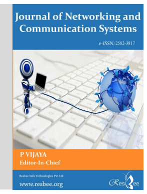
Vol.7 No.1 Jan 2024

Journal of Networking and Communication Systems

Received 2 December, Revised 28 December, Accepted 30 January



A Combined Machine Learning Model for Intrusion Detection in Imbalanced Dataset: A Hybrid Optimization-Incremental Learning Approach

Golla jyothirmai
Both UMKC, Kansas, Missouri, USA
jyothirmai100@gmail.com

Abstract: The issue of class imbalance, on the other hand, has yet to be resolved. In this research work, an intrusion detection model for class imbalance data is introduced by following four major phases: (a) pre-processing, (b) imbalance processing, (c) feature extraction, and (d) intrusion detection phase. The overall architecture of the proposed work is manifested in Fig.1. Initially, the collected class imbalance data is pre-processed via data cleaning and a data standardization approach. Then, the pre-processed class imbalance data is balanced by an improved over-sampling technique using SMOTE and the Multi-kernel FCM clustering model. Subsequently, multi-features like Exponential Moving Averages (EMA), Double Exponential Moving Averages (DEMA), Improved weighted holoentropy (IWH), and statistical features (mean, median, standard deviation, percentile, and moment) are extracted from these balanced data. The extracted overall features are given for K-fold validation. Then, the intrusion detection phase is modeled with a combination of four individual machine learning models, such as Support Vector Machine (SVM), Deep Belief Network (DBN) with incremental learning, Naïve Bayesian (NB), and Random forest (RF) to achieve the utmost detection performance. Each of the classifiers is trained with the acquired K-fold data features. Moreover, to achieve the utmost detection accuracy and better tradeoff performance, the weight of DBN is fine-tuned by a new Self improved Seagull optimization algorithm (SI-SOA), which is an improved version of the standard Seagull optimization algorithm (SOA). Then, the logoff performance is computed from the outcome acquired from each of the individual classifiers (SVM, DBN, NB, and RF). Finally, the log computed result will portray the detected attacks in the network.

Keywords: Intrusion Detection; Imbalanced Dataset; Improved Weighted Holoentropy (IWH); Incremental Learning; Optimized DBN; SI-SOA.

Nomenclature

Abbreviation	Description
SMOTE	Synthetic Minority Over-Sampling Technique
ROC	Receiver Characteristics Operating
GAN	Generative Adversarial Network
FNN	Feed-forward Neural Network
A-NIDS	Anomaly-based Network Intrusion Detection System
b-XGBoost	binary eXtreme Gradient Boosting
GMM	Gaussian Mixture Model
ICT	Information and Communications Technology
IGAN	Imbalanced Generative Adversarial Network
DNN	Deep Neural Network
EMA	Exponential Moving Averages
IDS	Intrusion Detection Systems
DM	Data Mining
IWH	Improved Weighted Holoentropy
CNN	Convolutional Neural Network
DBN	Deep Belief Network
SVM	Support Vector Machine
DEMA	Double Exponential Moving Averages
NB	Naïve Bayesian
RBF	Radial Basis Function
IDS	Intrusion Detection System
MA	Moving Average
DNN	Deep Neural Network
FCM	Fuzzy C-Means
AUC	Area Under Curve
U2R	User To Root
LSTM	Long Short-Term Memory

https://doi.org/10.46253/jnacs.v7i2.a5

ML	Machine Learning
I-OVO	Improved One-Vs-One
R2L	Remote To Local
DT	Decision Trees
DL	Deep Learning
MKFC	Multiple Kernel Fuzzy C-means
RF	Random Forest
CD	Contrastive Divergence
MSE	Mean Squared Error
GRU	Gated Recurrent Unit
RNN	Recurrent Neural Networks

1. Introduction

Many small and large businesses have been using internet services to carry out their everyday operations in recent decades [1]. The creation of applications operating on multiple platforms provides an impetus to the security of the web server, despite the immense rise in the use of computers via networks and information sharing. Malicious cyberattacks on web servers are becoming incredibly popular. Nevertheless, due to its immense volume, physically monitoring network traffic seems to be impractical. Generally, an attack is defined as any access to the network that compromises the data's availability, privacy, or consistency [2] [3]. [4] [5]. The number of security risks gets raised as the number of internet applications expands [6] [7] [8] [9] [10]. IDS [4][5] is indeed a system that automatically monitors network traffic to distinguish malicious assaults from regular traffic. ML algorithms can be used to efficiently process large amounts of data. The benchmark dataset NSL KDD [6] [11] [12] is used to evaluate IDSs. A classification problem may well be described as the procedure of accurately recognizing the intrusions from network traffic. A classification method must be designed to successfully detect intrusions with fewer false alarms and increase detection performance [21][22][23][24].

Many researchers suggested many ML methodologies and techniques for formulating an efficient IDS, including DT [7], SVM [8], AdaBoost [9], KNN [10], NB [11], and so on. When compared to normal traffic, the number of attacks is a bit less. Conventional classifiers are ineffective in identifying the sorts of attacks. A class Imbalance Problem occurs when the quantity of regular traffic exceeds that of attack traffic. [13], [14], [15], [16], [17]. Because of their low representation in comparison to the majority class (normal class), the instances belonging to the class of interest, i.e., the minority class (attack class), are frequently overlooked [25], [26], [27], [28]. Despite the great accuracy of this technique, the IDS is defunct without effective protection against attack traffic [18], [19], [20]. Hence, specialized techniques, which provide due importance to the minority class are required.

1.1 Data Level Approach for dealing with unbalanced data: Resampling Techniques

When dealing with unbalanced datasets, techniques such as enhancing classification algorithms or balancing classes throughout the training data (data preparation) must be used even before data is fed into the ML techniques. The latter method is recommended since it is more widely applicable. The basic goal of class balancing would be to increase the underrepresented class's occurrence while lowering the dominant class's occurrence. This one is done to ensure that each classes have a roughly equal number of incidents. The following resampling methods are commonly utilized:

1.1.1 Random Under-Sampling:

By randomly removing the majority class instances, Random Under sampling tries to balance the class distribution. This process is repeated until the overwhelming as well as minority class occurrences become equal.

Advantages

When the training data set is large, this can assist reduce run time and storage concerns by decreasing the amount of training data samples.

1.1.2 Random Over-Sampling

Over-sampling is a technique for increasing the number of examples in the minority class by randomly repeating them to offer a more representative sample of the minority class.

Advantages

Unlike sampling, this approach does not result in information loss

1.1.3Cluster-Based Over Sampling

The K-means clustering methodology is used for minorities and dominant class instances independently. This is done to find clusters inside the data. Following that, each cluster is oversampled to ensure that all clusters have the same class and instances with the same size.

Advantages

- This clustering approach aids in overcoming the issue of class imbalance. Where the number of positive class examples differs from the number of negative class instances.
- Overcome obstacles inside class imbalance, when a class is made up of many sub-clusters. And there aren't the same amount of instances in each sub-cluster.

1.1.4 Informed Over Sampling: SMOTE for imbalanced data

This is utilized to overcome the over-fitting issue. As an example, a sample of information from the minority class is collected, and new synthetic comparable cases are generated. The original dataset would then be supplemented with these synthetic examples. The fresh dataset is used to train the classification techniques as a sample.

Advantages

- Because synthetic examples are created rather than replications of real cases, the problem of over fitting induced by random oversampling is mitigated.
- There will be no information loss.
- In the current research work, the Data Level approaches for class imbalance problems are put together, therefore the proposed work becomes much more suitable to overcome the class imbalance problems in the Intrusion Detection.
- The major contribution of this research work is:
- Improved weighted holoentropy is also determined along with the statistical and other technical indicators.
- Introduces an optimized DBN model to detect the presence of intruders in the network, where the weights are optimally tuned by a new improved algorithm.
- Proposes a SI-SOA for solving the optimization issue.

The remaining section of this paper is organized as: Section 2 addresses the recent works undergone in literature regarding this subject. Section 3, Section 4 and Section 5 depict about proposed IDS in the imbalanced dataset, proposed IDS in the imbalanced dataset, and imbalance processing: improved over sampling technique with SMOTE and multi-kernel FCM clustering model respectively. The multi-feature extraction, k-fold validation, and intrusion detection phase are portrayed in Section 6, Section 7, and Section 8. The results acquired with the projected model are discussed comprehensively in Section 9. This paper is concluded in Section 10.

2. Literature Review

2.1Related Works

In 2020, Zhang *et al.* [1] proposed a class imbalance processing technology for the large-scale dataset, referred to as SGM, which combines SMOTE and under-sampling for clustering based on GMM. Researchers subsequently developed SGM-CNN, a flow-based intrusion detection model that incorporates unbalanced class processing with a CNN, and examined how varying amounts of convolution kernels and learning rates affect the performance of the model. The UNSW-NB15 and CICIDS2017 datasets were used to verify the effectiveness of the suggested model.

In 2022, Jiyuan Cui *et al.* [46] have based on GMM-WGAN-IDS. This method improved the accuracy of minority class detection and reduced false alarm rates in network data that had high dimensions, complexity, and imbalance issues. There were three parts: the SAE, GMM-WGAN, and CNN LSTM. By stacking numerous auto encoders, SAE was able to extract the best low-dimensional characteristics from high-dimensional data. GMM-WGAN reduced the imbalances between the majority class under sampling in the minority class generation and the GMM clustering algorithm using the same model's Wasserstein GAN technique. CNN LSTMs were used to maximize performance by merging CNN and LSTM networks for the classification module. Experiments were conducted on the UNSW NB15 and NSL KDD datasets, showing better results than existing methods, and confirming the effectiveness of the system.

In 2022, Balyan *et al.* [47] focused on developing an effectual IDS system. Due to the rapid development of IT technology, these IDS utilized ML techniques to track down suspicious network behavioral patterns and identify harmful intrusions in digital data. To deal with the issue of a limited training dataset, this research enhanced the genetic algorithm and particle swarm optimization (EGA-PSO), as well as improved RF. EGA-PSO helped enhance minor datasets by selecting the best features for better fitness outcomes, while IRF eliminated less significant attributes from each iterative process. Performance tests were conducted on NSL_KDD benchmark datasets, where the HNIDS achieved higher accuracy than other ML models such as SVM, RF LR, etc.

In 2022, Le *et al.* [48] have introduced the advanced digital technology and IIoT. It also talked about the IDS, a safety solution that helped to detect abnormal behavior in networks to avoid attacks. ML-based IDS models were used for building secure applications, but they suffered from reduced accuracy due to imbalanced multiclass IIoT datasets. This used the highly gradient boosting model to run an IDS on two contemporary imbalanced IIoT datasets, TON IOT, and X-IIotDS, with outstanding attack detection results attained by F1 scores.

In 2021, Andresini *et al.* [2] utilized a DL methodology for the binary classification of network traffic. The fundamental concept was to describe network flows as 2D pictures and train a GAN and a CNN using this imagery representation of network traffic. By supplementing the training data necessary to develop a CNN-based intrusion detection model, the GAN has been trained to provide fresh images of unexpected network intrusions. Furthermore, GAN-based data augmentation was permitted to focus on the potential unbalance of malicious activity in network traffic, which has been often more uncommon than regular traffic. It's being used to develop a strong intrusion detection model by simulating unexpected assaults. On four benchmark datasets, the suggested technique outperforms competitor IDSs in terms of prediction accuracy.

In 2020, Bedi *et al.* [3] proposed a Siam-IDS, which was a novel IDS based on the Siamese Neural Network (Siamese-NN). Without utilizing standard class balancings approaches like oversampling and random under sampling, the recommended Siam-IDS could recognize R2L and U2R assaults. The performance of Siam-IDS was compared to the current IDSs created using DNN and CNN methods. When compared to its competitors, Siam-IDS was able to obtain greater recall values for both R2L and U2R attack classes.

In 2021, Gupta *et al.* [4] implemented LIO-IDS based on an LSTM classifier and Improved the one-on-one technique for handling both frequent and infrequent network intrusions. LIO-IDS, a two-layer A-NIDS had recognized various network intrusions with high precision as well as minimal computational time. Using the LSTM classifier, Layer 1 of LIO-IDS distinguishes intrusions from regular network data. Layer 2 classifies the detected incursions into attack classes using ensemble methods. At the second layer of the suggested LIO-IDS, this study offers an I-OVO approach for performing multi-class classification. Unlike the standard OVO method, the suggested I-OVO method tests each sample with only three classifiers greatly decreasing the testing time. Further, to improve the detection capabilities of the proposed LIO-IDS, oversampling techniques were utilized at Layer 2. For the NSL-KDD, CIDDS-001, and CICIDS2017 datasets, the proposed system's performance was assessed in terms of Accuracy, Recall, Precision, F1-score, ROC curve, AUC values, training time, and testing time.

In 2021, Huang *et al.* [5] used IGAN to tackle the class imbalance problem. The major uniqueness of the model was that it supplements the traditional GAN with an unbalanced data filter and convolutional layers, resulting in additional primary indicators for minority classes. Furthermore, employing the samples created by IGAN, an IGAN-based IDS, dubbed IGAN-IDS, has been built to deal with class-imbalanced intrusion detection. IGAN-IDS has been made up of three different modules: feature extraction, IGAN, and DNN. To begin, raw network characteristics were transformed into feature vectors using a FNN. After that, the IGAN creates fresh samples inside the spatial domain. Finally, the final intrusion identification was carried out by the DNN, which would also be made up of convolutional and fully-connected layers.

In 2021, Bedi *et al.* [6] Improved Siam-IDS (I-SiamIDS), a two-layer ensemble for managing class imbalance problems was presented as an algorithm-level method. I-SiamIDS does not use data-level balancing approaches to identify both majority and minority classes at the algorithmic level. To identify the intrusions, the first layer of I-SiamIDS implements hierarchical filtering of input samples using an ensemble of b-XGBoost, Siamese Neural Network (Siamese-NN), and DNN. These attackers were transmitted to I-SiamIDS (second layer), which uses a multi-class eXtreme Gradient Boosting classifier for categorizing attacks into distinct types. For both the NSL-KDD and CIDDS-001 datasets, I-SiamIDS outperformed its competitors in Accuracy, Recall, Precision, F1-score, and values of Area AUC.

In 2021, Lee *et al.* [7] have solved the data imbalance problem by the GAN model. It has also suggested an RF-based model to identify the classification accuracy after correcting data imbalances with GAN. The experiment has revealed the performance of the suggested without taking into account the data imbalance.

