



Neural Network-based Hybrid Feature Extraction Method for Network Intrusion Detection Systems

Geeta Kocher^{1*}, Gulshan Kumar²

Abstract

Anomaly-based intrusion detection systems are a significant research focus of security practitioners. These systems detect normal and malignant network traffic behaviour and may identify new threats. Unfortunately, Intrusion detection system (IDS) efficiency relies mainly on feature space design, and accurate modelling of the feature set in network traffic is still a research problem. The high dimension features space and a considerable quantity of noisy data significantly impact the accuracy and efficiency of security mechanisms. Additionally, most network traffic dataset is imbalanced, significantly reducing IDS performance. A deep learning-based feed-forward neural network (DLFFNN) followed by feature extraction and selection approach, and a data balancing approach is proposed to address all these issues. This work proposes a model concept-based K-means clustering with information gain (KMC-IG) for feature extraction, ranking, and feature reduction. Secondly, a Synthetic Minority Over-sampling Technique and Edited Nearest Neighbour enable data balancing, and data cleaning techniques to balance the dataset are introduced. The third and most critical step is intrusion detection and network flow classification. For this, a DLFFNN model that identifies attack patterns in different scenarios and classifies the network traffic data into binary and multiple classes is proposed. The DLFFNN model is inspired by the human brain's structure and includes many processors working in parallel, arranged in layers.

To evaluate the performance of the proposed DLFFNN-KMC-IG approach, three standard datasets are considered in this work, including NSL-KDD, UNSW-NB 15, and CICIDS2017. For each dataset, the performance of the proposed approach is evaluated in two scenarios (with full features and a reduced feature set). Also, a comparative result analysis between the proposed DLFFNN-KMC-IG and benchmark machine learning approach is represented. The proposed method shows an accuracy of 97.9%, 95.3%, and 96.9% for the entire dataset, while 98.9%, 98.2%, and 98.8% for reduced features set for NSL-KDD, UNSW-NB15, and CICIDS2017 dataset respectively for binary classification. For NSL-KDD, UNSW-NB 15, and CICIDS2017, it shows an accuracy of 91.3%, 96.8%, and 96.9 % for the whole dataset, while 93.8%, 97.5%, and 97.3 % for reduced features set in case of multi-classification. The obtained result demonstrates that the proposed model outperforms over conventional machine learning approach for each dataset in both scenarios. With the addition of the KMC-IG-based feature reduction approach, the DLFFNN model is more resilient and has a higher generalization performance than conventional techniques but also saves training and testing time for processing large-scale datasets.

427

Key Words: Intrusion Detection, Feature Extraction, Network Attacks, Feed Forward Neural Network, Deep CNN.

DOI Number: 10.14704/nq.2022.20.8.NQ44049

NeuroQuantology 2022; 20(8):427-444

Introduction

With the advancement in Internet technology and availability, the network is becoming easier and more accessible. Since every technology has its pros and cons, same with network accessibility. Network access is prone to malicious activity, which can be suppressed by Network Intrusion Detection Systems (NIDS). Many service providers,

companies, and even individual users are now using Network Intrusion Detection Systems (NIDS) to make their networks more secure. Detecting malicious behaviour, monitoring network traffic, and alerting system security operators or stopping suspicious network connections are the functions of NIDS.

Corresponding author: Geeta Kocher

Address: ^{1*}Research Scholar, Department of Computational Sciences, MRSPTU, Bathinda, Punjab, India; ²Associate Professor, Department of Computer Applications, SBSSU, Ferozpur, Punjab, India.

E-mail:

^{1*}geetakocher76@gmail.com,

²gulshanahuja@gmail.com



NIDS for anomaly and abuse detection may be divided into two categories depending on its detecting method (Shone et al., 2018). The former sets a reference point for typical activity and monitors for departures from it. On the other hand, the latter keeps monitoring activities and describes known harmful actions in great detail.

Regarding real-world deployments, NIDS systems that identify misuse are almost exclusively signature-based NIDS that analyze network traffic for specific byte sequences. Once deployed, NIDS become targets of attacks aimed at compromising their functioning. These cyber-attacks remain one of the most significant threats in the field of cyber security (Farnaaz and Jabbar, 2016).

