can reduce the efficiency of IoT systems, necessitating the removal of repetitive and irrelevant features. The paper presents a hybrid feature reduction approach that uses feature ranking techniques and combines the resulting feature sets to optimize detection accuracy. The proposed system is evaluated using three datasets and achieves high detection rates and accuracy compared to other state-of-the-art techniques. They were not employing any data-balancing techniques in their study. Another study, described in [9], proposed a new detection method that limits the feature space to reduce computation time and overfitting. The authors used data preprocessing and feature selection to enhance model generalizability and classification accuracy, respectively. They applied k-NN, logistic regression, SVM, GB, and decision tree models, and found that the GB model outperformed state-of-the-art approaches with 99.97% accuracy. In [21], the authors reviewed class imbalanced learning scenarios on five clinical datasets using various ML classifiers and label balancing methods, such as SMOTE, SMOTETomek, SMOTEEN, and random oversampling. amongst all the methods, kNN with SMOTEEN had the highest recall, ACC, PR, and F1 Score. In [22], the authors implemented a lightweight IDS system for the Internet of Things environment that combined gain ratio and CC methods for feature selection. They evaluated the system's performance using RF, DT, J48, and a logistic model tree and found that RF and J48 performed best in detecting theft and DDoS attacks.

The literature mentioned above summarizes the key findings of the different studies. Despite the existence of many studies, the primary research gap may serve as the basis for the present work discussed here. None of the studies have focused on the selection of suitable features to increase the ACC (accuracy) and the DR (detection rate) of the classifier for failure or fault detection remains a challenging task. Several studies only focus on the information related to a physical interface; hence, a comparative study on failure detection or classification in an NFV-based environment is still missing in this area of research. Based on the literature gap, this work presents a framework that can detect

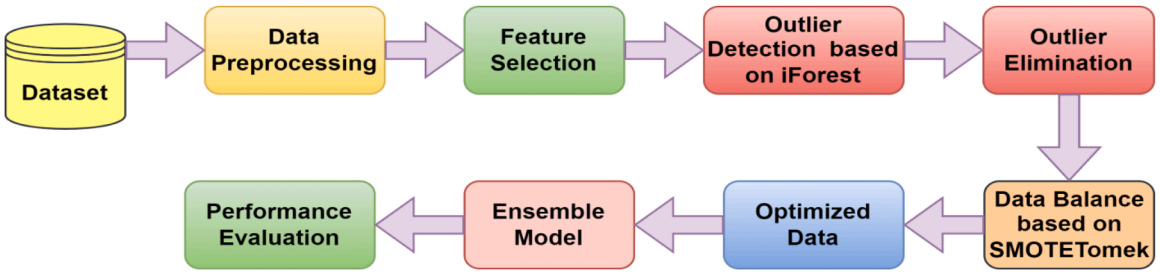

Fig. 1. The proposed flow chart for the failure detection framework.

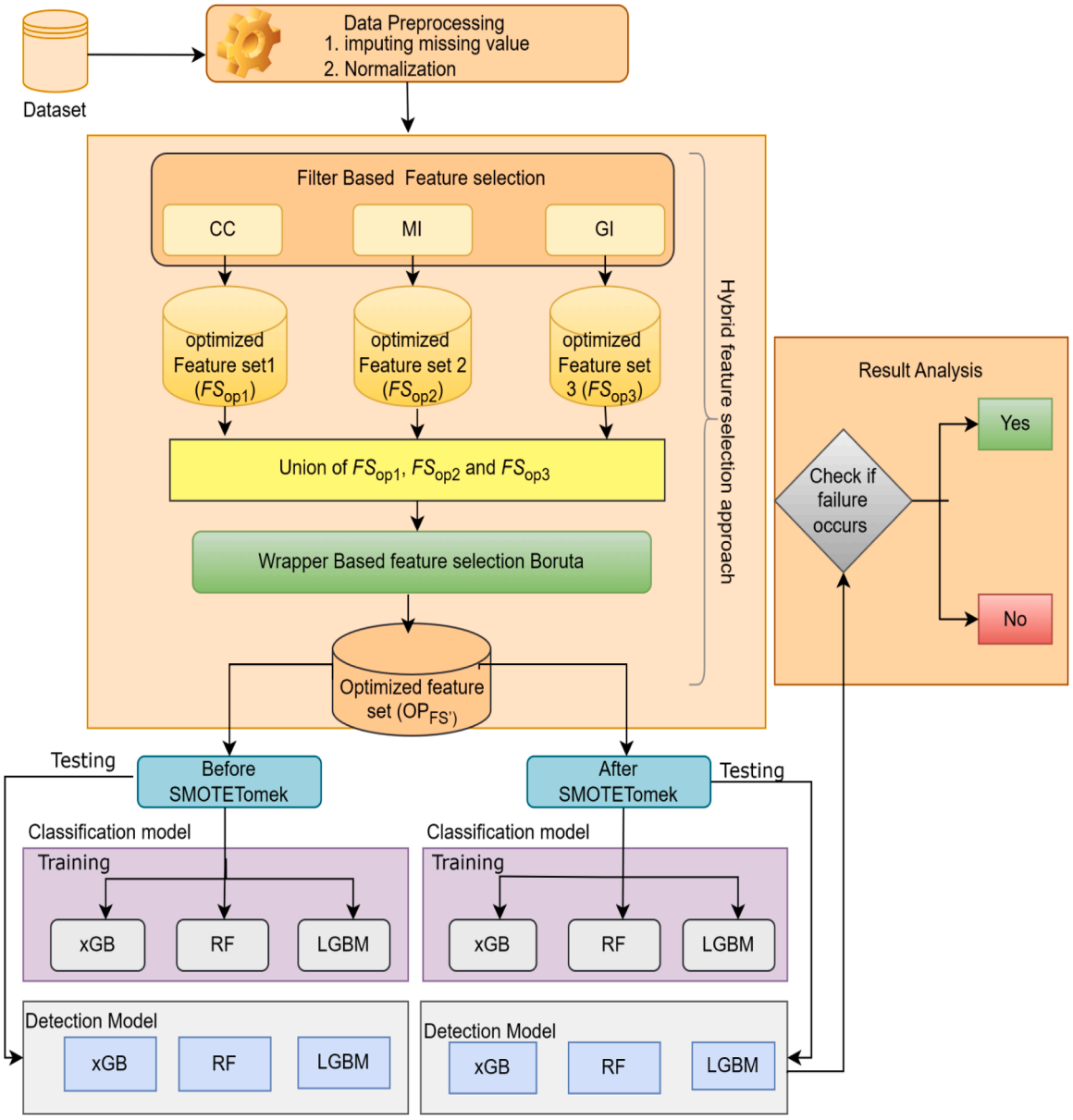

Fig. 2. The proposed framework for the failure detection framework using hybrid FS.

failure from a normal instance of network traffic by selecting the important features.