In 2020, Habib *et al.* [8] executed a method for IoT network intrusion detection Furthermore, this work introduced an efficient IDS method that solved the problem of feature selection. The updated approach was the MOPSO-Lévy, and it was evaluated using real IoT network data from the UCI repository. When compared to the state-of-the-art evolutionary multi-objective algorithms, MOPSO-Lévy has shown better performance in intrusion detection.

In 2022, Jung *et al.* [44] used an altered neural network and a hybrid synthetic minority oversampling methodology as part of a mixed resampling approach. To create a dataset that was balanced, this raised the minority population. After that, they optimized the model's performance using the freshly created dataset using a bagging ensemble approach. It was tested against two public intrusion detection datasets, such as PKDD 2007 (balanced) and CSIC 2012 (unbalanced), and showed improved results over existing methods.

In 2022, Lin *et al.* [45] introduced an ML framework to identify security attacks in difficult network environments. For data pre-processing from various sources (network packets, system/statistical logs), a variational auto encoder, multilayer perceptron models, and an effective range-based sequential search algorithm were combined. This method showed promise when tested on open datasets like HDFS. It was also discovered that handling imbalanced datasets could enhance F1 score and recall rate performance.

In 2022, Bo Cao *et al.* [41] executed a repeated edited nearest neighbors (RENN) and hybrid sampling algorithm combining adaptive synthetic sampling (ADASYN) to solve the inequity in the dataset. Initially, the RF algorithm was used for selection, and Pearson correlation analysis was used to reduce feature redundancy. CNN was used to extract spatial features, which were then processed using attention mechanisms and average pooling and max pooling techniques to give each feature a different weight. This reduced overhead while also enhancing performance. The long-distance-dependent information was gathered through the (GRU) through Softmax. Finally, this system was tested with various datasets to improve the accuracy rate compared to other CNN-GRUs while dealing with class imbalance problems.

In 2022, Azizjon Meliboev *et al.* [42] experimented with DL for instance, GRU, CNN, LSTM, and RNN through consecutive information in an arranged time period as a mean traffic report for the development of the IDS. Then they were tested on three different benchmark datasets with different architecture and parameters using learning rates in both balanced and imbalanced training scenarios, and a supervised-learning method label was provided. Thus, results showed that a single CNN model combined with LSTM outperformed other approaches because it could better identify patterns in network traffic record data.

In 2022, Fu *et al.* [43] have executed a DLNID for traffic anomaly detection. A Bi-LSTM network joined with DLNID obtained a sequence feature through a CNN network. ADASYN was used to avoid an imbalance in sample expansion of minority class samples to form a relatively symmetric dataset. To improve information fusion, data dimensionality was also reduced using a modified stacked auto encoder. The manual feature extraction technique is not necessary for DLNID. Experimental results on the publicly available benchmark dataset for network intrusion detection (NSL-KDD) demonstrate that this model's accuracy and F1 score were superior to those of other comparable techniques. For publicly available benchmark datasets, this strategy has been successfully tested.

2.2 Problem Statement

In the current digital era, network intrusion categorization has become a large and essential challenge in the ICT industry due to the unbalanced big data environment. IDSs are a popular tool for detecting and preventing internal and external network attacks/intrusions at the moment. Misuse-based IDSs generally are split into "Host-based and network-based systems", and employ pattern-matching algorithms to identify intrusions. Over the last few decades, "ML and DM" techniques have been frequently used in IDS for identifying intrusions. Improving intrusion categorization accuracy while lowering the false-positive rate is one of the key problems for creating IDS using ML and data mining techniques. R2L and U2R cyber attacks are some of the datasets' minority classes, and DL-based IDSs seem unable to identify them adequately owing to a paucity of sample points in these classes. This creates an issue of class imbalance and raises the risk of the network being hacked through unnoticed breaches. To resist new and complex attacks, modern IDSs have been created using DL techniques and trained on intrusion detection datasets such as KDD and NSL-KDD. Besides the Normal class, the Denial of Service and Probe attack classes in these two datasets involve a large number of cases, while the R2L and U2R assault classes have a small sample size. Siam-IDS, a new IDS based on Siamese Neural Networks was introduced, to address the problem of class imbalance in IDSs (Siamese-NN). Despite utilizing standard class balancings approaches like oversampling and random undersampling, the suggested Siam-IDS effectively identifies R2L and U2R intrusions. Siam-IDS, a new IDS based on Siamese Neural Networks was introduced, to address the problem of class imbalance in IDSs (Siamese-NN). Despite utilizing standard class balancings approaches like oversampling and random under sampling, the suggested Siam-IDS effectively identifies R2L and U2R intrusions. On the benchmark

Cybersecurity datasets KDD99, UNSW-NB15, UNSW-NB17, and UNSW-NB18, the resampling methods random under sampling, random oversampling, random under sampling, and random oversampling, random under sampling with Synthetic Minority Oversampling Technique, and random under sampling with Adaptive Synthetic Sampling Method were used. The findings were assessed using "Macro precision, Macro recall, and Macro F1-score". The following trends were discovered: first, oversampling increases training time while under sampling lowers it; second, if the data is very unbalanced, both oversampling and under sampling greatly enhance recall. third, resampling will not have much of an influence if indeed the information is not highly unbalanced; fourth, with resampling, primarily oversampling, more of the minority data (attacks) was discovered.

3. Proposed IDS in Imbalanced Dataset

3.1 Architectural Description

The following four key phases are used to establish an intrusion detection model for class imbalance data: (a) Pre-processing, (b) Imbalance processing, (c) Feature extraction, and (d) Intrusion detection phase. Fig. 1 depicts the general architecture of the planned project. The obtained data D^{in} on class imbalance is first pre-processed using a data cleaning and standardization approach. Then, the pre-processed class imbalance data D^{Norm} is more balanced in the imbalanced processing phase by an improved over-sampling technique. The imbalanced processing phase uses a new enhanced over-sampling approach using SMOTE and Multi-kernel FCM clustering model to balance the pre-processed class imbalance data. Subsequently, multi-features like EMA, DEMA, IWH, and statistical features (Mean, Median, Standard Deviation, Percentile, Momentum, Min-max, Skewness, and Kurtosis) are extracted from these balanced data. The extracted overall features F are given for K-fold validation. The data feature acquired from the k-fold model denoted as F^* is used to train the classifier in the intrusion detection phase. Then, the presence of an intruder is detected by a combined ML model, which is the combination of four individual ML models such as SVM (Out_{SVM}), RF (Out_{RF}), optimized DBN (Out_{DBN}) with incremental learning, and NB (Out_{NB}) to achieve the utmost detection performance. Moreover, to achieve the utmost detection accuracy as well as better tradeoff performance, we've fine-tuned the weight of DBN via a new SI-SOA, which is an improved version of standard SOA. Then, the logoff performance is computed from the outcome acquired from each of the individual classifiers (SVM, DBN, NB, and RF). Finally, the log computed result will portray the detected attacks in the network.

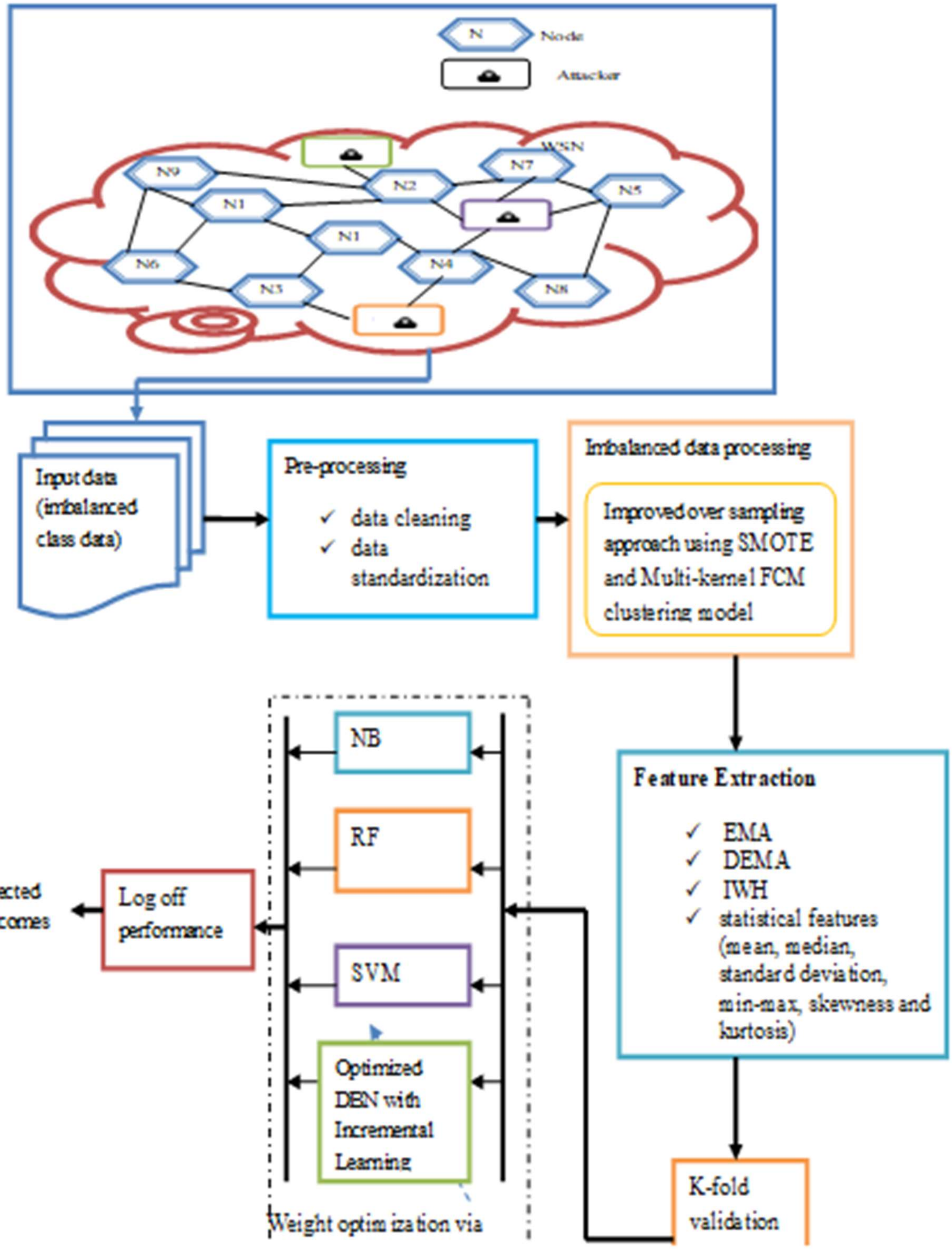


Fig 1. *Architecture of the proposed work*

4. Pre-processing via Data Cleaning and Data Standardization

Preprocessing the data is an important mining approach that transforms raw data into a usable and accessible structure. Data preprocessing is a collection of approaches being used applying a data mining approach, and this is regarded to be of greatest concern inside the well-known Knowledge Discovery from Data process. In this work, the collected raw data is pre-processed via data cleaning and data standardization approach.

4.1 Data Cleaning

The data cleaning/cleansing is used to remove the redundant and meaningless data available in D^{in} [1]. The data acquired at the end of the data cleaning phase is denoted as D^{clean} , which is subjected to the data standardization phase.

4.2 Data Standardization

It is done to scale the data values in a specified range. In this research work, the cleaned data is standardized D^{Norm} using Eq. (1).

$$D^{Norm} = \frac{D^{clean} - \mu}{\delta} \tag{1}$$

Here, μ and δ denotes the mean and standard deviation. The data is normalized to a Gaussian distribution with μ =0 and δ =1. The data normalization aids in improving the convergence speed and the accuracy of the model. The acquired pre-processed data is denoted as D^{Norm}. The final data set generated by a successful data preparation stage is considered a credible and acceptable source for any data mining method performed later. Then, the pre-processed class imbalance data D^{Norm} is more balanced in the imbalanced processing phase by an improved over-sampling technique.

5. Imbalance Processing: an improved over-sampling technique with SMOTE and Multi-kernel FCM clustering model

Inherently, the number of aberrant samples D^{Norm} is low. However, relying only on over-sampling might result in an excessive amount of duplicate data, as well as increased time and space expenditures. To address this, a new enhanced oversampling approach based on SMOTE is developed, with a Multi-kernel FCM clustering model. The current research work oversamples classes that are below $D^{resample}$ to $D^{balanced}$ using SMOTE.

5.1 SMOTE

SMOTE [1] is a traditional oversampling approach that "Synthesizes" minority class samples to expand the amount of minority class samples. To avoid over fitting throughout the phase of constructing a classification model, the "synthesis" is used to generate a sample that does not exist in the original dataset, rather than simply copying samples like ROS. SMOTE creates a new synthetic sample by randomly selecting a location between the minority class sample and its k-nearest neighbors. We utilise a Multi-kernel FCM clustering model to under-sample the majority of class samples to $D^{resample}$ for classes with more samples than $D^{resample}$.

5.2 Multi-Kernel FCM Clustering

One of the most promising fuzzy clustering algorithms is FCM. It's much more versatile than that of the analogous hard clustering methods in most instances. The kernel has been added to the FCM algorithm, which leads to improved performance. The MKFC technique chooses the appropriate membership values in the very same timeframe. Effective kernels or attributes are more attributable to clustering and thereby enhance the outcomes. Ultimately, MKFC produces fuzzy (soft) prediction performance, which seems to be best suited to situations wherein clusters overlap significantly. In this research work, 3 major kernels are used: RBF, sigmoid, and cosine.

1. **Cosine kernel:** The L2-normalized dot product of vectors is computed using cosine similarity. If x, y are row vectors, the cosine similarity $\ker nel_{\cos}$ between them is defined in Eq. (2).

$$\ker nel_{\cos}(x, y) = \frac{xy^T}{\|x\|\|y\|} \tag{2}$$

Because Euclidean (L2) normalization projects the coordinates onto the unit sphere, their dot product equals the cosine of the angle between the locations represented by the vectors. This is known as cosine similarity.

2. **Sigmoid kernel:** The sigmoid kernel between two vectors is computed using the function sigmoid kernel. The hyperbolic tangent, or Multilayer Perceptron, is another name for the sigmoid kernel $\ker nel_{sig}$ (because, in the neural network field, it is often used as a neuron activation function). It is defined as in Eq. (3)

$$\operatorname{kernel}_{sig}(x, y) = \tanh(\gamma x^T y + c0) \quad (3)$$

where: x, y are the input vectors; γ refers to the slope, $c0$ stands for intercept.

3. **RBF kernel:** The RBF kernel $\operatorname{kernel}_{RBF}$ between two vectors typically is computed by the function RBF kernel. This kernel's definition is as in Eq. (4).

$$\operatorname{kernel}_{RBF}(x, y) = \exp(-\gamma \|x - y\|^2) \quad (4)$$

The input vectors x, y are used. The Gaussian kernel with variance σ^2 is known as $\gamma = \sigma - 2$.