Using network evasion methods, attackers may now distribute and exploit code without triggering a misuse alert from an intrusion detection system. This is because the resilience theory in network protocol design dictates that a protocol execution should transmit correctly formatted data while also accepting correctly formatted data from other protocol implementations. Attackers may exploit these uncertainties to trick NIDS and endpoint systems into thinking they have been attacked to craft network traffic. When packet processing produces different representations of raw data in the NIDS and the end systems, an attack may go undetected to the intended target without detection. These privacy-invading tactics are evasion techniques (Khammassi and Krichen, 2017).

The anomaly-based detection technique, a significant research area in intrusion detection, identifies normal and suspicious patterns from network data and may identify unknown and novel threats by monitoring data traffic over the network. However, feature design dramatically impacts system performance, and developing a feature set correctly describes network traffic is still a work in progress. In addition, anomaly-based IDS have a significant false alarm rate (FAR), which significantly restricts their use (Kavitha et al., 2012).

Since Hinton initially suggested deep models in 2006, they have received industrial and academic attention due to their outstanding learning abilities. Deep learning, a subfield of machine learning, allows for the autonomous learning of effective feature representations from input data, resulting in improved classification accuracy (Shiravi et al., 2012). Computer vision, health, neurology, automobile design, and manufacturing are just a

few fields where deep learning has produced positive results. Multiple hidden layers are included in deep-learning models. Each hidden layer can obtain a compact data representation automatically, reducing the need to build link characteristics manually. (Zhang et al., 2015).

NIDS become targets of assaults as soon as they are installed, seeking to degrade their functionality. These attacks remain one of the most severe risks in the cyber security area. Attackers can now transfer exploit code to victims without being detected by a misuse-based NIDS, thanks to the development of network evasion techniques. Because of the robustness principle in the internet protocol design, which states that a protocol implementation should send well-formed datagrams but accept any datagrams that it can interpret, different protocol implementations have varied interpretations. Attackers can use these uncertainties to design network traffic so that NIDS and endpoint systems process packets differently. An attack can reach the destination unnoticed if the packet processing generates distinct representations of the raw data in the NIDS and the end systems. Evasion techniques are a collective term for these privacy methods (Jaiganesh et al., 2014).

This paper analyses the complexity of three datasets by replicating the attributes of UNSW-NB15, NSL-KDD, and CICIDS2017. The KMC-IG is proposed as a feature extraction technique to extract optimal features from the datasets. Furthermore, DLFFNN is applied to analyze the performance based on the accuracy metric for the given datasets.

This research work makes the following contributions:

- SMOTE-based ENN algorithm is used to perform the oversampling and cleaning of data from the dataset and produce balanced data for further process.
- K-means Clustering-based Information Gain (KMC-IG) method is developed to extract optimal features from the datasets such as UNSW-NB15, NSL-KDD, and CICIDS2017.
- A Deep Learning-based Feed Forward Neural Network (DLFFNN) algorithm is proposed to analyze models' accuracy, using the extracted optimal features from the dataset.

The rest of this research work is summarized



below. Segment II elaborates on the existing methods and related background studies of NIDS. Segment III describes the proposed feature extraction and deep learning-based classification model and analyzes it with three different datasets. Segment IV elaborates on the proposed model's experimental outcomes and performance evaluation. Segment V describes the conclusion of the paper.

Related Works

Initial network intrusion detection techniques such as anomaly and misuse detection were included. However, these detection methods have attained excellent outcomes. Still, inbuilt flaws exist that misuse detection failed to determine whether the unknown behavior can be safer. It produces a high false alarm rate for abnormal detection, a disadvantage of misuse detection. With the increase in the development of artificial intelligence-based models, deep learning and machine learning-based NIDS has improved the performance in identifying vulnerable attacks, becoming a new research hotspot. The authors apply the deep neural network for detecting intrusion behaviours using the KDD99 dataset (Ma et al., 2016). The intrusion detection methods are studied by Niyaz et al. in which authors used the NSL-KDD dataset to analyze attacks using the deep belief networks (DBN)(Niyaz et al., 2016). The authors designed the model to learn features manually from the traffic feature. Zhang et al. used genetic algorithms for finding the optimal network structure based on DBN and analyzed intrusion detection. However, the model validity is only concentrated by the authors, and more experiment based on the performance is not done for the test set (Zhang et al., 2013). The authors applied the DBN model for the training samples that were used as Portable Executable (PE) files for the detection of malware, and the performance achieved by the model is higher (Ding and Wang., 2017).