3. Proposed architecture

In this section, the proposed failure detection framework for detecting network and device failures is explained in detail. Fig. 1 shows the flow chart of the proposed failure detection system. The pre-processing of network traffic, feature ranking, and feature selection are some of the functional components that are included in each phase of the system, which utilizes a filter-wrapper-based Hybrid-FS technique. The iForest method is used to detect and eliminate outliers, and then the

Algorithm 1
A proposed Hybrid-FS technique.

Input: F ∈ {f1, f2, f3…. fn}
Output: FS_{Opf} (optimized feature set)
Examine dataset that contain F ∈ {f1, f2, f3…. fn}
CC is used to determine the degree of similarity amongst all the features.
for *i* = 1,2, 3…. N do// column index is represented by i.
Calculate CC using Eq. (2).
$FS_{Op1} \leftarrow$ F ∈ {f1, f2, f3…. fp}// where p shows most important features in set FS_{op1}
end for
To find the optimized feature subset MI is applied.
for *i* = 1,2, 3……N do
Calculate MI using Eq. (12).
$FS_{Op2} \leftarrow$ F ∈ {f1, f2, f3….fq}// where q shows important features in set FS_{op2}
end for
Gini index is used to calculate the feature importance (FI).
for *i* = 1,2, 3…. N do
Calculate GI using Eqs. (10) and (11).
$FS_{Op3} \leftarrow$ F ∈ {f1, f2, f3…. fr}// where r shows most important features in set FS_{op3}
end for
$FS_{opf} \leftarrow (FS_{Op1} \cup FS_{Op2} \cup FS_{Op3})$ // Union operation
Calculate the Z-score and then find the max *Z-score using Boruta method.*
for *i* = 1,2, 3……. N do
Calculate Z-score by applying Boruta method
$OP_{FS'} \leftarrow$ F ∈ {f1, f2, f3…. fb}// where b shows most important features in set FS_{opf}
end for
Return $OP_{FS'}$

SMOTETomek is applied to balance the data. Finally, ensemble models are applied as a classifier with optimized data. Fig. 2 shows the framework of the proposed detection system for network and device failure. This detection system employs the hybrid filter-wrapper-based FS approach. Filter-based methods CC, MI, and GI are applied to the pre-processed dataset. Then, features are merged by applying a union set operation to get the single optimized feature set for failure detection, and this optimized feature set is fed into the Boruta feature selection method. Finally, an optimized feature set is obtained. Subsequently, the SMOTETomek method is applied for data balancing, and then three well-known machine learning models, RF, LGBM, and xGBoost, are utilized as analytical tools for NFV-based test environments to detect failures in core networks. 10-fold CV (Cross-validation) techniques are simultaneously applied to train the machine learning model. As a result, each phase of the proposed models contributes significantly to improving the overall performance of the system. The discussion of all the phases is as follows:

3.1. Data pre-processing

3.1.1. Feature mapping

A categorical variable must be mapped into a numeric integer since incoming network traffic contains both numeric and categorical attributes; On incoming traffic, the LE (Label encoding) technique is applied. Each feature is converted to a numeric value using this technique. A different column is not added. Rather different categories are converted into numeric integer values. So, less computation is required.

3.1.2. Imputing missing value

Incoming traffic often contains a missing value. These values should be handled appropriately while performing the analysis. Imputation of categorical missing value is replaced with NA class and for missing integer values; it is replaced with the mean of the respective features.

3.1.3. Feature normalization

In the proposed work min-max normalization method is used. This feature scaling performs a linear transformation on the original data. The scaled data range lies between 0 and 1.

$$z^{new} = \frac{z - min(z)}{max(z) - min(z)} \tag{1}$$

Where z is an original value, z^{new} is the normalized value, min (z) is represented by the minimum value for a certain attribute in the dataset, and the maximum value is denoted by max (z).

3.2. Feature selection technique

The proposed Hybrid-FS technique for network and device failure detection or classification by an NFV-based test environment is presented in this section. A ranking method is used by filter methods to rank the variables for selection, and the data subset with the best performance is selected by using the wrapper method, which evaluates all data subsets. Although the filter approaches are identified as being scalable, classifier-independent, and, computationally simple, they fail to take into account how certain features may affect the model's performance. The wrapper methods function similarly to the filter methods, but they use a predetermined classification algorithm in place of an independent measure for subset evaluation. A subset of the features is used to calculate the model's accuracy that is subsequently evaluated. Wrapper approaches outperform filter methods in terms of results, but they are more computationally expensive when the number of available features is huge. Therefore, we propose a hybrid filter-wrapper-based FS technique that utilizes the advantages of both methods. The proposed FS method is represented in Algorithm I. Hence, the proposed detection system demonstrates a higher accuracy and improved detection rate as it utilizes the advantages of both filters- wrapper hybrid-FS techniques. Section 7.2 explain the working of the proposed Algorithm 1.