From the balanced dataset, the multi-features are extracted. This balanced dataset is denoted as $D^{balanced}$.

Algorithm 1: **Pseudo-code of new improved over-sampling technique with SMOTE and Multi-kernel FCM clustering model**

Input: $D_i^{Norm} = \{D_1^{Norm}, D_2^{Norm}, ..., D_M^{Norm}\}$

here, M denotes the overall count of classes

The overall count of samples is pointed as $\left|D_i^{Norm}\right| = N$

Output: Balanced dataset $D^{balanced}$

Compute $D^{resample} = \operatorname{int}\left(\frac{N}{M}\right)$

for $i \leftarrow 1$ to M do

  if $\left|D_i^{Norm}\right| < D^{resample}$ then

    $D_i^{balanced} = SMOTE(\ D_i^{Norm}, D^{resample})$# $D_i^{balanced}$ is oversampled using SMOTE, so $\left|D_i^{Norm}\right| = D_i^{balanced}$

  endif

  if $\left|D_i^{Norm}\right| < D^{resample}$ then

    $G_k = MK - FCM\ (D_i^{Norm}, M)$# to cluster D_i^{Norm} into M clusters, the Multi-kernel FCM $MK - FCM$ clustering model is used

    $k = 1,2,....,M$

    For $k \leftarrow 1$ to M do

      $G_k^i = \operatorname{Re}sample\left(G_k, \frac{D^{resample}}{M}\right)$

      #select $\frac{D^{resample}}{M}$ randomly from G_k

    end for

    $D_i^{balanced}$ =Concatenate$\left(G_k^i\right)$

  endif

  $D^{balanced}$ = Concatenate$\left(D_i^{balanced}\right)$

end for

return $D^{balanced}$

---

6. Multi-Feature Extraction

The multi-features like EMA, DEMA, IWH, and statistical features (Mean, Median, Standard Deviation, Percentile, Standardized moment) are extracted from these balanced data $D^{balanced}$.

6.1 Statistical Features

The statistical features like Mean, Median, Min and Max, Skewness, and Variance are extracted from $D^{balanced}$.

Mean: Within the time window, μ is the average of R values of $D_i^{balanced} = \left\{D_1^{balanced}, D_2^{balanced}, ..., D_R^{balanced}\right\}$. This is mathematically given in Eq. (5).

$$\mu = \frac{1}{R}\sum_{i=1}^{R} D_i^{balanced} \tag{5}$$

Standard Deviation (σ): It is computed "To measure how the data values $D_i^{balanced} = \{D_1^{balanced}, D_2^{balanced}, ..., D_R^{balanced}\}$ are spread out".

Median: When a data set is ordered, the value that falls in the center of the data. "The median is the middle data entry in a data collection with an odd number of items. If there are an even number of items in the data, the median is calculated by adding the two values in the center and dividing the result by two."

Min and Max: "The maxima and minima (The plurals of maximum and minimum) of a function, known collectively as Extrema (Plural of extremum), are the largest and smallest value of the function, either within a given range (The local or relative extrema) or on the entire domain of a function (the global or absolute extrema)", according to mathematical analysis.

Skewness [36] $f^{skewness}$: "It's a symmetry measure, or more precisely, the lack of symmetry. Only if the left and right sides of the center point are comparable is a dataset or distribution symmetric."The mathematical expression of skewness SF_1 is given in Eq. (6).

$$f^{skewness} = \frac{\sum_{i=1}^{M}\left(D_i^{balanced} - \mu\right)^3 \Big/ R}{\sigma^3} \tag{6}$$

For any symmetric data, the skewness value is near 0, and for the normal distribution, it is zero.

Kurtosis [37] $f^{kurtsosis}$: "It's a statistic that determines whether data is light-tailed or heavy-tailed, and it's connected to the normal distribution". Mathematical formula of kurtosis $f^{kurtsosis}$ for univariate data such as $D_i^{balanced} = \{D_1^{balanced}, D_2^{balanced}, ..., D_R^{balanced}\}$ is expressed in Eq. (7).

$$f^{kurtsosis} = \frac{\sum_{i=1}^{M}\left(D_i^{balanced} - \overline{D_i^{balanced}}\right)^4 \Big/ R}{\sigma^4} \tag{7}$$

Standardized Moment: In probability theory and statistics, a standardized moment of a probability distribution is a normalized moment (usually a higher degree central moment). The normalization appears to be a standard deviation term, which renders the moment scale invariant. The standardized moment is calculated mathematically using Eq. (8)

$$\begin{aligned}\mu_m &= E\left[\left(D_i^{balanced} - \mu\right)^m\right] = \\ &= \int_{-\infty}^{+\infty}\left(D_i^{balanced} - \mu\right)^m .B(D_i^{balanced})\end{aligned} \tag{8}$$

Here, probability distribution and expected value $D_i^{balanced}$ are pointed as B and E respectively. For the degree m, the standardized moment is denoted as $\frac{\mu_m}{\sigma_m}$, which is said to be the m^{th} moment of the mean. In addition, σ_m denotes the m^{th} power of standard deviation.

Percentile: It provides the idea of "How the data values $D_i^{balanced}$ are spread over the interval from the smallest value to the largest value". The extracted statistical features are denoted as $F^{statistical}$.

A. EMA

An EMA [38] is indeed a kind of MA that gives the much more recent data points more weight and importance. The exponentially weighted MA is yet another name for the exponential MA. The extracted EMA features are denoted as f^{EMA}. This is mathematically given in Eq. (9).

$$F^{EMA} = \frac{1}{R}\sum_{i=1}^{R} D_i^{balanced} \tag{9}$$

The derived EMA features are indicated by F^{EMA}.

B. Propsoed weighted Holoentropy features

"The holo-entropy $HL_\psi(\upsilon)$ is defined as the sum of the entropy and the total correlation of the random vector υ, and can be expressed by the sum of the entropies on all attributes. It is modeled as in Eq. (10). The weighted holo-entropy $We_\psi(\upsilon)$ is the sum of the weighted entropy on each attribute of the random vector υ" and it is modeled as in Eq. (11).

$$HL_\psi(\upsilon) = H_\psi(\upsilon) + \Xi_\psi(\upsilon) = \sum_{i=1}^{R} H_\psi\left(D_i^{balanced}\right) \tag{10}$$

$$We_{\psi}(\upsilon) = \sum_{i=1}^{R} w_{\psi}\left(D_i^{balanced}\right) H_{\psi}\left(D_i^{balanced}\right) \tag{11}$$

The weight function $w_{\psi}\left(D_i^{balanced}\right)$ is computed using the newly developed mathematical expression, which is based on the tanh activation function given in Eq. (12).

$$w_{\psi}\left(D_i^{balanced}\right) = \left(\frac{2}{1+\exp^{-2\left(-H_{\psi}\left(D_i^{balanced}\right)\right)}}\right) - 1 \tag{12}$$

The extracted proposed weighted holoentropy feature is denoted as F^{IWH}

C. DEMA

The DEMA is a mixture of smoothed EMA and a standard EMA. The extracted DEMA feature is pointed as F^{DEMA}. The extracted overall feature is denoted as $F = F^{IWH} + F^{statistical} + F^{EMA} + F^{DEMA}$, which is given for K-fold validation.

7. K-Fold Validation

Cross-validation is indeed a re-sampling method for evaluating ML algorithms on a small set of data. The technique includes only one parameter, k, which specifies how many groups a given data sample should be divided into. As a result, k-fold cross-validation is a popular name for the process.
The general procedure is as follows:

1. Randomly shuffle the dataset feature F.
2. Divide F into a total of k groups.

For each one-of-a-kind group:

a) Use the group as a test data set or a holdover.
b) As a training data set, use the remaining groupings.
c) Create a model and test it on the training set.
d) Keep the average rating but discard the model out.
e) Using a sampling of model assessment ratings, summarise the model's ability.

The data feature acquired from the K-fold model denoted as F^* is used to train the classifier in the intrusion detection phase. In the result and discussion phase, this K value is varied, and the performance of the projected model is recorded.

8. Intrusion Detection Phase

The intrusion detection phase is modeled with four individual ML models like SVM, Optimized DBN with incremental learning, NB, and RF to achieve the utmost detection performance. Each of the classifiers is a trained acquired data feature F^*. The final result is the logoff of outcomes from each of the classifiers.

8.1 RF based Classification

RF is indeed a supervised classification approach that is based on the ML model's DT [36]. The RF, in general, is the combination of "unbiased, unconnected, and de-correlated DT," thus the name "RF". The decision tree technique is used to create the tree-like model, and each tree is created by randomly picking nodes. The significant concepts used to determine the results are entropy and Gini values. The Gini coefficient is used to calculate impurity, and the Entropy is used to calculate the information gain of nodes. The node with the least Gini impurity is selected for further decomposition during the computation of the Gini impurity. Moreover, from the Gini value, the impurity can be calculated by subtracting the square of the probability $prob$ of each class from 1. Furthermore, by subtracting the square of each class's probability from 1, the impurity can be calculated from the Gini value. Eq. (13) is used to calculate it mathematically. It is said to be a perfect split when $Gini$ =0. The time is indicated as t.

$$Gini = 1 - \sum_{t=0}^{t=1} prob.t^2 \tag{13}$$

Entropy is also used to divide the node with the most information gain. When $Entropy\ gain = 1$ at the leaf node, it is considered to be a full information-acquired node and hence a perfect split. The entropy $(Entropy)$ is computed using the formula Eq. (14).

$$Entropy = \sum_{i=1}^{c} -prob.i.\log_2 .prob.i \tag{14}$$

The information gain is determined using the computed entropy value and the difference between the computed entropy and the other classes. As a result, RF is regarded as one of the finest supervised classification algorithms, providing improved classification while avoiding overfitting. The outcome from RF is denoted as Out_{RF}.

8.2 Navie Bayers

It's a **categorization** method that uses Bayes' Hypothesis and assumes predictor independence. An NB classifier assumes that one feature in a class is independent of the other attribute. The NB model is very simple to construct and is especially effective when dealing with big datasets. NB is renowned for surpassing only the most advanced classification algorithms due to its simplicity. The Bayes theorem allows us to calculate posterior probability $P(C|F^*)$ from $P(C)$, $P(F^*)$. Consider the following equation in Eq. (15)

$$P(C|F^*) = \frac{P(C|F^*).P(C)}{P(F^*)} \tag{15}$$

Here, $P(C|F^*)$ is the likelihood and P denotes the posterior probability. In addition, $P(C)$ and $P(F^*)$ denotes the class prior probability and predictor prior probability. The outcome from Naive Bayes is denoted as Out_{NB}.

8.3 SVM

SVM [39] is well-known for its straightforward non-linear regression procedure, which also makes it easy to solve the optimation problem:

$$\begin{gathered} -L(\lambda) = -\sum_{d=1}^{Q} \chi_d + \\ 1/2.\sum_{d=1}^{Q}\sum_{\kappa=1}^{Q} \chi_d \cdot \chi_\kappa \cdot y_d \cdot y_\kappa . \lambda(z_d, z_\kappa) \to \min_{\chi}, \\ \sum_{d=1}^{Q} \chi_d \cdot y_d = 0, 0 \le \chi \le Cr, d = \overline{1}, \overline{Q} \end{gathered} \tag{16}$$

Here, z_d and $\lambda(z_d, z_\kappa)$ denotes the elements in training data features F^* and kernel function respectively. In addition, χ_d and y_d represents the dual variable and the number that is +1 or -1. Further, Cr is the regularization parameter, Q denotes the count of objects that are said to be available F^*. To enhance the precision of the SVM order, it's important to narrow down the region of the improperly grouped items. The isolating hyperplane is where the majority of the improperly placed items are found. As a result, using the additional tools to improve the order quality of the items within the isolating strip is critical. The outcome from SVM is denoted as Out_{SVM}.

8.4 Optimized DBN with Incremental Learning

Smolensky introduced DBN [40] in 1986, and it consists of three layers: "Input, Output, and Hidden". The visible neurons are in the input layer, whereas the hidden neurons are in the output layer. There is an association between hidden neurons and input neurons. There is a unique symmetrical link between the visible and hidden neurons. The stochastic neuron model may be used to extract the relevant output from DBN for each input. In addition, the Boltzman network is used in DBN to achieve probabilistic results. The DBN result is represented by the notation r in binary form. The probability $(J_p(\delta))$ in the sinusoidal function is expressed in Eqs. (17) and (18). When $S > 0$, the pseudo temperature parameter S reduces the probability noise level. Furthermore, Eq. (19) shows the stochastic probability model in a deterministic form.

$$r = \begin{Bmatrix} 1 & with\ J_p(\delta) \\ 0 & with\ 1 - J_p(\delta) \end{Bmatrix} \tag{17}$$

$$J_p(\delta) = \frac{1}{1 + e^{\frac{-\delta}{S}}} \tag{18}$$

$$\lim_{S \to 0^+} J_p(\delta) = \lim_{S \to 0^+} \frac{1}{1 + e^{\frac{-\delta}{S}}} = \begin{cases} 0 & for \quad \delta < 0 \\ \frac{1}{2} & for \quad \delta = 0 \\ 1 & for \quad \delta > 0 \end{cases} \tag{19}$$

The energy factor is important for designing the neuron states h of the Boltzmann machine, and the mathematical formula for the energy of the Boltzmann machine is Eq. (20). Furthermore, in DBN, the weight between the neuron and the biases of the neurons is represented by $M_{i,j}$ and α respectively.