Wang et al. learned the spatial features using the CNN model for analyzing the network traffic and detected the malware traffic classification with the help of the image-classification method (Wang et al., 2017). The authors combined an artificial neural network (ANN) and fuzzy clustering algorithm to solve the low-frequency attacks' low detection rate (DR) problem and increased the DR and more robust stability of IDS. Bamakan et al. selected the parameters adaptively for the support vector machine by introducing particle swarm

optimization based on time-varying chaos, which helps improve higher DR (Bamakan et al., 2015).

The authors used least square SVM to combine flexible, mutual information for feature selection. The model contributes to faster detection speed and low power consumption (Ambusaidi et al., 2016).

Osanaiye et al. achieved an optimal selection of features using the combination of ensemble-based feature selection method and output four-filter methods. They showed higher performance in the detection of DDoS attacks. However, the authors used only the manually designed features for the analysis for targeting the IDS, and redundant features have been removed, resulting in a loss of information (Osanaiye et al., 2016).

Kumar et al. used Random Forest (RF) and Binary Particle-Swarm Optimization (BPSO) algorithms to classify Probe attacks. The authors proved that the forest reduces the false positive rate as the number of trees increases when classifying the attacks. The large volume of data is predicted by finding suitable patterns with the successful collaborative random forest and filtering technique (Kumar et al., 2010). Hu et al. used a Support Vector Machine (SVM) to develop an approach consisting of multiple classes and merging feature selection in the NSL-KDD dataset for intrusion detection. The authors achieved more than 90% accuracy using their proposed model, and only three input features were used for the classification (Hu et al., 2009). Mostafa and Slay introduced the UNSW-NB15 dataset in 2015, which has different features than the KDD 99 dataset. The UNSW-NB15 shares a few features present in the KDD-99 dataset and these datasets are helpful in proposing an effective intrusion detection system (Mostafa and Slay, 2015). The authors developed self-taught learning (STL) based intrusion detection system (IDS) with deep learning (DL) approach. The NSL-KDD dataset was used to analyze the model's performance based on the performance metric known to be accuracy (Tang et al., 2014).

Yin et al. developed IDS based on the DL algorithm built using the Recurrent Neural Network (RNN) (Yin et al., 2017). The method is applied to the NSL-KDD dataset for analyzing the performance of the deep learning model, and experimental outcomes outperform when compared with existing machine learning methods. The authors introduced DBN based deep learning approach with RBM, in which the other four layers are used for feature reduction. In a fine-tuning phase, the DBN's



weights were updated, and the classification task was completed using a Logistic Regression (LR) classifier. The proposed method was tested on the KDD-99 dataset and yielded a 97.9% accuracy rate with a 0.5 percent false alarm rate (Alrawashdeh and Purdy, 2016).

Proposed Methodology

IDS serves a vital role in ensuring the security of networked computer systems; nevertheless, performance remains a significant problem for different IDS. The high dimensional feature space is one of the major issues which, when combined with a considerable quantity of noisy data, reduces the accuracy and efficiency of IDS. Furthermore, when the feature space expands, the accuracy of current machine learning-based IDS methods suffers

significantly. In this paper, an intrusion detection model based on a deep learning algorithm using concepts of feed-forward deep neural networks (DLFFNN) is presented for classification and K-means cluster with Information gain for feature extraction and selection method. In order to improve the performance of DLFFNN, SMOTE and ENN are also used for data balancing and cleaning. For the effective and efficient IDS, the proposed framework is divided into five stages: Knowledge discovery, Data preprocessing, data modelling, building model, and Model Evaluation. Each of these components plays an important function and has a significant consequence on the performance of the IDS model. Figure 1 shows the architecture of the proposed work for developing IDS.