3.2.1. CC (Correlation coefficient)

In this work, CC based FS method is used. Most of the features (F) that are present in the dataset are irrelevant and contain the same value. The best candidate for removal is the independent features that have no correlation with the target variable. Additionally, rather than utilizing both features, if two independent features are highly correlated to one another, the system will become more complex, increasing the computational time, rather than improving performance [23]. Therefore, the proposed system eliminates all of these features from the dataset and presents a simpler model for efficiently detecting network and device failures. In the proposed model, PCC filter-based FS method is applied. It works on the theory that calculates the linear dependency of two attributes, whether they are independent or dependant. The PCC value of the dependant feature will be ± 1 and the 0 PCC value will be for the independent feature. Subsequently, PCC makes an attempt to calculate how much another feature will increase if one is increased, and conversely. ρ (PCC) could be calculated for two features where f ∈ {*F1, F2, F3…. FN*} and Eq. (2) shows the target variable denoted by t_g.

$$\rho(f, tg) = \frac{\mathrm{cov}(f, tg)}{\sigma_f \sigma_{tg}} \tag{2}$$

Where the standard deviation is represented by $\sigma_f \sigma_{tg}$and covariance is denoted by cov between f and tg.

σ_f and σ_{tg}Calculated using Eqs. (3) and (4)

$$\mathrm{SD}(\sigma_f)\& = \sqrt{\mathrm{var}_f} \tag{3}$$

$$\mathrm{SD}(\sigma_{tg})\& = \sqrt{\mathrm{var}_{tg}} \tag{4}$$

And with the help of Eqs. (5) and (6) variance is calculated for the features where total values present in each feature is N and the mean for f and tg is $\mu_f and \mu_{tg}$

$$\mathrm{var}\left(\sigma_f^2\right) = \frac{1}{N}\sum_{i=1}^{N}\left(f_i - \mu_f\right)^2 \tag{5}$$

$$\mathrm{var}\left(\sigma_{tg}^2\right) = \frac{1}{N}\sum_{i=1}^{N}\left(tg_i - \mu_{tg}\right)^2 \tag{6}$$

Using Eqs. (7) and (8), val represents the values for the relevant features in the considered dataset, and the mean for both features can be calculated.

$$\mu_f \& = \frac{\sum_{i=1}^{N} \mathrm{val}_i}{N} = \frac{\mathrm{val}_1 + \mathrm{val}_2 + \mathrm{val}_3 + \cdots + \mathrm{val}_N}{N} \tag{7}$$

$$\mu_{tg} \& = \frac{\sum_{i=1}^{N} \mathrm{val}_i}{N} = \frac{\mathrm{val}_1 + \mathrm{val}_2 + \mathrm{val}_3 + \cdots + \mathrm{val}_N}{N} \tag{8}$$

Using Eq. (9), one can determine the covariance between the two features.

$$\mathrm{cov}(f, tg) = \frac{1}{N} \sum_{i=1}^{N} (f_i - \mu_f)(t_i - \mu_{tg}) \tag{9}$$

All the equations presented above are used to find the optimized feature set and for ranking the features.

3.2.2. GI (Gini index)

In this phase, GI is applied to rank and then select the optimal feature subset. GI is an impurity split method, and it is advisable for a binary system, continuous numerical values, etc. The GI of feature A and the collection of patterns N is defined as

$$\mathrm{GI}(N, A) \equiv \mathrm{Gini}(N) - \sum_{v \in \text{ value } (A)} Prob_v \mathrm{Gini}(N_v) \tag{10}$$

where

$$\mathrm{Gini}(N) = 1 - \sum_{v \in \text{ Value } (N)} (Prob_v)^2 \tag{11}$$

Where values (A) is the set of all possible values which are taken by the feature A, $\mathrm{Prob_v}$ shows the probability of patterns N belonging to the value v concerning the feature A, Nv is a subset of patterns N for which feature A has value v, and GI is also known as the Gini impurity which is used to compute the importance of features.

It calculates the amount of probability of specific features that are not classified correctly when chosen randomly. The value of GI varies between 0 and 1, where 0 represents the purity of classification means that the entire element exists in a specified class and 1 represents the random distribution of elements across different classes. In designing a decision tree, the least value would be considered for feature possessing [24].

The GI is determined by subtracting the sum of squared probabilities of each class from one, mathematically, GI can be represented as where Pi is a probability of an element being classified for a different class.

3.2.3. MI (Mutual information)

In this work, the mutual information filter-based FS method is used. MI is a measure between two given random features V1 and F2 that estimates the amount of information obtained about feature V1, through feature F2. The MI function is applied by using Eq. (12)

$$I(V1; F2) = \int_{V1} \int_{F2} p(V1, F2) \log \frac{p(V1, F2)}{p(V1)p(F2)} dV1 dF2 \tag{12}$$

where p (V1, F2), is the joint probability function of features V1 and F2. p(V1) and p(F2) are the marginal density function. In the case of FS, we would maximize the MI between selected features subset Sm and target variable F2.

$$\widetilde{M} = \arg\max_{M} I(S_m; \mathrm{F2}), \mathrm{s.t.} |M| = N \tag{13}$$

Where a number of features are represented by N, we want to select N. All these operations will be performed on the training dataset in order to get an optimal feature set [25].

3.2.4. Boruta feature selection

This is a wrapper-based FS algorithm that is placed under the random forest classification technique. It makes use of shadow features, replicas of the original features. DTs are generated based on the shadow features which are randomly given to objects. The significance of an attribute is determined by how much the classification model's performance suffers when attributes are given to objects at random. Then, the standard deviation (SD) and the mean (M) of the ACC loss is calculated, and the Z-scores are obtained by dividing the SD and its M. The Z-score is regarded as the key metric in this algorithm. In order to determine the significance of original attributes, the set of the importance of shadow attributes is used as a reference. The relevance of original features is then compared with the significance of shadow features, which is the highest [26,27].

The following steps of the Boruta algorithm are as follows:

(1) The generation of the shadow attributes extends the information system.
(2) The Z-scores are calculated after the RF algorithms have been run on each copy of the new dataset.
(3) The shadow attributes' maximum Z-score is calculated.
(4) The maximum Z-score is used to compare the significance of each attribute.
(5) The attributes whose value is significantly higher, than the maximum Z-score are retained (considered important), and the attributes whose importance is significantly lower than the maximum Z-score are eliminated (considered unimportant).
(6) After eliminating all shadow attributes, the process is repeated until all traits have been given equal importance.

Comparing this algorithm to other feature selection algorithms reveals its advantages. In contrast to most other algorithms, which rely on a small feature subset to select a classifier with the minimum level of error, this algorithm adopts an all-relevant variable selection approach, which means it takes into account all attributes that are related to the output variable. Additionally, it may look into interactions between variables and takes into account multivariable connections.