The joint configuration between the visible q as well as hidden neurons d in terms of energy is indicated in Eq. (21), Eq. (22), Eq. (23), and Eq. (24). The binary states of the visible state and the hidden state i and j are represented as h_i and h_j respectively.

$$Eg(h) = -\sum_{i<j} L_{i,j} h_i h_j - \sum_{i} \alpha_i . h_i \tag{20}$$

$$\Delta Eg(h_i) = \sum_{j} h_j L_{i,j} + \alpha_i \tag{21}$$

$$Eg\left(\vec{q},\vec{d}\right) = -\sum_{(i,j)} L_{i,j} q_i d_j - \sum_{i} q_i u_i - \sum_{j} d_i g_j \tag{22}$$

$$\Delta Eg\left(q_i,\vec{d}\right) = \sum_{j} L_{ij} d_j + u_i \tag{23}$$

$$\Delta Eg\left(\vec{q},d_j\right) = \sum_{j} L_{ij} q_j + g_j \tag{24}$$

The weight parameters derived from the encoded probability distribution of the input data are used to determine the RBM learning pattern. Eq. (25) determines RBM's weighting assignment, and the assigned probability can be maximized by RBM. q stands for the input visible vector. RBM also can assign probability to each unique visible and hidden vector, as illustrated by the energy function in Eq. (26). The allocated weight is denoted by the notation G_e, while the training set is denoted by the symbol O. The partition function (V) is then derived by summing the energy of all possible states of the neurons, as shown in Eq. (27).

$$G_e = \max_{\hat{G}} \prod_{q \in O} P\left(\vec{q}\right) \tag{25}$$

$$P(\vec{q},\vec{d}) = \frac{1}{V} e^{-Eg(\vec{q},\vec{d})} \tag{26}$$

$$V = \sum_{\vec{q},\vec{d}} e^{-Eg\left(\vec{q},\vec{d}\right)} \tag{27}$$

Furthermore, the normal Boltzmann Machine does not rely on the visible or hidden neurons during the assessment of energy between the visible or hidden neurons, but the RBM does. RBM is good at data reconstruction but not so good at data categorization; therefore unsupervised learning is used to train RBM. Because RBM takes a long time to converge with the required model, a CD is used. The hidden neurons are used as input in the training patterns MLP layer, which is identical to the RBM layer.

The error function "MSE" is computed in DBN to know about the error interrupted within it during the training process. Mathematically, the MSE is given in Eq. (28).

$$MSE = Act - \Pr e \tag{28}$$

Here, Act points to the actual output (i.e. actual outcome) and $\Pr e$ denotes the predicted output (i.e. predicted outcome acquired with proposed work). This error ought to be lower, to prove that the projected prediction model is more accurate in detecting the attacks during the testing phase. We've attempted to lessen this error (MSE) by using the new SI-SOA model. The objective function or fitness function of this work is shown in Eq. (29).

$$Obj - \min(MSE) \tag{29}$$

To achieve this objective, the weight L of DBN is fine-tuned using the SI-SOA model. The solution fed as input to SI-SMO is L. Here, p denotes the overall weight of DBN. The solution encoding model is shown in Fig.2.

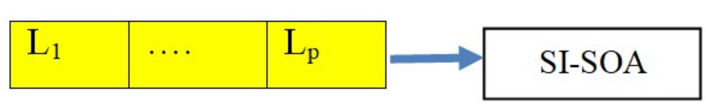


Fig.2. *Solution Encoding*

Below is a list of the stages involved in the CD algorithm.

1. The binary input is assumed by picking the training samples q and attaching them to the visible neurons.
2. The probability of the hidden neurons U_d is calculated by multiplying the visible vector q with the weight matrix G, which is denoted by $\left(U_d = \psi\left(G.q\right)\right)$ in the Eq. (30).

$$p\left(d_j \to 1 \mid \vec{q}\right) = \psi\left(g_j + \sum_i q_i L_{i,j}\right) \tag{30}$$

3. The hidden states d are sampled using probability U_d.
4. The positive gradient ζ^+ is calculated as $\zeta^+ = q.U_d^T$ as the outer product of q and U_d.
5. According to Eq. (31), the visible state q' is reconstructed from the hidden state $\hat{d}$. Furthermore, the visible state q' is reconstructed by resampling the concealed states $\hat{d}$.

$$p\left(q_i \to 1 \mid \vec{d}\right) = \psi\left(u_i + \sum_j d_j L_{i,j}\right) \tag{31}$$

6. The exterior product of vectors q' and $\hat{d}$ is the negative gradient ζ^-. $\zeta^- = q'.\hat{d}^T$ is the formula for calculating the negative gradient.
7. The weight updates are obtained by subtracting the negative gradient ζ^+ from the positive gradient ζ^+. This weight update is carried out by Eq. (32)

$$\Delta G = \chi\left(\zeta^+ - \zeta^-\right) \tag{32}$$

8. The updating of weights occurs using the newly acquired values, as described in Eq. (33)

$$\Delta L_{i,j}^* = \Delta L_{i,j} + L_{i,j} \tag{33}$$

The outcome from DBN is denoted as Out_{DBN} .

Incremental Learning: In a nutshell, incremental learning is indeed a continual learning process in which information classes of labeled data are progressively made accessible in batches. The phrase "Incremental Learning" is also used in the literature about incremental network pruning and growth, as well as online learning. In addition, terms including lifelong learning, constructive learning, and evolutionary learning have been utilized to determine the acquisition of new information. To imitate genuine, biological brains, a pure incremental learning model must be developed. Animals and humans can remember new events despite forgetting previous ones because of the supremacy of the biological brain. In artificial neural networks, however, precise sequential learning doesn't function completely. The adoption of a fixed architecture and/or a training procedure focused on minimizing an objective function, which results in catastrophic interference. It's because the objective function's minima for one number of instances may vary from the minima of succeeding sets of instances. As a result, each new training dataset induces the network to forget about the prior ones partially or completely. The stability-plasticity conundrum [15] is the name given to this issue. To address these issues, an incremental learning algorithm is used that satisfies the following requirements:

i. It must be able to expand the network and accommodate new tasks (classes) as they are introduced through new examples.
ii. There should be minimal overhead when training for new tasks (classes).
iii. It shouldn't be necessary to have accessibility to the previously observed instances that were used to train the preexisting classifiers.
iv. It must maintain prior knowledge, avoiding catastrophic forgetting. For illustration: Consider that perhaps a base network is trained with 4 task 0 classes (C1–C4), and after training, all training data for all those four classes is deleted. The network must next handle sample data for task 1 with two classes (C5, C6) while maintaining knowledge of the first four classes.

As a result, network capacity should be upgraded, as well as the network must be efficiently retrained using only the new data of task 1 (of C5 and C6), such that the upgraded structure could categorize both task classes (C1 and C6). This is a task-specific categorization if the tasks are categorized individually. When they are categorized jointly, however, it is referred to as combined classification. We'll concentrate on the task-specific situation while considering the combined categorization. The incremental learning approach is depicted in Fig. 3.

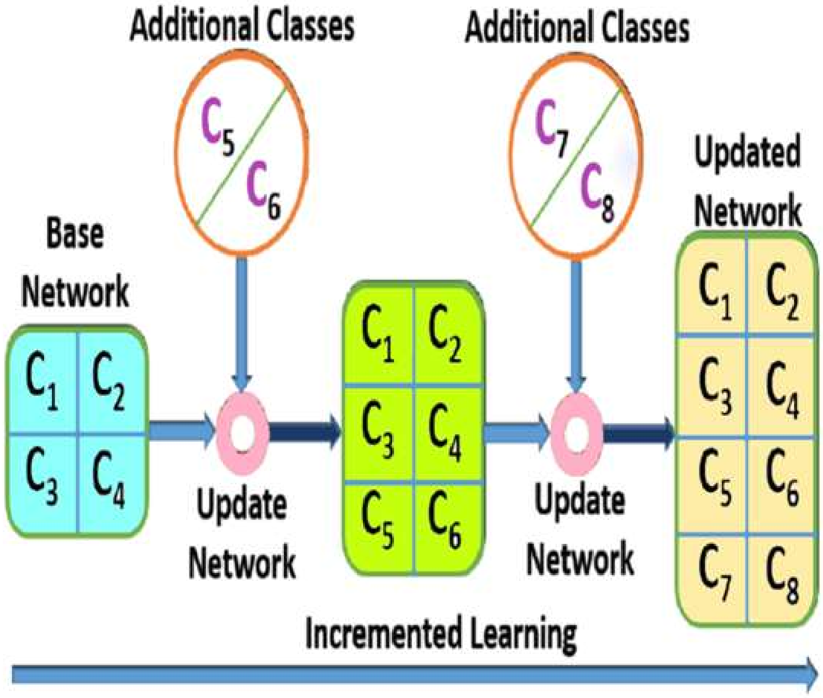


Fig.3. *Incremental learning: An illustration*

SI- SOA: The SOA [29] is based on the seagulls' assaulting behavior. In general, the SOA excels in solving complicated optimization problems with a faster rate of convergence. Furthermore, it is claimed in the literature that self-adaptivity in the conventional optimization model improves convergence and prevents solutions from becoming stuck in local optima [32] [33] [34] [35]. The proposed SI-SOA model is illustrated below:

Step 1: Step 1: Set up the search agent's motion behaviors A, B, as well as the current iteration itr as well as the maximum iteration Max^{itr}. Furthermore, the populace pop of the search agent is set up.

Step 2: Set the constants for the spiral form definition to 1 and the variable for managing the frequency of using variables to f_c =2.

Step 3: While $itr < Max^{itr}$ do

a) Compute the search agent's fitness Fit using Eq. (29).

b) Proceed to the SOA migration step. During migration, the algorithm replicates how a flock of seagulls migrates from one area to another. At this stage, a seagull must satisfy three requirements: Keeping crashes at bay: To avoid collisions between neighbors, a new search agent position is created using an optimum variable A (i.e., other seagulls).

c) c) Rather than utilizing the standard SOA model, we have developed a novel mathematical formula to determine the optimal variable A. The point of the search agents where no collision occurs is mathematically described as Eq. (34). Eq. (35) shows the newly developed equation for computing the search agent's movement behavior A.

$$C = A \times \overrightarrow{H(itr)} \tag{34}$$

$$A = A^0 + \int_1^{itr} V.f(N;\lambda,t)ditr \tag{35}$$

Wherein,

$$V.f(N;\lambda,t)ditr = \frac{itr}{\lambda}\left(\frac{N}{\lambda}\right)^{itr-1}.e^{\left(\frac{N}{\lambda}\right)^{itr}} \tag{36}$$

Here, N, λ are the parameters of the Weibull probability distribution and S is the span factor.

d) Movement in the direction of the best neighbor: In this paper, a new mathematical model for movement in the direction of the best neighbor is created by adding a new levy flight function $Levy(\beta)$, a random walk displayed by the search agents while seeking the optimum location to update the solutions.

e) e) The performance enrichment of the projected model is also aided by this factor. After avoiding accidents with their neighbors, the search agents head in the direction of the best neighbor. $Rand1$ is a random number that is initialized. This process is mathematically described as Eq. (37).

$$\overrightarrow{M} = C + B \times Levy(\beta) * \left(\overrightarrow{Fit(itr)} - \overrightarrow{H(itr)}\right) \tag{37}$$

Here, $\overrightarrow{M}$ denotes the positions of the search agent towards the best fitness $\overrightarrow{Fit(itr)}$, and C is the location of the search agent that doesn't collide with the other search agents. Mathematically, B is modeled as per Eq. (38).

$$B = 2 \times A^2 \times Rand1 \tag{38}$$

a) Compute the search agent's fitness Fit using Eq. (29).
b) Finally, the search agent can modify its location to the optimal search agent by maintaining tight contact with them. Eq. (39) is used to calculate the distance between $\overrightarrow{Fit(itr)}$ and the search agent.

$$D = \left(\vec{C} + \overrightarrow{M}\right) \tag{39}$$

b. After that, go to the "attacking phase" to update the search agents' positions.
c. Step 4: End while
d. Step 5: Return Fit
e. Step 6: End

Finally, the log-off performance is computed onto the outcomes acquired from each of the classifiers SVM (Out_{SVM}), RF (Out_{RF}), optimized DBN (Out_{DBN}) with incremental learning, and NB (Out_{NB})

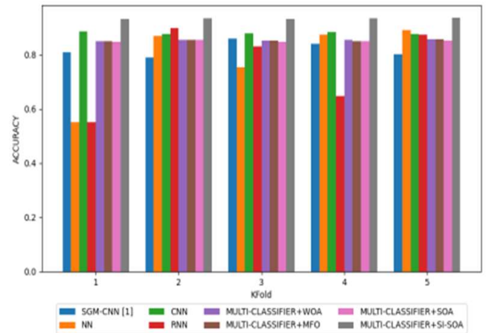
9. Result and Discussion

9.1 Simulation procedure

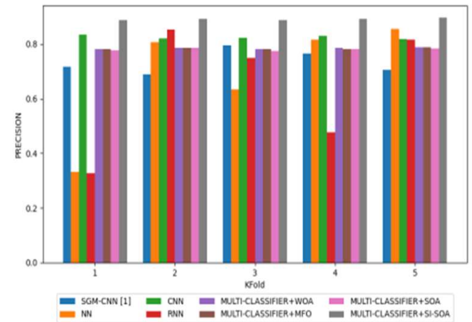
The proposed intrusion detection model with class imbalance data was implemented in PYTHON and the outcomes were verified. Dataset 1 and Dataset 2 are gathered from:[30] and [31], respectively. The performance of the proposed work (MULTI-CLASSIFIER+SI-SOA) is compared over the existing models like SGM-CNN [1], NN, CNN, RNN, MULTI-CLASSIFIER+WOA, MULTI-CLASSIFIER+MFO, MULTI-CLASSIFIER+SOA respectively. In this research work, we have fixed the K-value of K -fold evaluation as 5(K =5), and we've varied this K value from 1, 2, 3, 4, and 5, to evaluate the performance of the projected model. Furthermore, the performance was based on the different performance measures including "Accuracy, Sensitivity, Specificity, Precision, F-measure, FDR, FNR, FPR, NPV, and MCC" respectively.

9.2 Overall Performance Analysis

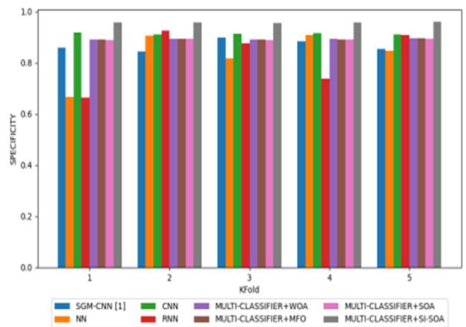
The performance analysis of the proposed scheme is computed to the existing schemes like SGM-CNN[1], NN, CNN, RNN, MULTI-CLASSIFIER+WOA, MULTI-CLASSIFIER+MFO, MULTI-CLASSIFIER+SOA respectively in terms of certain metrics like "positive measures (Accuracy, Sensitivity, Specificity, Precision); negative measures (FPR and FNR) and %, other measures (F-measure, NPV, and MCC)", and it is illustrated in Fig. 4, 5 and 6 respectively. On observing the outcomes, the projected model has attained the best performance (higher positive measures and lower negative measures) for every variation in the K -values; hence the projected model is said to be best for intrusion detection. All these improvements are owing to the extraction of the most relevant features (IWH features) along with the standard features to train the modeled ensemble classifier. Moreover, the count of the abnormal samples has been handled using the improved oversampling model, and this is also a major reason for enhancement in the performance of the projected model. The accuracy of the projected model is higher for every variation in the K -values. At K =5, the proposed work attained the highest accuracy value of 92%, which is 13%, 9.7%, 11.4%, 11.55, 11.6%, 11.6%, and 11.8% better than the accuracy value recorded by the existing models like SGM-CNN, NN, CNN, RNN, MULTI-CLASSIFIER+WOA, MULTI-CLASSIFIER+MFO, MULTI-CLASSIFIER+SOA, respectively. In addition, the specificity, sensitivity, and precision of detecting the attack by the proposed work are maximum when computed to the existing scheme for every variation in the K -value. At K =1, the precision of the projected model in identifying the intrusion in network is 26.3%, 63.15%, 13.6%, 63.15%, 231.05%, 21.05% and 23% better than the existing approaches like SGM-CN, NN, CNN, RNN, MULTI-CLASSIFIER+WOA, MULTI-CLASSIFIER+MFO, MULTI-CLASSIFIER+SOA respectively.