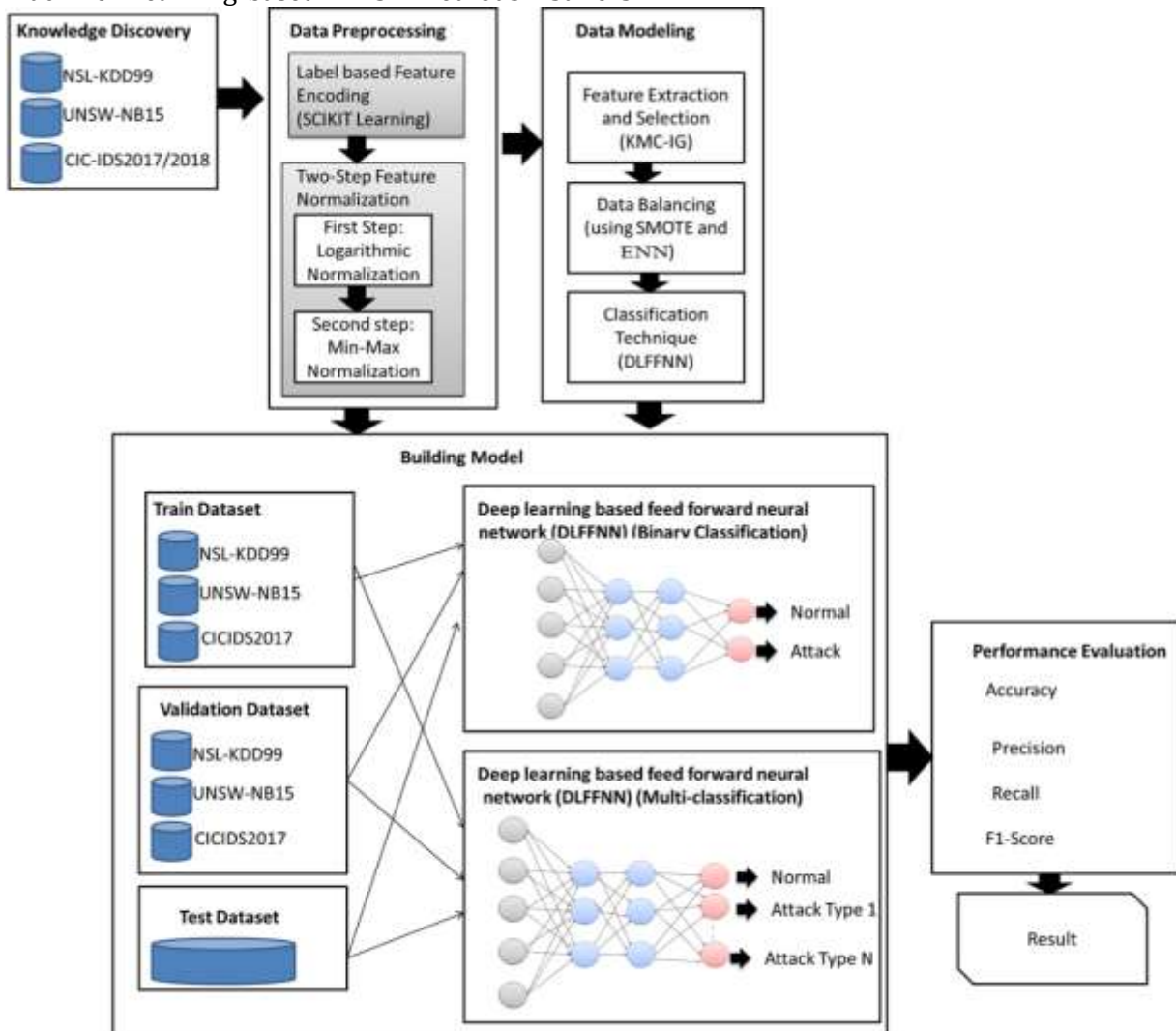


Fig. 1. Proposed architecture workflow

1. Knowledge Discovery

The selection of data is critical since the proper

dataset determines the reliability of the model assessment. The model's accuracy improves only