4. Isolation forest method for outlier detection

An observation point that is far off from the majority of observations is called an outlier. Identifying and removing outliers from a dataset is accomplished through the process of outlier detection. The main advantage of removing outliers is that accuracy will be improved. As far as we know, the outlier removal method was not applied in any of the failure detection studies. As a result, we have focused on how outlier detection methods can improve the detection model's accuracy for network and device failure. The iForest (Isolation forest) outlier detection technique is applied in this study. By developing isolation trees and handling outliers as points with a short average length inside the isolation tree, iForest differentiates between outliers. As an outlier detection technique, iForest is employed in numerous studies such as fault detection in the artificial pancreas [28]. In Section 5, we have described how the hybrid-FS method is used to obtain the optimal set of features (1770, 195). We found 19 anomalies in KDDI dataset after applying the iForest outlier detection method. After removing these anomalies, we now have a new feature set that is (1751, 195).

5. SMOTETomek method for imbalance dataset

SMOTE specifies the nearest neighbors (NN) for each minority sample $a_i \in a_{min}$. For generating a synthetic data model a_{new} for a_i; SMOTE randomly chooses an element $\hat{a}_i$ in a_{min}. The feature vector of a_{new} is the sum of feature vectors of a_i and the value which is calculated by multiplying ($\hat{a}_i$ - a_i) with a random value θ, where $0 < \theta < 1$. In this way, we reach a synthetic point along the line segment joining up a_i and $\hat{a}_i$. SMOTETomek is an extension of SMOTE in which Tomek links are

Algorithm 2
SMOTETomek method for handling data imbalance.

Input: Imbalance Training Data
Output: Balance Training Data
From the minority classes, select a_i randomly.
Obtain a_i the nearest neighbour
Generate $a_{new} = a_i + (\hat{a}_i - a_i) \times \theta$
if the ratio of balance is not satisfied repeat step 1
else
Filter out noise from the sample by applying TOMEK
TOMEK (m, n)
{
n's nearest neighbour is m….
The nearest neighbour of m is n.
m and n belong to distinct classes.
}
End

Table 1
Result of the SMOTETomek method.

Type Number	Type Name	Before SMOTETomek	After SMOTETomek
57	Packet loss and delay	1027	1027
9	BGP injection	246	1014
11	BGP hijack	225	1010
3	Interface down	181	1002
1	Node down	72	990

applied to filter out the noisy data. The TOMEK links are defined as; m and n are the two instances. 'm' is the NN of n where n is the NN of m, and, m and n belong to separate classes [29,28]. Algorithm 2 shows the working of the SMOTETomek method. This study examined datasets with both balanced and imbalanced class distributions. Imbalanced data occurs when one class has a significantly greater number of instances than the other classes. This imbalance has a direct negative impact on classification performance because most ML models assume a balanced distribution of classes in the data. Therefore, it is essential to use appropriate evaluation metrics to assess and compare classifiers' predictive ability in the case of imbalanced data. While accuracy is commonly used in machine learning studies, it can be a misleading metric when the majority class is used as the prediction for any given instance in an imbalanced dataset. In addition to accuracy, it is important to consider other metrics to evaluate a classifier's ability to distinguish one class from another, even when the classes are imbalanced [21]. Consequently, evaluation metrics such as F1-score, precision, and detection rate, which are based on the classes' performance are selected alongside accuracy in the proposed failure detection framework's performance evaluations. Table 1 shows the results of the balancing method.

6. Ensemble model for detection technique

Three well-known ensemble machine learning models xGBoost, LGBM, and, RF are applied in the proposed detection system. Multiple base estimators' anticipated outcomes are combined through ensemble approaches. Therefore, the outcomes are better as compared to some individual estimators. The random forest includes a technique that considers the outcomes of many individual learners and combines their outcomes using the average. Likewise, xGBoost and LGBM include techniques that combine many weak learners to get a significant outcome from an ensemble [4]. In this study, we have considered these models to prevent overfitting by simplifying the objective function that combines predictive and regularization terms and maintaining optimal computational speed.

1 ***Random Forest:*** To overcome the limitation of a decision tree, a random forest ensemble classifier is used. It consists of a large number of individual DTs (Decision trees) that operates as an ensemble. By applying a random sampling with a replacement method variety of small data subsets (bootstrap sample) is generated from the real training data. A significant number of tiny samples are frequently drawn with replacement from a single original sample using this resampling technique. Bootstrap samples are used to train each decision tree separately. The outcome of the ensemble model is determined by calculating the majority vote from all decision trees. OOB score (Out of the bag) is applied to verify the random forest model. The model's accuracy depends on the number of variables available for dividing at each tree node [30,31].
2 ***xGBoost:*** Extreme gradient boosting is an implementation of the GB (gradient boosting) method that uses the 2nd-order derivative of the ***regularization***, loss function to obtain a more precise approximation. In addition to preventing overfitting, it also optimizes computing resources. This is obtained by simplifying the objective functions that integrate regularization terms and predictive while also retaining an optimal computational speed. Regularization serves as a penalty that punishes complex models and rewards simplified models. xGBoost uses the two regularization parameters L1 and L2 (selection operator and ridge regression respectively). The objective is to offer less computational complexity because it will reduce bias as well as prevent overfitting [32].

The objective function (regularization and loss function) at iteration k that we need to minimize is the following-

$$Objective = \sum_{j=1}^{N} Loss\left(y_j, \hat{y}j\right) + \sum_{k=1}^{M} \Omega(f_k), f_k \in F$$
$$where\, \Omega(f) = \gamma T + \frac{1}{2}\lambda \| \omega \|^2 \quad (14)$$

Where the first term Loss shows the Loss function and the second term Ω shows the regularization term. regularization parameter is the λ, the vector of the score isω in the leaves, to further partition the leaf node minimum loss required is γ, and the number of leaves is denoted by T.