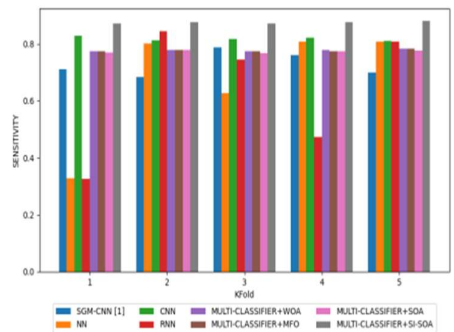
(a)



(b)

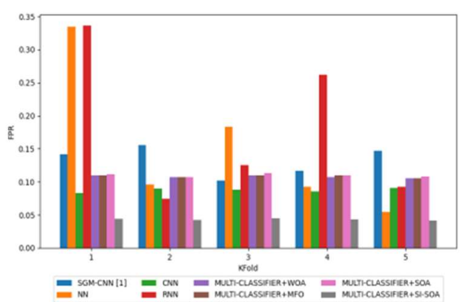


(c)

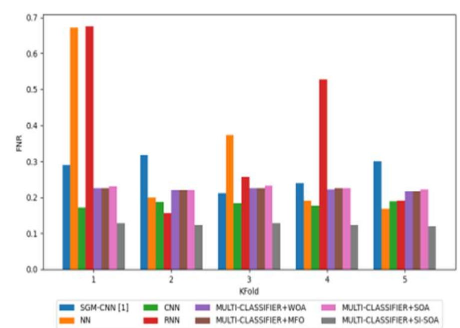


(d)

Fig.4. *Performance Analysis of Proposed and Traditional Optimization Model: (a)Accuracy, (b) Precision, (c) Specificity, (d) Sensitivity,*

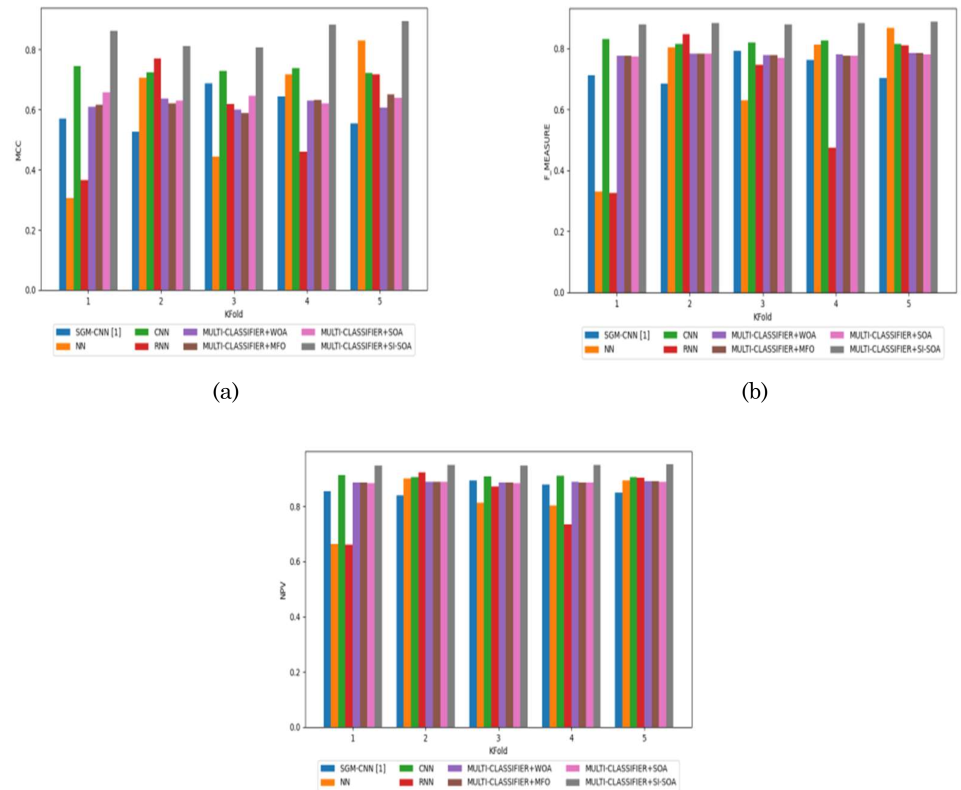


(a)



(b)

Fig.5. *Performance evaluation of the proposed and extant tactics in terms of (a) FPR and (b) FNR*



(a) (b) (c)

Fig.6. *Performance evaluation of the proposed and existing algorithms (a) MCC, (b) F-measure and (c) NPV*

On the other hand, negative metrics like FPR and FNR of the adopted model over other existing schemes for varying K -values are represented in Fig. 5. On observing the outcomes, the proposed work recorded the least error function in terms of FPR and FNR for every variation the k-value. As we've handled the unbalanced scenario using the newly proposed over-sampling procedure, we could acquire negligible errors. At K =1, the FNR of the proposed work is 66.6%, 85.7%, 50%, 85.7%, 60%, 60%, and 62.3% better than the existing approaches like SGM-CN, NN, CNN, RNN, MULTI-CLASSIFIER+WOA, MULTI-CLASSIFIER+MFO, MULTI-CLASSIFIER+SOA, respectively. Therefore, it is proved that the adoptedmodel minimizes the detection errors that lead to precise detection of intrusion in the networks.

Fig. 6 illustrates the analysis of other measures such as NPV, MCC, and F-measure of the adopted model than other traditional models. Under every variation in the K -value, the projected work attained the highest NPV, MCC, and F-measure. Moreover, the F-measure of the proposed work at K =4 is 26.3%, 15.7%, 13.6%, 7.89%, 21%, 23.5% and 23.5% better than the existing approaches like SGM-CN, NN, CNN, RNN, MULTI-CLASSIFIER+WOA, MULTI-CLASSIFIER+MFO, MULTI-CLASSIFIER+SOA, respectively. Therefore, the performance of the presented model has shown its improvement over other traditional approaches.

9.3 Evaluation of Proposed work with and without Incremental Learning

Table I and Table II manifest the performance evaluation of the projected work for dataset-1 (CICIDS2018) and dataset-2 in terms of incremental outcomes. The Proposed work with multi-classifier and no incremental learning is evaluated over the proposed work with multi-classifier and with incremental learning based over-sampling for varying k-values. At K =1, the accuracy of the Proposed work with a multi-classifier and no incremental learning is 0.43% better than the accuracy of the proposed work with a multi-classifier and with incremental learning-based over-sampling. In addition, the accuracy of the Proposed work with multi-classifier and no incremental learning at K =2 is 0.849399815, which is 0.18% better than the accuracy of the proposed work with multi-classifier and with incremental learning-based over-sampling. At K =4, the accuracy of the Proposed work with a multi-classifier and no incremental learning is 0.206% better than the accuracy of the proposed work with a multi-classifier and with incremental learning-based over-sampling. For dataset-2 (UNSWNB15), the performance evaluation of the Proposed work with multi-classifier and no incremental learning and with Proposed work with multi-classifier and with incremental learning is evaluated for varying K-values, and the outcomes acquired are noted in Table II. At K =2, the accuracy of the Proposed work with a multi-classifier and no incremental learning is 0.20% better than the accuracy of the proposed work with a multi-classifier and with incremental learning-based over-sampling. At K =2, the precision of the Proposed work with a multi-classifier and no incremental learning is 0.604306, which is better than the precision of the proposed work with a multi-classifier and with incremental learning-based over-sampling. Thus, from the analysis, it is clear that the proposed work with incremental learning performs better than the proposed work without incremental learning.

Table 1. *Evaluation of the proposed work with and without incremental learning for varying K -values: Dataset 1*

For K =1		
Measures	Proposed multi-classifier (SVM, Optimized DBN, NB and RF) without incremental learning	Proposed multi-classifier (SVM, Optimized DBN, NB, and RF) with incremental learning
Sensitivity	0.779201102	0.782644628
Specificity	0.892847222	0.894583333
Accuracy	0.854755309	0.857063712
Precision	0.785694444	0.789166667
F-Measure	0.782434302	0.785892116
MCC	0.653975844	0.620692716
NPV	0.889142462	0.890871369
FPR	0.107152778	0.105416667
FNR	0.220798898	0.217355372
For K =2		
Sensitivity	0.77107438	0.771212121
Specificity	0.88875	0.888819444
Accuracy	0.849307479	0.849399815
Precision	0.7775	0.777638889
F-Measure	0.774273859	0.774412172
MCC	0.600121379	0.649908252
NPV	0.885062241	0.885131397
FPR	0.11125	0.111180556
FNR	0.22892562	0.228787879
For K =3		
Sensitivity	0.769972452	0.769972452
Specificity	0.888194444	0.888194444
Accuracy	0.84856879	0.84856879
Precision	0.776388889	0.776388889
F-Measure	0.773167358	0.773167358
MCC	0.637587175	0.639202369
NPV	0.88450899	0.88450899
FPR	0.111805556	0.111805556
FNR	0.230027548	0.230027548
For K =4		
Sensitivity	0.779889807	0.773140496
Specificity	0.893194444	0.889791667
Accuracy	0.85521699	0.850692521
Precision	0.786388889	0.779583333
F-Measure	0.783125864	0.776348548
MCC	0.653505691	0.636642545
NPV	0.889488243	0.886099585
FPR	0.106805556	0.110208333
FNR	0.220110193	0.226859504
For K =5		
Sensitivity	0.775757576	0.773140496
Specificity	0.891111111	0.889791667
Accuracy	0.852446907	0.850692521
Precision	0.782222222	0.779583333
F-Measure	0.778976487	0.776348548
MCC	0.631870348	0.619152745
NPV	0.887413555	0.886099585
FPR	0.108888889	0.110208333
FNR	0.224242424	0.226859504

Table 2. *Evaluation of the proposed work with and without incremental learning for varying K -values: Dataset 2*

Measures	Proposed multi-classifier (SVM, Optimized DBN, NB, and RF) without incremental learning	Proposed multi-classifier (SVM, Optimized DBN, NB, and RF) without incremental learning with incremental learning
	For K =1	
Sensitivity	0.591873	0.586226
Specificity	0.798403	0.795556
Accuracy	0.729178	0.725392
Precision	0.596806	0.591111
F-Measure	0.594329	0.588658
MCC	0.484718	0.534811
NPV	0.79509	0.792254
FPR	0.201597	0.204444
FNR	0.408127	0.413774
	For K =2	
Sensitivity	0.596832	0.599311
Specificity	0.800903	0.802153
Accuracy	0.732502	0.734164
Precision	0.601806	0.604306
F-Measure	0.599308	0.601798
MCC	0.563322	0.517572
NPV	0.79758	0.798824
FPR	0.199097	0.197847
FNR	0.403168	0.400689
	For K =3	
Sensitivity	0.591736	0.592562
Specificity	0.798333	0.79875
Accuracy	0.729086	0.72964
Precision	0.596667	0.5975
F-Measure	0.594191	0.595021
MCC	0.541734	0.55423
NPV	0.795021	0.795436
FPR	0.201667	0.20125
FNR	0.408264	0.407438
	For K =4	
Sensitivity	0.58719	0.589394
Specificity	0.796042	0.797153
Accuracy	0.726039	0.727516
Precision	0.592083	0.594306
F-Measure	0.589627	0.59184
MCC	0.521296	0.549793
NPV	0.792739	0.793845
FPR	0.203958	0.202847
FNR	0.41281	0.410606
	For K =5	
Sensitivity	0.594077	0.592149
Specificity	0.799514	0.798542
Accuracy	0.730656	0.729363
Precision	0.599028	0.597083
F-Measure	0.596542	0.594606
MCC	0.546853	0.512672
NPV	0.796196	0.795228
FPR	0.200486	0.201458
FNR	0.405923	0.407851

9.4 Performance Analysis of Proposed Work: Existing Feature Set

The performance of the classifiers is verified by training them with an existing feature set (Statistical Features, EMA, DEMA, Existing Holoentropy, Percentile, and Moment)in terms of "Accuracy, Sensitivity, Specificity, Precision, F-measure, FDR, FNR, FPR, NPV, and MCC" respectively for varying k-values. This evaluation is undergone for dataset1 (CICIDS2018) and dataset2 (UNSWNB15.

Analysis of dataset1: The evaluation of dataset1 (CICIDS2018) for "Accuracy, Sensitivity, Specificity, Precision, F-measure, FDR, FNR, FPR, NPV, and MCC" corresponding to diverse K -values is shown in Table III-Table VII. For K =1, the accuracy of the projected work trained with existing feature set is 26.4%, 62.7%, 19%, 12.87%, 8.5%, 7.57% and 8.5% better than the existing models trained with existing feature set like SGM-CNN[1], NN, CNN, RNN, MULTI-CLASSIFIER+WOA, MULTI-

CLASSIFIER+MFO, MULTI-CLASSIFIER+SOA. In addition, F-measure of the proposed work trained with existing feature set at k=1 is 24.6%, 61.8%, 17%, 10.7%, 6.3%, 5.3% and 6.3% better than the existing models trained with existing feature set like SGM-CNN[1], NN, CNN, RNN, MULTI-CLASSIFIER+WOA, MULTI-CLASSIFIER+MFO, MULTI-CLASSIFIER+SOA, respectively. For K =2, the accuracy of the proposed trained with existing feature setis 0.893481, which is better than the existing models trained with the existing feature set like SGM-CNN [1] = 0.678486, NN=0.576547, CNN=0.761219, RNN=0.816251, MULTI-CLASSIFIER+WOA=0.848476, MULTI-CLASSIFIER+MFO=0.848476, MULTI-CLASSIFIER+SOA=0.851247 trained with existing feature set. Similarly, under all other variations in the k-value, the proposed work exhibited superior performance. This is owing to the balanced data utilized for attack detection.