when assessed with more accurate and reliable data. Unfortunately, due to the security and privacy concerns of organizations, actual traffic data of the network is not accessible. For dataset gathering, many techniques are used, including simulation datasets, sanitized or cleaning datasets, testbed datasets, and standardized benchmark datasets. However, all these datasets except standardized benchmark datasets are complex and challenging. For instance, information production for traffic in the simulation dataset is complicated, and the testbed dataset is time-consuming and expensive, while the sanitized dataset is unsafe to use. As a result, several researchers utilize publicly accessible standardized datasets to analyze the IDS. However, the conventional dataset has insufficient quantity, a shortage of traffic flow diversity, and a variety of current traffic attack methods. As a result, in order to accomplish a fair and reasonable performance assessment, this work employs one historical dataset, NSL-KDD (Tavallae et al., 2009), as well as two newly published standard datasets, CICIDS2017 (Sharafaldin et al., 2018) and UNSW-NB15 (Sonule et al., 2020). The overview and description of these datasets are discussed in section 4.1. However, their statistical information is given in Table 1.

Table 1. Statistical Information of Standard Dataset

Dataset Name	# of features and attributes	# of instances	# of attack types
NSL-KDD	41 features	8,05,049	Four (4)
CICIDS2017	78 features	2,30,092	Fourteen (14)
UNSW-NB15	43 features	2,57,673	Nine (9)

2. Data Preprocessing

Data preprocessing is a critical step in improving the detection abilities and efficiency of the IDS model. In the proposed paradigm, data preprocessing consists of two major tasks:

Label-based Feature Encoding: The technique of translating non-numeric (symbol or string) characteristics to numeric values is known as feature encoding. In intrusion detection, datasets often comprise discrete, symbolic and continuous characteristics. Most deep learning algorithms are intended to work exclusively with numeric values and, therefore, cannot get along with features with symbolic characteristics. As a result, an encoding

method is required to translate all symbolic characteristics into numeric.

Label encoding, as well as One Hot encoding, are the two most popular methods. The Label encoder converts a symbol-based feature into C classes by mapping each symbolic characteristic feature into a numerical value between 0 to C – 1. On the other hand, one Hot encoder converts a symbol-based feature with C classes by generating C pointer variables C in which 1 represents the presence of the relevant class, and 0 shows the absence of another class. As a result of the high dimensionality, these pointer variables generated for each class significantly increase the complexity of the dataset and impact the performance of deep learning algorithms. Therefore, Scikit' 1'learn' 1'based' 1'label' 1'encoding' is used.

Feature Normalization: Normalization keeps values within the same range for optimum processing. A two-step normalization method is used in this paper. The logarithm standardization is performed first, as stated in equation (1), to make all the features within an accepted range and then proportionally limit the values in the range of [0,5] using equation (2).

$$fr_{norm} = \log(fr_i + 1) \quad (1)$$

$$fr_{norm} = (q - p) \frac{fr_i - \min(fr_i)}{\max(fr_i) - \min(fr_i)} \quad (2)$$

Where p=0 and q=5. Algorithm 1 represents the feature encoding and normalization process in a stepwise manner.

Algorithm 1: Feature Encoding and Normalization

Input; Dataset DS with feature Fr(fr_1, fr_2, \dots, fr_n)

Output: $Fr_{Normalised}(fr_1^{norm}, \dots, fr_n^{norm})$

1. for i=1 to n do
 - a. if (feature (fr_i) is symbolic) then
 - i. Map symbolic feature into numeric using mapping procedure SCKIKIT Learn
 - ii. Execute logarithmic normalization using eq (1)
 - iii. Execute min-max normalization using eq. (2)
 - endif
 - b. Execute logarithmic normalization using eq (1)
Execute min-max normalization using eq. (2)
2. end for

3. Data Modeling



Data modelling is an essential stage in data analysis that involves the reduction of high dimensional feature space by choosing the best and most relevant feature and improving the intrusion detection accuracy by selecting the best classification techniques. In addition to feature selection and intrusion detection using classification techniques, this work focuses on incurring the data imbalance issue that significantly contributes to achieving higher accuracy. In order to accomplish all these tasks, data modelling is further categorized into three steps: Feature extraction and selection (FES), Data Balancing and Classification.