3 ***LGBM:*** A gradient boosting framework that makes use of a tree-based learning algorithm is called LGBM (Light gradient boosting machine). It offered to answer the problem of gradient-boosting decision trees (GBDT) in large data. The major differentiation is that the DT in LGBM is developed leaf-wise, necessitating the verification of all preceding leaves prior to each new leaf in order to increase accuracy and avoid overfitting. Moreover, LGBM employs an improved histogram to recognize the optimal segmentation point. The histogram algorithm can successfully prevent overfitting by having a regularization impact. This algorithm employs a leaf-wise generation approach to reduce training data. When developing the same leaf, the leaf-wise strategy can reduce additional losses compared to the conventional method like level (depth)-wise. Additionally, the additional parameter is utilized to restrict the decision tree's depth and prevent overfitting [33,9].

7. Results and discussion

In this section, we include a discussion of the experimental analysis performed using the KDDI dataset [4]. Python programming language is used to implement proposed experiments. To implement the machine learning algorithm, Scikit learn library is used [34]. The experiment is performed on an IntelXeonR CPU E3–1240v6 operating at 3.70 GHz with 16 GB RAM and a 1 TB HDD. To systematically evaluate the efficiency of the proposed detection system, three ensemble models are used, such as LGBM, xGBoost, and RF. Further, each model is trained

Table 2
Performance metrics of different ensemble machine learning models.

Model No.	Machine learning models	Evaluation metrics after SMOTETomek (%)				Evaluation metrics before SMOTETomek (%)			
		ACC	DR	PR	F1-Score	ACC	DR	PR	F1-Score
1.	RF	97.4	97.5	97.5	97.5	96.0	93.0	98.4	95.2
2.	xGBoost	98.1	98.2	98.1	98.2	95.4	92.9	97.5	95.4
3.	LGBM	98.4	98.5	98.4	98.5	96.0	93.4	98.2	95.5

using a 10-fold CV technique [35]. The performance of the proposed system is compared with some existing techniques in a similar scenario, such as [4]. The FDF- HybridFS outperforms in terms of ACC and has a high DR for different failure types.

7.1. Description of the KDDI corporation dataset

This sub-section briefly discussed the analysis of the proposed FDF-HybridFS framework using the KDDI dataset. The proposed model is explained step by step by applying the KDDI dataset.

The dataset utilized in this work was generated in the NFV-based test environment simulated for a commercial IP core network that connects the 5th generation core network for internet connectivity. In this sense, synthetic data is analogous to real data resulting from the NFV-based test environment [4,5]. The dataset consists of labels for normal or abnormal traffic, CPU usage ratios, memory usage ratios, and border gateway route information. This dataset is taken from [4] the dataset consists of 1770 instances and 997 features (1770, 997) extracted from the 28 GB per day of an unstructured log file. In this experiment, the dataset contains three types of failure scenarios: network element failure, interface failure, and route information failure. In a network element, a failure scenario is considered the type name “node down”, interface failure is considered the type name “interface down” and “packet loss and delay”, and route information failure is considered the type name “BGP injection” and “BGP hijack.” In order to evaluate the performance of the proposed detection system, the KDDI dataset is divided into training and testing sets in an 80:20 ratio. Description of different categories of labels present in the KDDI dataset.

- Node-down: This process causes the virtual machine to stop running completely, which leads to conditions where the virtual network function is not available. This kind of failure could be caused by a power-related issue [5].
- Interface down: This recipe causes one of the interfaces on a virtual network function to go down, causing the packet to be inaccessible. Such a failure could be caused by a loose cable connection [5].
- Ixnetwork-BGP-injection: This shows the anomaly injection route from another SP [4,36].
- Ixnetwork-BGP-hijacking: This shows the hijack of the origin route by another SP [4,36].
- Packet loss & delay: Cause the delay & loss on an interface [4].

7.2. Evaluation of feature selection method

The KDDI data set contains 1770 instances and 997 features. During the preprocessing step, it was discovered that the value of some features for each entry in the dataset is zero. As a result, features that contain zero values are eliminated from the dataset. The updated dataset contains 1770 instances and 511 features. Our proposed model used a three-filter-based method such as CC, GI, and MI, and one wrapper-based Boruta feature selection method. First, CC is applied to measure the correlation score for each feature present in the dataset. The 254 optimal features are selected by CC; the 201 features are selected by the Gini Index; and the 199 features from the 997 are selected by mutual information. The new feature sets generated by CC, GI, and MI are Fsop1 (1770, 254), Fsop3 (1770, 201), and Fsop2 (1770, 199), respectively. In conclusion, using the filter-based FS approach, only those attributes are kept that were included in Fsop1, Fsop2, and Fsop3 feature sets. In other words, a union set operation is performed for these sets. The new feature set is generated after using the filter-based feature selection method, FS_{opf} (1770, 401). We found that out of the 510 features, 109 features are repeated and not relevant. Therefore, after applying CC, GI, and MI, the obtained feature set (FS_{opf}) is fed to the Boruta feature selection method. As a result of using the wrapper-based methods, we are able to obtain the optimal feature set, which is $OP_{FS'}$ (1770, 195). We conclude that out of the 997 features, only 195 are the most relevant.

7.2.1. Mathematical representation for features

In this work, Union (U) operations are used for combining the features subset. The mathematical term is represented by

Let $F = \{f_1; f_2; f_3 \ldots fn\}$ be the original feature sets.

FSop1= $\{f_{11}, f_{12}, f_{13} \ldots fp\}$ is a subset of selected features with the first model (CC), where fp indicates that p number of features are selected with the CC method and $p < F$

FSop2 = $\{f_{21}, f_{22}, f_{23} \ldots fq\}$ is a subset of selected features with the second method (MI), where fq indicates that q number of features are selected with the MI method and $q < F$

FSop3 = $\{f_{31}, f_{32}, f_{33} \ldots fr\}$ is a subset of selected features with the third method (GI), where fr indicates that r number of features are selected with the Gini index method and $r < F$.

Union operation (∪)

FSop1 is the feature set selected by the CC model. It contains p number of features. FSop2 is another feature set selected by the MI model, which contains the q number of features, and FSop3 is a set of features selected by the GI model, which contains the r number of features. To create an optimal feature set $OP_{FS'}$ with a union operator, we combine all three selected feature subsets FSop1, FSop2, and FSop3.