For dataset 2: Similarly, in the case of dataset 2, the proposed work trained existing feature set for varying K -values (K =1 to 5) is shown in Table VII- Table XII respectively. On observing the outcomes the proposed work had exhibited the higher outcomes for every variation in the K -value. The proposed work had attained the maximal accuracy as 0.790269 at K =1. 0.793213 at K =2, 0.78967 at K =3, 0.795395 at K =4 and 0.792335 at K =5.

Table 3. *Evaluation of the proposed work trained with existing Feature set (Statistical Features, EMA, DEMA, Existing weighted Holoentropy, Percentile, and Moment) for dataset 1 corresponding to* K *=1*

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+WOA	MULTI-CLASSIFIER+MFO	MULTI-CLASSIFIER+SOA	MULTI-CLASSIFIER+ISOA
Sensitivity	0.619835	0.31405	0.682094	0.733884	0.770661	0.778512	0.770523	0.842311
Specificity	0.8125	0.658333	0.843889	0.87	0.888542	0.8925	0.888472	0.917995
Accuracy	0.747922	0.542936	0.789658	0.824377	0.84903	0.854294	0.848938	0.896598
Precision	0.625	0.316667	0.687778	0.74	0.777083	0.785	0.776944	0.814114
F-Measure	0.622407	0.315353	0.684924	0.736929	0.773859	0.781743	0.773721	0.826037
MCC	0.433231	0.336241	0.527073	0.605136	0.609086	0.600797	0.633659	0.599973
NPV	0.809129	0.655602	0.840387	0.86639	0.884855	0.888797	0.884786	0.935247
FPR	0.1875	0.341667	0.156111	0.13	0.111458	0.1075	0.111528	0.082005
FNR	0.380165	0.68595	0.317906	0.266116	0.229339	0.221488	0.229477	0.157689

Table 4. *Evaluation of the proposed work trained with existing Feature set (Statistical Features, EMA, DEMA, Existing weighted Holoentropy, Percentile, and Moment) for dataset 1 corresponding TO* K *=2*

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+WOA	MULTI-CLASSIFIER+MFO	MULTI-CLASSIFIER+SOA	MULTI-CLASSIFIER+ISOA
Sensitivity	0.516253	0.364187	0.639669	0.721763	0.769835	0.769835	0.773967	0.836999
Specificity	0.760278	0.683611	0.8225	0.863889	0.888125	0.888125	0.890208	0.915666
Accuracy	0.678486	0.576547	0.761219	0.816251	0.848476	0.848476	0.851247	0.893481
Precision	0.520556	0.367222	0.645	0.727778	0.77625	0.77625	0.780417	0.8087
F-Measure	0.518396	0.365698	0.642324	0.724758	0.773029	0.773029	0.777178	0.820624
MCC	0.312657	0.488977	0.463127	0.586866	0.620112	0.63481	0.621894	0.587377
NPV	0.757123	0.680775	0.819087	0.860304	0.88444	0.88444	0.886515	0.933181
FPR	0.239722	0.316389	0.1775	0.136111	0.111875	0.111875	0.109792	0.084334
FNR	0.483747	0.635813	0.360331	0.278237	0.230165	0.230165	0.226033	0.163001

Table 5. *Evaluation of the proposed work trained with existing Feature set (Statistical Features, EMA, DEMA, Existing weighted Holoentropy, Percentile, and Moment) for dataset 1 corresponding to* $K=3$

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+ WOA	MULTI-CLASSIFIER+ MFO	MULTI-CLASSIFIER+ SOA	MULTI-CLASSIFIER+ ISOA
Sensitivity	0.583471	0.312948	0.666667	0.379614	0.772039	0.772176	0.821488	0.84647
Specificity	0.794167	0.657778	0.836111	0.691389	0.889236	0.889306	0.914167	0.919765
Accuracy	0.723546	0.542198	0.779317	0.586888	0.849954	0.850046	0.883102	0.898967
Precision	0.588333	0.315556	0.672222	0.382778	0.778472	0.778611	0.828333	0.81839
F-Measure	0.585892	0.314246	0.669433	0.381189	0.775242	0.77538	0.824896	0.830306
MCC	0.37842	0.375978	0.50382	0.308193	0.662767	0.594033	0.626091	0.616246
NPV	0.790871	0.655048	0.832642	0.68852	0.885546	0.885615	0.910373	0.936756
FPR	0.205833	0.342222	0.163889	0.308611	0.110764	0.110694	0.085833	0.080235
FNR	0.416529	0.687052	0.333333	0.620386	0.227961	0.227824	0.178512	0.15353

Table 6. *Evaluation of the proposed work trained with existing Feature set (Statistical Features, EMA, DEMA, Existing weighted Holoentropy, Percentile, and Moment) for dataset 1 corresponding TO* $K=4$

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+ WOA	MULTI-CLASSIFIER+ MFO	MULTI-CLASSIFIER+ SOA	MULTI-CLASSIFIER+ ISOA
Sensitivity	0.723416	0.696419	0.63416	0.570799	0.77686	0.768044	0.769972	0.83513
Specificity	0.864722	0.851111	0.819722	0.787778	0.891667	0.887222	0.888194	0.914838
Accuracy	0.817359	0.799261	0.757525	0.715051	0.853186	0.847276	0.848569	0.892372
Precision	0.729444	0.702222	0.639444	0.575556	0.783333	0.774444	0.776389	0.806803
F-Measure	0.726418	0.699308	0.636791	0.573167	0.780083	0.771231	0.773167	0.818725
MCC	0.589357	0.548665	0.454823	0.35932	0.606099	0.613801	0.606741	0.6053
NPV	0.861134	0.84758	0.816321	0.784509	0.887967	0.883541	0.884509	0.932435
FPR	0.135278	0.148889	0.180278	0.212222	0.108333	0.112778	0.111806	0.085162
FNR	0.276584	0.303581	0.36584	0.429201	0.22314	0.231956	0.230028	0.16487

Table 7. *Evaluation of the proposed work trained with existing Feature set (Statistical Features, EMA, DEMA, Existing weighted Holoentropy, Percentile, and Moment) for dataset 1 corresponding TO* $K=5$

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+ WOA	MULTI-CLASSIFIER+ MFO	MULTI-CLASSIFIER+ SOA	MULTI-CLASSIFIER+ ISOA
Sensitivity	0.626446	0.334435	0.662259	0.370248	0.773554	0.770523	0.770523	0.841583
Specificity	0.815833	0.668611	0.833889	0.686667	0.89	0.888472	0.888472	0.917442
Accuracy	0.752355	0.556602	0.776362	0.580609	0.85097	0.848938	0.848938	0.895872
Precision	0.631667	0.337222	0.667778	0.373333	0.78	0.776944	0.776944	0.813424
F-Measure	0.629046	0.335823	0.665007	0.371784	0.776763	0.773721	0.773721	0.825357
MCC	0.443196	0.409471	0.497176	0.470823	0.61683	0.598398	0.613765	0.586867
NPV	0.812448	0.665837	0.830429	0.683817	0.886307	0.884786	0.884786	0.934553
FPR	0.184167	0.331389	0.166111	0.313333	0.11	0.111528	0.111528	0.082558
FNR	0.373554	0.665565	0.337741	0.629752	0.226446	0.229477	0.229477	0.158417

Table 8. *Evaluation of the proposed work trained with existing Feature set (Statistical Features, EMA, DEMA, Existing weighted Holoentropy, Percentile and Moment) for datset 2 corresponding TO K =1*

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+ WOA	MULTI-CLASSIFIER+ MFO	MULTI-CLASSIFIER+ SOA	MULTI-CLASSIFIER+ ISOA
Sensitivity	0.328926	0.334435	0.402755	0.322865	0.540634	0.542837	0.543526	0.679626
Specificity	0.665833	0.668611	0.703056	0.662778	0.772569	0.773681	0.774028	0.843012
Accuracy	0.552909	0.556602	0.602401	0.548846	0.694829	0.696307	0.696768	0.790269
Precision	0.331667	0.337222	0.406111	0.325556	0.545139	0.547361	0.548056	0.665999
F-Measure	0.33029	0.335823	0.404426	0.324205	0.542877	0.54509	0.545781	0.672445
MCC	0.390638	0.317403	0.309213	0.475815	0.456268	0.449025	0.4273	0.56598
NPV	0.663071	0.665837	0.700138	0.660028	0.769364	0.77047	0.770816	0.845971
FPR	0.334167	0.331389	0.296944	0.337222	0.227431	0.226319	0.225972	0.156988
FNR	0.671074	0.665565	0.597245	0.677135	0.459366	0.457163	0.456474	0.320374

Table 9. *Evaluation of the proposed work trained with existing Feature set (Statistical Features, EMA, DEMA, Existing weighted Holoentropy, Percentile, and Moment) for dataset 2 corresponding TO K =2*

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+ WOA	MULTI-CLASSIFIER+ MFO	MULTI-CLASSIFIER+ SOA	MULTI-CLASSIFIER+ ISOA
Sensitivity	0.450138	0.304683	0.490358	0.344904	0.547107	0.543113	0.547107	0.685137
Specificity	0.726944	0.653611	0.747222	0.673889	0.775833	0.773819	0.775833	0.84516
Accuracy	0.634164	0.536657	0.661127	0.56362	0.699169	0.696491	0.699169	0.793213
Precision	0.453889	0.307222	0.494444	0.347778	0.551667	0.547639	0.551667	0.672722
F-Measure	0.452006	0.305947	0.492393	0.346335	0.549378	0.545367	0.549378	0.678614
MCC	0.312728	0.329939	0.312948	0.337092	0.453992	0.428943	0.461712	0.575294
NPV	0.723928	0.650899	0.744122	0.671093	0.772614	0.770609	0.772614	0.847867
FPR	0.273056	0.346389	0.252778	0.326111	0.224167	0.226181	0.224167	0.15484
FNR	0.549862	0.695317	0.509642	0.655096	0.452893	0.456887	0.452893	0.314863

Table 10. *Evaluation of the Proposed Work Trained With Existing Feature Set (Statistical Features, Ema, Dema, Existing Weighted Holoentropy, Percentile, And Moment) For Dataset 2 Corresponding To=3*

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+ WOA	MULTI-CLASSIFIER+ MFO	MULTI-CLASSIFIER+ SOA	MULTI-CLASSIFIER+ ISOA
Sensitivity	0.338292	0.434711	0.400551	0.338292	0.543664	0.541873	0.539945	0.678914
Specificity	0.670556	0.719167	0.701944	0.670556	0.774097	0.773194	0.772222	0.84255
Accuracy	0.559187	0.623823	0.600923	0.559187	0.696861	0.69566	0.694367	0.78967
Precision	0.341111	0.438333	0.403889	0.341111	0.548194	0.546389	0.544444	0.665461
F-Measure	0.339696	0.436515	0.402213	0.339696	0.54592	0.544122	0.542185	0.671829
MCC	0.469637	0.380566	0.36512	0.459014	0.433159	0.435758	0.439827	0.53911
NPV	0.667773	0.716183	0.699032	0.667773	0.770885	0.769986	0.769018	0.845474
FPR	0.329444	0.280833	0.298056	0.329444	0.225903	0.226806	0.227778	0.15745
FNR	0.661708	0.565289	0.599449	0.661708	0.456336	0.458127	0.460055	0.321086

Table 11. *Evaluation of the proposed work trained with existing Feature set (Statistical Features, EMA, DEMA, Existing weighted Holoentropy, Percentile, and Moment) for dataset 2 corresponding TO K =4*

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+ WOA	MULTI-CLASSIFIER+ MFO	MULTI-CLASSIFIER+ SOA	MULTI-CLASSIFIER+ ISOA
Sensitivity	0.364187	0.323967	0.500275	0.372452	0.548209	0.553444	0.547245	0.687333
Specificity	0.683611	0.663333	0.752222	0.687778	0.776389	0.779028	0.775903	0.846863
Accuracy	0.576547	0.549584	0.667775	0.582087	0.699908	0.703416	0.699261	0.795395
Precision	0.367222	0.326667	0.504444	0.375556	0.552778	0.558056	0.551806	0.67383
F-Measure	0.365698	0.325311	0.502351	0.373997	0.550484	0.55574	0.549516	0.680217
MCC	0.324728	0.455355	0.333234	0.455909	0.439796	0.41512	0.428936	0.52882
NPV	0.680775	0.660581	0.749101	0.684924	0.773167	0.775795	0.772683	0.849793
FPR	0.316389	0.336667	0.247778	0.312222	0.223611	0.220972	0.224097	0.153137
FNR	0.635813	0.676033	0.499725	0.627548	0.451791	0.446556	0.452755	0.312667

Table 12. *Evaluation of the proposed work trained with existing Feature set (Statistical Features, EMA, DEMA, Existing weighted Holoentropy, Percentile, and Moment) for dataset 2 corresponding TO K =5*

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+ WOA	MULTI-CLASSIFIER+ MFO	MULTI-CLASSIFIER+ SOA	MULTI-CLASSIFIER+ ISOA
Sensitivity	0.381818	0.472176	0.32562	0.332231	0.551515	0.551653	0.541185	0.681996
Specificity	0.6925	0.738056	0.664167	0.6675	0.778056	0.778125	0.772847	0.844607
Accuracy	0.588366	0.648938	0.550693	0.555125	0.702124	0.702216	0.695199	0.792335
Precision	0.385	0.476111	0.328333	0.335	0.556111	0.55625	0.545694	0.667712
F-Measure	0.383402	0.474136	0.326971	0.33361	0.553804	0.553942	0.54343	0.674455
MCC	0.365465	0.440161	0.482189	0.312658	0.459247	0.448974	0.459967	0.552774
NPV	0.689627	0.734993	0.661411	0.66473	0.774827	0.774896	0.76964	0.847696
FPR	0.3075	0.261944	0.335833	0.3325	0.221944	0.221875	0.227153	0.155393
FNR	0.618182	0.527824	0.67438	0.667769	0.448485	0.448347	0.458815	0.318004

9.5 Performance Analysis of Proposed Work: Proposed Feature Set

The performance of the proposed work trained with the proposed feature set (Statistical Features, EMA, DEMA, IWH, Percentile, and Moment) is compared over the existing works like SGM-CNN[1], NN, CNN, RNN, MULTI-CLASSIFIER+WOA, MULTI-CLASSIFIER+MFO, MULTI-CLASSIFIER+SOA trained with the same proposed feature set (Statistical Features, EMA, DEMA, IWH, Percentile and Moment). This evaluation is carried out in terms of "Accuracy, Sensitivity, Specificity, Precision, F-measure, FDR, FNR, FPR, NPV, and MCC" respectively for varying k-values.This assessment is undergone for dataset1 (CICIDS2018) and dataset2 (UNSWNB15), respectively. The results acquired for dataset-1 are shown in Table XIII to Table XVIII respectively. In addition, the results acquired for Dataset 2 are shown in Tables XIX to XXIII. On observing the outcomes, the proposed work trained with the proposed feature set has attained the highest performance in the case of both datasets. This is owing to the introduction of a new IWH-based feature extraction mechanism, wherein the weight functions are computed for the input, rather than the standard value used in the standard holoentropy model. Moreover, the accuracy of the proposed work trained with the proposed feature set for dataset 1 corresponding to K =1 is 0.932433, which is the best value compared to the accuracy value of the proposed work trained with the existing feature set (0.896598). for dataset 1 corresponding to K =2, the proposed work trained with the proposed feature set had attained the accuracy value of 0.934932, which accuracy of the proposed work trained with the existing feature set 0.793213. Thus, from the evaluation, it is clear that the proposed work trained with the proposed feature set is much more efficient in detecting network intrusions.