FSopf= FSop1∪ FSop2 ∪ FSop3.Created feature subset FSopf contains the optimized feature set where FSopf< F.

7.3. Description of evaluation metrics

(a) ACC (Accuracy)- It calculates the fraction of instances that the system correctly detected to all of the observations in the test set. In order to measure the model’s accuracy, it considers both true positives and true negatives [37,38].

$$Accuracy = \frac{True\ Positive + True\ Negative}{True\ Positive + False\ Negative + False\ Positive + True\ Negative} \quad (15)$$

Table 3
Overall Performance comparison of machine learning model After data balancing.

Model with SMOTETomek	Random Forest			xGBoost			LGBM		
Failure criteria	PR	DR	F1-score	PR	DR	F1-score	PR	DR	F1-score
Packet delay(loss):57	0.94	0.95	0.94	0.96	0.96	0.96	0.97	0.96	0.96
Interface down: 3	0.97	0.97	0.97	0.98	0.99	0.99	0.98	**1.00**	0.99
Injection BGP:9	0.97	**1.00**	0.99	0.98	**1.00**	0.99	0.98	**1.00**	0.99
Hijacking BGP:11	0.99	0.95	0.97	0.98	0.96	0.97	0.99	0.97	0.98
Node down: 1	1.00	**1.00**	1.00	1.00	**1.00**	1.00	1.00	**1.00**	1.00

Table 4
Overall Performance comparison of the machine learning model before data balancing.

Model without SMOTETomek	Random Forest			xGBoost			LGBM		
Failure criteria	PR	DR	F1-score	PR	DR	F1-score	PR	DR	F1-score
Packet delay(loss):57	0.94	**1.00**	0.97	0.94	0.99	0.96	0.94	**1.00**	0.97
Interface down: 3	1.00	0.95	0.98	1.00	0.88	0.94	1.00	0.95	0.98
Injection BGP:9	0.98	0.98	0.98	1.00	0.98	0.99	1.00	0.98	0.99
Hijacking BGP:11	1.00	0.72	0.84	0.94	0.79	0.86	0.97	0.74	0.84
Node down: 1	1.00	**1.00**	1.00	1.00	**1.00**	1.00	1.00	**1.00**	1.00

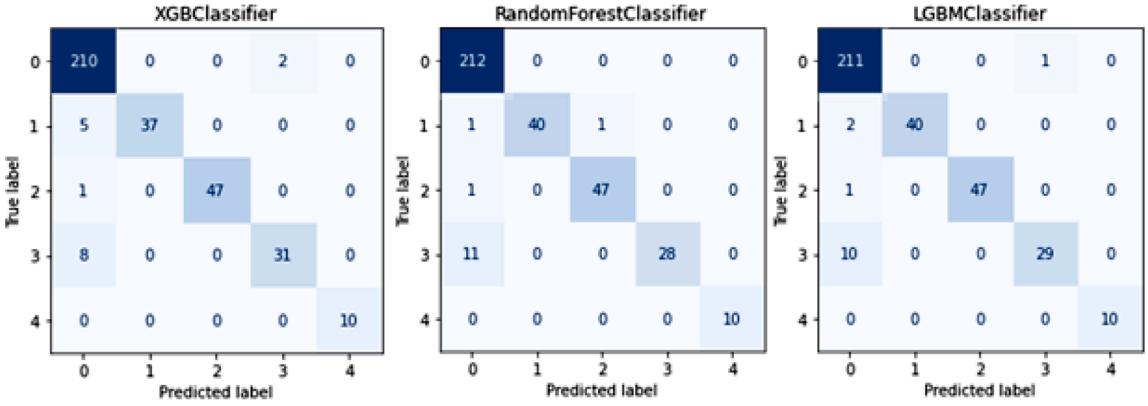

FIg. 3. Confusio Matrix for xGBoost, RF and LGBM after balancing.

(b) Detection Rate - DR calculates the ratio of the total number of activities included in the test set to the number of abnormal tasks detected by the models. Recall is another term for the detection rate [37].

$$Detection\ Rate = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{16}$$

(c) F1 score- It determines the weighted average of precision and detection rate. Since it calculates both false positives and false negatives, accuracy is less important when the class distribution is imbalanced [37].

$$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{17}$$

(d) PR (Precision) – It calculates the number of abnormal activities detected, divided by the number of observations the model considers a failure [37,39].

$$Precision = \frac{True\ Positive}{True\ Positive\ False\ Positive} \tag{18}$$

7.4. Result analysis

This section represents the results obtained by the different ensemble models. Table 2 shows the performance metrics of ensemble machine-learning models. LGBM shows the best accuracy performance, then the xGBoost. Random forest relatively shows lower accuracy compared to other models. The result showed the outstanding performance of the ensemble machine learning model. Tables 3 and 4 show the PR,

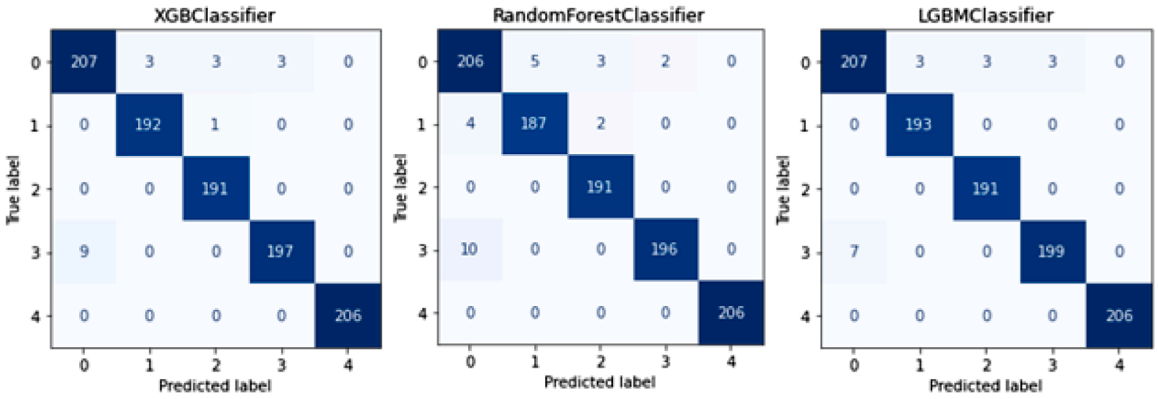

FIg. 4. Confusio Matrix for xGBoost, RF and LGBM after balancing.