Table 13. *Evaluation of the proposed work trained with the proposed Feature set (Statistical Features, EMA, DEMA, IWH, Percentile, and Moment) for dataset 1 corresponding to* K *=1*

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+ WOA	MULTI-CLASSIFIER+ MFO	MULTI-CLASSIFIER+ SOA	MULTI-CLASSIFIER+ ISOA
Sensitivity	0.710744	0.328926	0.827548	0.32562	0.773967	0.773967	0.769972	0.871877
Specificity	0.858333	0.665833	0.917222	0.664167	0.890208	0.890208	0.888194	0.955761
Accuracy	0.808864	0.552909	0.887165	0.550693	0.851247	0.851247	0.848569	0.932433
Precision	0.716667	0.331667	0.834444	0.328333	0.780417	0.780417	0.776389	0.88815
F-Measure	0.713693	0.33029	0.830982	0.326971	0.777178	0.777178	0.773167	0.879852
MCC	0.570257	0.305857	0.746314	0.365693	0.610707	0.617735	0.658654	0.861967
NPV	0.854772	0.663071	0.913416	0.661411	0.886515	0.886515	0.884509	0.948727
FPR	0.141667	0.334167	0.082778	0.335833	0.109792	0.109792	0.111806	0.044239
FNR	0.289256	0.671074	0.172452	0.67438	0.226033	0.226033	0.230028	0.128123

Table 14. *Evaluation of the proposed work trained with the proposed Feature set (Statistical Features, EMA, DEMA, IWH, Percentile, and Moment) for dataset 1 corresponding to* K *=2*

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+ WOA	MULTI-CLASSIFIER+ MFO	MULTI-CLASSIFIER+ SOA	MULTI-CLASSIFIER+ ISOA
Sensitivity	0.682645	0.801102	0.812672	0.844628	0.779339	0.779201	0.779201	0.876952
Specificity	0.844167	0.903889	0.909722	0.925833	0.892917	0.892847	0.892847	0.957446
Accuracy	0.790028	0.869437	0.877193	0.898615	0.854848	0.854755	0.854755	0.934932
Precision	0.688333	0.807778	0.819444	0.851667	0.785833	0.785694	0.785694	0.892979
F-Measure	0.685477	0.804426	0.816044	0.848133	0.782573	0.782434	0.782434	0.884814
MCC	0.527903	0.706452	0.723892	0.772058	0.637256	0.621643	0.630496	0.813399
NPV	0.840664	0.900138	0.905947	0.921992	0.889212	0.889142	0.889142	0.950495
FPR	0.155833	0.096111	0.090278	0.074167	0.107083	0.107153	0.107153	0.042554
FNR	0.317355	0.198898	0.187328	0.155372	0.220661	0.220799	0.220799	0.123048

Table 15. *Evaluation of the proposed work trained with the proposed Feature set (Statistical Features, EMA, DEMA, IWH, Percentile, and Moment) for dataset 1 corresponding to* K *=3*

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+ WOA	MULTI-CLASSIFIER+ MFO	MULTI-CLASSIFIER+ SOA	MULTI-CLASSIFIER+ ISOA
Sensitivity	0.788981	0.627548	0.816529	0.743802	0.774793	0.774793	0.767631	0.872373
Specificity	0.897778	0.816389	0.911667	0.875	0.890625	0.890625	0.887014	0.955056
Accuracy	0.861311	0.753093	0.879778	0.831025	0.851801	0.851801	0.846999	0.931779
Precision	0.795556	0.632778	0.823333	0.75	0.78125	0.78125	0.774028	0.888335
F-Measure	0.792254	0.630152	0.819917	0.746888	0.778008	0.778008	0.770816	0.880203
MCC	0.688182	0.444857	0.729705	0.620084	0.600587	0.589589	0.646672	0.807347
NPV	0.894053	0.813001	0.907884	0.871369	0.886929	0.886929	0.883333	0.948124
FPR	0.102222	0.183611	0.088333	0.125	0.109375	0.109375	0.112986	0.044944
FNR	0.211019	0.372452	0.183471	0.256198	0.225207	0.225207	0.232369	0.127627

Table 16. *Evaluation of the proposed work trained with the proposed Feature set (Statistical Features, EMA, DEMA, IWH, Percentile, and Moment) for dataset 1 corresponding to* $K=4$

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+ WOA	MULTI-CLASSIFIER+ MFO	MULTI-CLASSIFIER+ SOA	MULTI-CLASSIFIER+ ISOA
Sensitivity	0.75978	0.808815	0.82259	0.472727	0.778375	0.774105	0.774242	0.876174
Specificity	0.883056	0.907778	0.914722	0.738333	0.892431	0.890278	0.890347	0.957176
Accuracy	0.841736	0.874608	0.883841	0.649307	0.854201	0.851339	0.851431	0.934536
Precision	0.766111	0.815556	0.829444	0.476667	0.784861	0.780556	0.780694	0.892236
F-Measure	0.762932	0.812172	0.826003	0.474689	0.781604	0.777317	0.777455	0.884052
MCC	0.644167	0.718078	0.73884	0.460451	0.63103	0.633247	0.622146	0.883075
NPV	0.879391	0.804011	0.910927	0.73527	0.888728	0.886584	0.886653	0.950213
FPR	0.116944	0.092222	0.085278	0.261667	0.107569	0.109722	0.109653	0.042824
FNR	0.24022	0.191185	0.17741	0.527273	0.221625	0.225895	0.225758	0.123826

Table 17. *Evaluation of the proposed work trained with the proposed Feature set (Statistical Features, EMA, DEMA, IWH, Percentile, and Moment) for dataset 1 corresponding to* $K=5$

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+ WOA	MULTI-CLASSIFIER+ MFO	MULTI-CLASSIFIER+ SOA	MULTI-CLASSIFIER+ ISOA
Sensitivity	0.699725	0.80832	0.81157	0.808264	0.782369	0.782369	0.77741	0.879728
Specificity	0.852778	0.845278	0.909167	0.9075	0.894444	0.894444	0.891944	0.958288
Accuracy	0.801477	0.892447	0.876454	0.874238	0.856879	0.856879	0.853555	0.936219
Precision	0.705556	0.855556	0.818333	0.815	0.788889	0.788889	0.783889	0.895587
F-Measure	0.702628	0.868603	0.814938	0.811618	0.785615	0.785615	0.780636	0.887512
MCC	0.553647	0.83019	0.722231	0.717248	0.606465	0.651928	0.639113	0.894271
NPV	0.849239	0.894136	0.905394	0.903734	0.890733	0.890733	0.888243	0.951393
FPR	0.147222	0.054722	0.090833	0.0925	0.105556	0.105556	0.108056	0.041712
FNR	0.300275	0.168044	0.18843	0.191736	0.217631	0.217631	0.22259	0.120272

Table 18. *Evaluation of the proposed work trained with the proposed Feature set (Statistical Features, EMA, DEMA, IWH, Percentile, and Moment) for dataset 2 corresponding to* $K=1$

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+ WOA	MULTI-CLASSIFIER+ MFO	MULTI-CLASSIFIER+ SOA	MULTI-CLASSIFIER+ ISOA
Sensitivity	0.530028	0.373003	0.511846	0.310744	0.595317	0.595041	0.591598	0.753959
Specificity	0.767222	0.688056	0.758056	0.656667	0.800139	0.8	0.798264	0.88132
Accuracy	0.687719	0.582456	0.675531	0.54072	0.731487	0.731302	0.728994	0.839238
Precision	0.534444	0.376111	0.516111	0.313333	0.600278	0.6	0.596528	0.761782
F-Measure	0.532227	0.37455	0.51397	0.312033	0.597787	0.59751	0.594053	0.757849
MCC	0.368424	0.485787	0.376506	0.353878	0.523316	0.520986	0.543582	0.68397
NPV	0.764039	0.685201	0.75491	0.653942	0.796819	0.79668	0.794952	0.877734
FPR	0.232778	0.311944	0.241944	0.343333	0.199861	0.2	0.201736	0.11868
FNR	0.469972	0.626997	0.488154	0.689256	0.404683	0.404959	0.408402	0.246041

Table 19. *Evaluation of the proposed work trained with the proposed Feature set (Statistical Features, EMA, DEMA, IWH, Percentile, and Moment) for dataset 2 corresponding to* $K=2$

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+ WOA	MULTI-CLASSIFIER+ MFO	MULTI-CLASSIFIER+ SOA	MULTI-CLASSIFIER+ ISOA
Sensitivity	0.489256	0.332782	0.497521	0.327824	0.586777	0.587603	0.586777	0.747891
Specificity	0.746667	0.667778	0.750833	0.665278	0.795833	0.79625	0.795833	0.878517
Accuracy	0.660388	0.555494	0.665928	0.55217	0.725762	0.726316	0.725762	0.835447
Precision	0.493333	0.335556	0.501667	0.330556	0.591667	0.5925	0.591667	0.756035
F-Measure	0.491286	0.334163	0.499585	0.329184	0.589212	0.590041	0.589212	0.751938
MCC	0.305657	0.396763	0.315754	0.321822	0.517907	0.506909	0.527985	0.671511
NPV	0.743568	0.665007	0.747718	0.662517	0.792531	0.792946	0.792531	0.874892
FPR	0.253333	0.332222	0.249167	0.334722	0.204167	0.20375	0.204167	0.121483
FNR	0.510744	0.667218	0.502479	0.672176	0.413223	0.412397	0.413223	0.252109

Table 20. *Evaluation of the proposed work trained with the proposed Feature set (Statistical Features, EMA, DEMA, IWH, Percentile, and Moment) for dataset 2 corresponding to* $K=3$

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+ WOA	MULTI-CLASSIFIER+ MFO	MULTI-CLASSIFIER+ SOA	MULTI-CLASSIFIER+ ISOA
Sensitivity	0.505234	0.338843	0.513499	0.326722	0.598072	0.597796	0.597934	0.756804
Specificity	0.754722	0.670833	0.758889	0.664722	0.801528	0.801389	0.801458	0.883093
Accuracy	0.671099	0.559557	0.676639	0.551431	0.733333	0.733149	0.733241	0.841512
Precision	0.509444	0.341667	0.517778	0.329444	0.603056	0.602778	0.602917	0.76514
F-Measure	0.507331	0.340249	0.515629	0.328077	0.600553	0.600277	0.600415	0.760947
MCC	0.464472	0.417056	0.375705	0.407199	0.498984	0.511917	0.478388	0.64223
NPV	0.751591	0.66805	0.75574	0.661964	0.798202	0.798064	0.798133	0.879433
FPR	0.245278	0.329167	0.241111	0.335278	0.198472	0.198611	0.198542	0.116907
FNR	0.494766	0.661157	0.486501	0.673278	0.401928	0.402204	0.402066	0.243196

Table 21. *Evaluation of the proposed work trained with the proposed Feature set (Statistical Features, EMA, DEMA, IWH, Percentile, and Moment) for dataset 2 corresponding to* $K=4$

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+ WOA	MULTI-CLASSIFIER+ MFO	MULTI-CLASSIFIER+ SOA	MULTI-CLASSIFIER+ ISOA
Sensitivity	0.530028	0.324518	0.507989	0.332782	0.599174	0.596419	0.598623	0.759461
Specificity	0.767222	0.663611	0.756111	0.667778	0.802083	0.800694	0.801806	0.88435
Accuracy	0.687719	0.549954	0.672946	0.555494	0.734072	0.732225	0.733703	0.843204
Precision	0.534444	0.327222	0.512222	0.335556	0.604167	0.601389	0.603611	0.767702
F-Measure	0.532227	0.325864	0.510097	0.334163	0.60166	0.598893	0.601107	0.763557
MCC	0.385805	0.454716	0.304809	0.315268	0.503154	0.541694	0.518478	0.694757
NPV	0.764039	0.660858	0.752974	0.665007	0.798755	0.797372	0.798479	0.880701
FPR	0.232778	0.336389	0.243889	0.332222	0.197917	0.199306	0.198194	0.11565
FNR	0.469972	0.675482	0.492011	0.667218	0.400826	0.403581	0.401377	0.240539

Table 22. *Evaluation of the proposed work trained with the proposed Feature set (Statistical Features, EMA, DEMA, IWH, Percentile, and Moment) for dataset 2 corresponding to* K *=5*

Measures	SGM-CNN [1]	NN	CNN	RNN	MULTI-CLASSIFIER+WOA	MULTI-CLASSIFIER+MFO	MULTI-CLASSIFIER+SOA	MULTI-CLASSIFIER+ISOA
Sensitivity	0.52011	0.363085	0.512948	0.32011	0.588567	0.585262	0.59022	0.751886
Specificity	0.762222	0.683056	0.758611	0.661389	0.796736	0.795069	0.797569	0.879418
Accuracy	0.681071	0.575808	0.67627	0.546999	0.726962	0.724746	0.72807	0.836957
Precision	0.524444	0.366111	0.517222	0.322778	0.593472	0.590139	0.595139	0.75841
F-Measure	0.522268	0.364592	0.515076	0.321438	0.59101	0.58769	0.592669	0.755134
MCC	0.385089	0.422662	0.496873	0.409297	0.509142	0.522074	0.501478	0.658644
NPV	0.759059	0.680221	0.755463	0.658645	0.79343	0.79177	0.79426	0.87606
FPR	0.237778	0.316944	0.241389	0.338611	0.203264	0.204931	0.202431	0.120582
FNR	0.47989	0.636915	0.487052	0.67989	0.411433	0.414738	0.40978	0.248114

10. Conclusion

This paper proposed an intrusion detection model for class imbalance data by following four major phases: (a) Pre-processing, (b) Imbalance processing, (c) Feature extraction, and (d) Intrusion detection phase. Initially, the collected class imbalance data was pre-processed via data cleaning and data standardization approach. Then, the pre-processed class imbalance data was more balanced in the imbalanced processing phase by a new improved over-sampling technique using SMOTE and the Multi-kernel FCM clustering model. Subsequently, multi-features like EMA, DEMA, IWH, and statistical features (Mean, Median, Standard deviation, percentile, and moment) were extracted from these balanced data. The extracted overall features are given for K-fold validation. Then, the intrusion detection phase was modeled with a combination of four individual ML models such as SVM, optimized DBN with incremental learning, NB, and RF to achieve the utmost detection performance. Each of the classifiers is trained with the acquired K-fold data features. Moreover, to achieve the utmost detection accuracy as well as better tradeoff performance, the weight of DBN was fine-tuned via a new SI-SOA, which was an improved version of standard SOA. Then, the logoff performance was computed from the outcome acquired from each of the individual classifiers (SVM, DBN, NB, and RF). Finally, the log computed result will portray the detected attacks in the network. The performance of the proposed work (MULTI-CLASSIFIER+SI-SOA) is compared over the existing models like SGM-CNN[1], NN, CNN, RNN, MULTI-CLASSIFIER+WOA, MULTI-CLASSIFIER+MFO, MULTI-CLASSIFIER+SOA respectively. In this research work, we have fixed the K-value of K -fold evaluation as 5(K =5), and we've varied this K value from 1, 2, 3, 4, and 5, to evaluate the performance of the projected model. Furthermore, the performance was based on the different performance measures including "accuracy, sensitivity, specificity, precision, F-measure, FDR, FNR, FPR, NPV, and MCC". The accuracy of the projected model is higher for every variation in the K -values. At K =5, the proposed work attained the highest accuracy value as 92%, which is 13%,9.7%, 11.4%, 11.55,11.6%, 11.6% and 11.8% better than the accuracy value recorded by the existing models like SGM-CNN, NN, CNN, RNN, MULTI-CLASSIFIER+WOA, MULTI-CLASSIFIER+MFO, MULTI-CLASSIFIER+SOA respectively.