Table 5
Analysis of accuracy & F1 Score with some of the existing ML methods for the KDDI testing dataset.

Refs.	Method	Year	FS techniques	Dataset	ACC (%)	F1 Score (%)
Xia et al. [4]	xGBoost	2021	xGBoost and RF Feature Importance	KDDI Dataset	93.6	95.0
Xia et al. [4]	LGBM	2021	xGBoost and RF Feature Importance	KDDI Dataset	93.3	94.6
Xia et al. [4]	RF	2021	xGBoost and RF Feature Importance	KDDI Dataset	92.7	93.8
Xia et al. [4]	MLP	2021	xGBoost and RF Feature Importance	KDDI Dataset	81.3	80.8
Xia et al. [4]	Decision Tree	2021	xGBoost and RF Feature Importance	KDDI Dataset	80.9	84.2
Xia et al. [4]	SVM	2021	xGBoost and RF Feature Importance	KDDI Dataset	79.0	79.4
Our proposed FDF-HybridFS	RF	2023	Hybrid Feature Selection	KDDI Dataset	97.4	97.5
	xGBoost	2023	Hybrid Feature Selection	KDDI Dataset	98.1	98.2
	LGBM	2023	Hybrid Feature Selection	KDDI Dataset	98.4	98.5

detection rate, and F1 score results of the machine learning model before and after applying the data balancing technique on different failures. After data balancing, it can be seen that node-down type 1 failure can be well predicted with all machine learning models with 100 percent precision and recall rate. Type 9 injection BGP failure can be well predicted in each model with a 100 percent recall rate and achieves a 99 percent F1 score. With LGBM, type 3 interface down failure achieves 100 percent recall. It has been observed that LGBM performance is better than xGBoost and RF in terms of all performance measures after applying the SMOTETomek sampling method. Similarly, before data balancing, it has been noticed that each machine learning model can accurately predict type 1 node down failure with 100 percent precision and recall rate. Type 57 packet delays and loss failure achieve a 100 percent recall rate with LGBM and random forest and 99 percent with xGBoost. It has been observed that overall LGBM performance is better than xGBoost and RF in terms of all performance measures such as PR, accuracy, detection rate, and f1-score before and after applying the SMOTETomek sampling method. In order to evaluate the performance of models, we have used a confusion matrix to visualize the performance of the predictive model and observe confusion between two labels [23,40]. Rows in the matrix represent the true labels, while columns represent the predicted labels. The diagonal of the matrix shows the correct prediction, whereas the value outside the diagonal shows a misclassified prediction. Figs. 3 and 4 show the confusion matrix for xGBoost, RF, and LGBM before and after the balancing method. In the confusion matrix, labels 0, 1, 2, 3, and 4 represent packet loss and delay, interface down, BGP injection, BGP hijacks, and node down, respectively. Table 5 shows the ACC analysis of the proposed FDF-HybridFS with some of the existing models.

8. Conclusion

This study presented a novel FDF-HybridFS system for detecting device and network failures in IP networks that connect the 5GC networks using an NFV-based test environment. This system uses the new hybrid feature selection technique and machine learning models. The proposed FDF-HybridFS system works in various stages. The model's initial stage includes preprocessing steps to convert categorical values into numerical values and normalize network traffic to a certain scale. In the second stage, the new hybrid feature selection method using filter-based FS methods is applied to select and rank features using the CC, GI, and MI features, which are concatenated using an appropriately designed mechanism, and then merged features are fed into the one wrapper-based Boruta feature selection method. Finally, the well-known tree-based machine learning algorithms RF, xGBoost, and LGBM are deployed as analytical tools that provide effective decision-making characteristics in the next-generation network during the final stage of data analysis. The novel failure detection framework's performance is evaluated and compared with the same recent state-of-the-art method using the KDDI dataset. The extensive outcome confirms the proposed failure detection system's advantages in terms of precision, detection rate, F1 score, and accuracy over the existing detection method. Overall, the proposed system has achieved a 100% detection rate for three out of five types of failure while using a well-balanced dataset; for the remaining two types, the detection rate reached up to 96 −97%. Using LGBM, the system outperforms other methods in terms of overall accuracy and other metrics. Furthermore, the performance of the proposed failure detection system is better than the other existing related work, indicating an improvement in performance measures. Using analytical