Compliance with Ethical Standards

Conflicts of interest: Authors declared that they have no conflict of interest.

Human participants: The conducted research follows the ethical standards and the authors ensured that they have not conducted any studies with human participants or animals.

References

[1] M. J. Moradi, M. Khaleghi, A. M. Ramezanianpour, "Predicting the compressive strength of concrete containing metakaolin with different properties using ANN", Measurement, 2021
[2] Xiao-hui Zeng, Xu-li Lan, You-jun Xie, "Investigation on the air-void structure and compressive strength of concrete at low atmospheric pressure", Cement and Concrete Composites, 2021

[3] Liu Jin, Jian Li, Xiuli Du, "Size effect modeling for dynamic biaxial compressive strength of concrete: Influence of lateral stress ratio and strain rate", International Journal of Impact Engineering, 2021

[4] Stephen Adeyemi Alabi, Jeffrey Mahachi, "Compressive strength of concrete containing palm oil fuel ash under different curing techniques", Materials Today: Proceedings, 2020

[5] P. Mahakavi, R. Chithra, "Effect of RCA, foundry sand on strength and toughness of fiber reinforced self-compacting concrete", Materials Today: Proceedings,2021

[6] J. Sliseris & A. Korjakins, "Numerical Modeling of the Casting Process and Impact Loading of a Steel-Fiber-Reinforced High-Performance Self-Compacting Concrete", Mechanics of Composite Materials, 2019

[7] Seung-Ho Choi, Deuck Hang Lee, Jin-Ha Hwang, Jae-Yuel Oh, Kang Su Kim & Sang-Ho Kim, "Effective compressive strengths of corner and exterior concrete columns intersected by slabs with different compressive strengths", Archives of Civil and Mechanical Engineering, 2018

[8] F. H. Chiew, "Prediction of Blast Furnace Slag Concrete Compressive Strength Using Artificial Neural Networks and Multiple Regression Analysis," *2019 International Conference on Computer and Drone Applications (IConDA)*, Kuching, Malaysia, 2019. doi: 10.1109/IConDA47345.2019.9034920

[9] S. Ma, X. Shi, C. Yu, W. Wang, Y. Ren, and X. Tian, "Research on Improved BP Neural Network Gangue Powder Concrete Compressive Strength Prediction Model," *2020 IEEE 9th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*, Chongqing, China, 2020. doi: 10.1109/ITAIC49862.2020.9339112

[10] P. Singh and P. Khaskil, "Prediction of Compressive Strength of Green Concrete with Admixtures Using Neural Networks," *2020 IEEE International Conference on Computing, Power and Communication Technologies (GUCON)*, Greater Noida, India, 2020.doi: 10.1109/GUCON48875.2020.9231230

[11] J. Zhang and Y. Zhao, "Prediction of Compressive Strength of Ultra-High Performance Concrete (UHPC) Containing Supplementary Cementitious Materials," *2017 International Conference on Smart Grid and Electrical Automation (ICSGEA)*, Changsha, China, 2017. doi: 10.1109/ICSGEA.2017.150

[12] M. KAYA KELEŞ, A. E. KELEŞ and Ü. KILIÇ, "PREDICTION OF CONCRETE STRENGTH WITH DATA MINING METHODS USING ARTIFICIAL BEE COLONY AS FEATURE SELECTOR," *2018 International Conference on Artificial Intelligence and Data Processing (IDAP)*, Malatya, Turkey, 2018. doi: 10.1109/IDAP.2018.8620905

[13] Guan Ping and Liu Peng, "Study of strength criterion for dynamic biaxial compressive properties of concrete under constant confining pressure," *2011 International Conference on Electric Technology and Civil Engineering (ICETCE)*, Lushan, China, 2011. doi: 10.1109/ICETCE.2011.5775828

[14] J. A. Abdalla, R. Hawileh and A. Al-Tamimi, "Prediction of FRP-concrete ultimate bond strength using Artificial Neural Network," *2011 Fourth International Conference on Modeling, Simulation and Applied Optimization*, Kuala Lumpur, Malaysia, 011 doi: 10.1109/ICMSAO.2011.5775518

[15] R. Wang, Y. Dai, C. Han, K. Xu, and L. Dong, "Application of DPO—BP in strength prediction of concrete," *2017 IEEE 3rd Information Technology and Mechatronics Engineering Conference (ITOEC)*, Chongqing, China, 2017. doi: 10.1109/ITOEC.2017.8122505

[16] S. Saad, M. Ishtiyaque and H. Malik, "Selection of most relevant input parameters using WEKA for artificial neural network based concrete compressive strength prediction model," *2016 IEEE 7th Power India International Conference (PIICON)*, Bikaner, India, 2016. doi: 10.1109/POWERI.2016.8077368

[17] M. A. Shafiq, "Predicting the compressive strength of concrete using neural network and kernel ridge regression," *2016 Future Technologies Conference (FTC)*, San Francisco, CA, USA, 2016.

[18] Xiaoyun Zhang, H. Wang, Delin Wang, and Chendong Li, "Prediction of concrete strength based on self-organizing fuzzy neural network," *Proceeding of the 11th World Congress on Intelligent Control and Automation*, Shenyang, China, 2014. doi: 10.1109/WCICA.2014.7053679

[19] K. L. Chung, C. Zhang, and S. Kharkovsky, "Microwave near-field noninvasive prediction of compressive strength of Engineered Cementitious Composites (ECCs)," *TENCON 2015 - 2015 IEEE Region 10 Conference*, Macao, China, 2015. doi: 10.1109/TENCON.2015.7373059

[20] D. K. Sinha, S. Rupali, and S. Bawa, "Application of Adaptive Neuro-Fuzzy Inference System for the Prediction of Early Age Strength of High-Performance Concrete," *2019 International Conference on Data Science and Engineering (ICDSE)*, Patna, India, 2019. doi: 10.1109/ICDSE47409.2019.8971798

[21] K. Poongodi *et al.*, "ANN-based prediction of Bond and Impact Strength of Light Weight Self Consolidating Concrete with coconut shell," *2018 International Conference on Intelligent Computing and Communication for Smart World (I2C2SW)*, Erode, India, 2018, pp. 364-370. doi: 10.1109/I2C2SW45816.2018.8997421

[22] D. L. Silva, K. L. M. de Jesus, B. S. Villaverde, C. J. C. Cahilig, J. P. L. Dela Cruz and J. M. S. Sario, "Sensitivity Analysis and Strength Prediction of Fly Ash — Based Geopolymer Concrete with Polyethylene Terephtalate using Artificial Neural Network," *2020 IEEE 12th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*, Manila, Philippines, 2020, pp. 1-6. doi: 10.1109/HNICEM51456.2020.9400076

[23] Y. Li, J. Gou, and Z. Fan, "Particle swarm optimization-based extreme gradient boosting for concrete strength prediction," *2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference*

(IAEAC), Chengdu, China, 2019, pp. 982-986. doi: 10.1109/IAEAC47372.2019.8997825

[24] J. J. Regin, P. Vincent, D. S. Shiny and L. Porcia, "Neural Network Prediction of Compressive Strength of Lightweight Coconut Shell Concrete," *2019 International Conference on Recent Advances in Energy-efficient Computing and Communication (ICRAECC)*, Nagercoil, India, 2019, pp. 1-7. doi: 10.1109/ICRAECC43874.2019.8995134

[25] V. Veloso De Melo and W. Banzhaf, "Predicting High-Performance Concrete Compressive Strength Using Features Constructed by Kaizen Programming," *2015 Brazilian Conference on Intelligent Systems (BRACIS)*, Natal, Brazil, 2015, pp. 80-85. doi: 10.1109/BRACIS.2015.56

[26] Y. Bigdeli and M. Barbato, "Use of a low-cost concrete-like fluorogypsum-based blend for applications in underwater and coastal protection structures," *OCEANS 2017 - Anchorage*, Anchorage, AK, USA, 2017, pp. 1-5.

[27] S. J. C. Clemente, M. G. M. Ventanilla, E. P. Dadios and A. W. C. Oreta, "Feed Forward Back Propagation Artificial Neural Network Modeling of Compressive Strength of Self-Compacting Concrete," *2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM)*, Baguio City, Philippines, 2018, pp. 1-5. doi: 10.1109/HNICEM.2018.8666407

[28] Neenavath Veeraiah and Dr.B.T.Krishna, "Intrusion Detection Based on Piecewise Fuzzy C-Means Clustering and Fuzzy Naive Bayes Rule", Multimedia Research, Vol.1, No.1, pp.27-32,2018.

[29] Gaurav Dhiman, Vijay Kumar, "Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems", Knowledge-Based Systems, Vol.165, February 2019.

[30] Dataset 1, from :" https://www.analyticsvidhya.com/blog/2017/03/imbalanced-data-classification/", access date: 2021-08-10

[31] Datset2, from : "https://www.kdnuggets.com/2017/06/7-techniques-handle-imbalanced-data.html", access date: 2021-08-10

[32] S. M. Swamy, B. R. Rajakumar and I. R. Valarmathi, "Design of Hybrid Wind and Photovoltaic Power System using Opposition-based Genetic Algorithm with Cauchy Mutation", IET Chennai Fourth International Conference on Sustainable Energy and Intelligent Systems (SEISCON 2013), Chennai, India, Dec. 2013, DOI: 10.1049/ic.2013.0361

[33] G.Gokulkumari, "Classification of Brain tumor using Manta Ray Foraging Optimization-based DeepCNN classifier", Multimedia Research, Vol 3, No 4, 2020.

[34] B. R. Rajakumar, "Impact of Static and Adaptive Mutation Techniques on Genetic Algorithm", International Journal of Hybrid Intelligent Systems, Vol. 10, No. 1, pages: 11-22, 2013, DOI: 10.3233/HIS-120161

[35] B. R. Rajakumar, "Static and Adaptive Mutation Techniques for Genetic Algorithm: A Systematic Comparative Analysis", International Journal of Computational Science and Engineering, Vol. 8, No. 2, pages: 180-193, 2013, DOI: 10.1504/IJCSE.2013.053087

[36] Skewness, from:" https://en.wikipedia.org/wiki/Skewness",

[37] Kurtosis from :" https://en.wikipedia.org/wiki/Kurtosis".

[38] EMA,from :" https://en.wikipedia.org/wiki/Moving_average"

[39] Renjith Thomas and Dr. MJS. Rangachar, "Fractional Rider and Multi-Kernel-Based Spherical SVM for Low Resolution Face Recognition", Multimedia Research, Vol.2, No.2, pp.35-43,2019.

[40] Renjith Thomas and MJS. Rangachar, "Hybrid Optimization based DBN for Face Recognition using Low-Resolution Images", Multimedia Research, Vol.1, No.1, pp.33-43,2018.

[41] Cao, B., Li, C., Song, Y., Qin, Y. and Chen, C., "Network Intrusion Detection Model Based on CNN and GRU", Applied Sciences, Vol. 12(9), pp.4184, 2022.

[42] Meliboev, A., Alikhanov, J. and Kim, W., "Performance evaluation of deep learning based network intrusion detection system across multiple balanced and imbalanced datasets", Electronics, Vol. 11(4), pp.515, 2022.

[43] Fu, Y., Du, Y., Cao, Z., Li, Q. and Xiang, W., "A deep learning model for network intrusion detection with imbalanced data. Electronics", Vol. 11(6), pp.898, 2022.

[44] Jung, I., Ji, J. and Cho, C., "EmSM: ensemble mixed sampling method for classifying imbalanced intrusion detection data", Electronics, Vol. 11(9), pp.1346, 2022.

[45] Lin, Y.D., Liu, Z.Q., Hwang, R.H., Nguyen, V.L., Lin, P.C. and Lai, Y.C., "Machine learning with variational Auto Encoder for imbalanced datasets in intrusion detection", IEEE Access, Vol. 10, pp.15247-15260, 2022.

[46] Cui, J., Zong, L., Xie, J. and Tang, M., "A novel multi-module integrated intrusion detection system for high-dimensional imbalanced data", Applied Intelligence, pp.1-17, 2022.

[47] Balyan, A.K., Ahuja, S., Lilhore, U.K., Sharma, S.K., Manoharan, P., Algarni, A.D., Elmannai, H. and Raahemifar, K., "A hybrid intrusion detection model using ega-pso and improved random forest method", Sensors, Vol. 22(16), pp.5986, 2022.

[48] Le, T.T.H., Oktian, Y.E. and Kim, H., "XGBoost for imbalanced multiclass classification-based industrial internet of things intrusion detection systems", Sustainability, Vol. 14(14), pp.8707, 2022.