tools such as Apache Hadoop, we intend to extend the proposed framework in the future to a real network internet traffic streaming environment.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] Communicationtoday,"5 gtechnologies market to grow" april 2022 <https://www.communicationstoday.co.in/5 g-technologies-market-to-grow-to-99-76-billion-by-2026/>.
[2] Cérin, C., et al. "Downtime statistics of current cloud solutions." International Working Group on Cloud Computing Resiliency, Tech. Rep 1 (2013): 2.
[3] Nokia "IP networking for the 5G era" 2020 <https://www.nokia.com/blog/>.
[4] Fei, X., et al. "Analysis on route information failure in IP core networks by NFVbased test environment." (2021).
[5] J. Kawasaki, G. Mouri, Y. Suzuki, Comparative analysis of network fault classification using machine learning, in: Proceedings of the NOMS 2020IEEE/IFIP Network Operations and Management Symposium, IEEE, 2020.
[6] V. Tong, et al., Machine learning based root cause analysis for SDN network, in: Proceedings of the IEEE Global Communications Conference (GLOBECOM), IEEE, 2021.
[7] N. Nahar, et al., Feature selection based machine learning to improve prediction of parkinson disease, in: Proceedings of the International Conference on Brain Informatics, Springer, Cham, 2021.
[8] X.F. Song, et al., A fast hybrid feature selection based on correlation-guided clustering and particle swarm optimization for high-dimensional data, IEEE Trans. Cybern. (2021).
[9] R.K. Batchu, H. Seetha, A generalized machine learning model for DDoS attacks detection using hybrid feature selection and hyperparameter tuning, Comput. Netw. 200 (2021), 108498.
[10] M. Vásquez-Bermúdez, et al., Analysis of a network fault detection system to support decision making, in: Proceedings of the International Conference on Technologies and Innovation, Springer, Cham, 2017.
[11] I. Katzela, M. Schwartz, Schemes for fault identification in communication networks, IEEE/ACM Trans. Netw. 3 (6) (1995) 753–764.
[12] AT. Bouloutas, GW. Hart, M. Schwartz, Fault identification using a finite state machine model with unreliable partially observed data sequences, IEEE Trans. Commun. 41 (7) (1993) 1074–1083.
[13] CS. Hood, C. Ji, Proactive network-fault detection [telecommunications], IEEE Trans. Reliab. 46 (3) (1997) 333–341.
[14] İ.F. Kilinçer, et al., Automatic fault detection with Bayes method in university campus network, in: Proceedings of the International Artificial Intelligence and Data Processing Symposium (IDAP), IEEE, 2017.
[15] C. Sauvanaud, et al., Anomaly detection and root cause localization in virtual network functions, in: Proceedings of the IEEE 27th International Symposium on Software Reliability Engineering (ISSRE), IEEE, 2016.
[16] M. Ruiz, et al., Service-triggered failure identification/localization through monitoring of multiple parameters, in: Proceedings of the ECOC 2016; 42nd European Conference on Optical Communication, VDE, 2016.
[17] S.M. Srinivasan, T. Truong-Huu, M. Gurusamy, Machine learning-based link fault identification and localization in complex networks, IEEE Internet Things J. 6 (4) (2019) 6556–6566.
[18] R. Stadler, R. Pasquini, V. Fodor, Learning from network device statistics, J. Netw. Syst. Manag. 25 (4) (2017) 672–698.
[19] K. Qader, Mo Adda, M. Al-Kasassbeh, Comparative analysis of clustering techniques in network traffic faults classification, Int. J. Innov. Res. Comput. Communi. Eng. 5 (4) (2017) 6551–6563.
[20] P. Kumar, GP. Gupta, R. Tripathi, Toward design of an intelligent cyber-attack detection system using hybrid feature reduced approach for iot networks, Arab. J. Sci. Eng. 46 (4) (2021) 3749–3778.
[21] V. Kumar, et al., Addressing binary classification over class imbalanced clinical datasets using computationally intelligent techniques, Healthcare 10 (7) (2022).
[22] Y.N. Soe, et al., Towards a lightweight detection system for cyber-attacks in the IoT environment using corresponding features, Electronics 9 (1) (2020) 144 (Basel).
[23] P. Kumar, GP. Gupta, R. Tripathi, Design of anomaly-based intrusion detection system using fog computing for IoT network, Autom. Control Comput. Sci. 55 (2) (2021) 137–147.
[24] S. Tabakhi, P. Moradi, F. Akhlaghian, An unsupervised feature selection algorithm based on ant colony optimization, Eng. Appl. Artif. Intell. 32 (2014) 112–123.
[25] P. Kumar, GP. Gupta, R. Tripathi, A distributed ensemble design-based intrusion detection system using fog computing to protect the internet of things networks, J. Ambient Intell. Humaniz Comput. 12 (10) (2021) 9555–9572.
[26] S. Subbiah, et al., Intrusion detection technique in wireless sensor network using grid search random forest with Boruta feature selection algorithm, J. Commun. Netw. 24 (2) (2022) 264–273.
[27] MAlFA Fida, T. Ahmad, M. Ntahobari, Variance Threshold as Early Screening to Boruta Feature Selection for Intrusion Detection System, in: Proceedings of the 13th International Conference on Information & Communication Technology and System (ICTS), IEEE, 2021.
[28] M.F. Ijaz, M. Attique, Y. Son, Data-driven cervical cancer prediction model with outlier detection and over-sampling methods, Sensors 20 (10) (2020) 2809.
[29] V. Kumar, G. Singh Lalotra, R.K. Kumar, Improving performance of classifiers for diagnosis of critical diseases to prevent COVID risk, Comput. Electr. Eng. 102 (2022), 108236.
[30] V. Svetnik, et al., Random forest: a classification and regression tool for compound classification and QSAR modeling, J. Chem. Inf. Comput. Sci. 43 (6) (2003) 1947–1958.
[31] M.D.A Talukder, et al., A dependable hybrid machine learning model for network intrusion detection, J. Inf. Secur. Appl. 72 (2023), 103405.
[32] T. Chen, C. Guestrin, Xgboost: a scalable tree boosting system, in: Proceedings of the 22nd acmsigkdd International Conference on Knowledge Discovery and Data Mining, 2016.
[33] G. Ke, et al., Lightgbm: A Highly Efficient Gradient Boosting Decision Tree, Advances in neural information processing systems, 2017, p. 30.
[34] F. Pedregosa, et al., Scikit-learn: machine learning in Python, J. Mach. Learn. Res. 12 (2011) 2825–2830.
[35] J.M.N. Gonzalez, J.A. Jimenez, J.C.D. Lopez, Root cause analysis of network failures using machine learning and summarization techniques, IEEE Commun. Mag. 55 (9) (2017) 126–131.
[36] B. Al-Musawi, P. Branch, G. Armitage, BGP anomaly detection techniques: a survey, IEEE Commun. Surv. Tutor. 19 (1) (2016) 377–396.
[37] R. Boutaba, et al., A comprehensive survey on machine learning for networking: evolution, applications and research opportunities, J. Internet Serv. Appl. 9 (1) (2018) 1–99.
[38] N. Islam, et al., Building machine learning based firewall on spanning tree protocol over software defined networking, in: Proceedings of the International Conference on Trends in Computational and Cognitive Engineering: Proceedings of TCCE 2020, Springer Singapore, 2021.
[39] M. Prasad, S. Tripathi, K. Dahal, An efficient feature selection-based Bayesian and Rough set approach for intrusion detection, Appl. Soft Comput. 87 (2020), 105980.
[40] S. Einy, C. Oz, Y.D. Navaei, The anomaly-and signature-based IDS for network security using hybrid inference systems, Math. Probl. Eng. 2021 (2021) 1–10.