▪ **Feature Extraction and Selection (FES)**

It is the process of extracting features and selecting the most relevant feature set by removing redundancy and duplicity. This step enhances the performance and accuracy and minimizes data collection costs and time. In this work, NSL-KDD, UNSW-NB 15 and CICIDS2017 are used, which have a high dimensionality feature set including 41 features, 43 features, and 78 features, respectively. In the context of network analysis, computation and identification of suspicious attack-based network feature characteristics are complex and expensive. This creates room for determining the most helpful feature characteristics which can detect each malicious type of attack in developing successful IDS. In this work, a DLFFNN model is proposed to address this issue which uses clustering and entropy-based information gain concept for FES. The data mining-based K-means clustering technique is also proposed, followed by the information gain feature ranking approach for feature extraction and selection.

KMC-IG-based FES: Feature extraction is performed using K-means clustering in which clusters of datasets are formed based on the classification category. In binary classification, network traffic data are categorized into two categories: Normal and Anomalous. So, for binary classification, the dataset is clustered into two groups (K=2). While in multi-class classification, clusters are formed based on the number of attack types (K=# of attack types). After clustering using K-means, each extracted feature is selected using an entropy-based information gain feature ranking approach. The information gain (IG) feature ranking approach is applied to each cluster that calculates the scores of each feature in the clusters. High score rankings are selected as it helps in

improving classification accuracy, while lower ranking features are discarded. The IG of each feature is calculated w.r.t to every cluster class using the following equation:

$$IG(F_x|F_y) = E(F_x) - CE(F_x|F_y) \quad (3)$$

Where x and y are two random variables, $E(x)$ and $CE(x|y)$ represent entropy and condition entropy for measuring uncertainty of variables and is computed using the following:

$$E(F_x) = - \sum_{x \in F_x} Prb(x) \log_2(x) \quad (4)$$

$$CE(F_x|F_y) = - \sum_{x \in F_x} Prb(x) \sum_{y \in F_y} Prb(x|y) \log_2(Prb(x|y)) \quad (5)$$

Where Prb represents the probability, the feature with a strong correlation is selected using the information gain. Therefore, if $IG(F_x|F_y) > IG(F_v|F_y)$ then it is deduced that feature F_y is strongly correlated to feature F_x than feature F_v .

Algorithm 2: Feature Extraction and Selection Unit

Input: Normalized feature $Fr_{Normalised}(fr_1^{norm}, \dots, fr_n^{norm})$ and attack type $(1, 2, \dots, k-1, k)$ for every record

Output: Reduced feature set with feature ranking

1. Begin
2. for $i=1$ to n do // make the cluster based on attack type $k=No.$ of attack using K-means clustering
 - $C_s = Kmeans(DS[fr_i], k)$
 - $IG_i\{F_x|F_y\} \leftarrow Information\ Gain(C_s)$
 - Compute Information Gain IFG_i using equations (3),(4), and (5)
 - if $(IG_i \geq IG_{threshold})$ then
 - load IG_i into Fr_{ranked}
 - end if
- end for

▪ **Data Balancing**

When the unbalanced NSL-KDD, CICIDS2017, and UNSWNB15 datasets are utilized, the classification method favours the majority class. To address this problem, the under-sampling and oversampling methods are utilized to balance the datasets. To balance the NSL-KDD, CICIDS2017, and UNSWNB15 datasets, SMOTE and ENN are used in the proposed model. In this work, SMOTE achieves oversampling of datasets by balancing minority classes. The SMOTE is used for oversampling, while ENN achieves data cleaning and noise removal.

Data balancing using SMOTE and ENN: To balance the dataset using SMOTE, SMOTE



technique is applied to M set on the minority instance. For each fx_i an instance of M set, n number of synthetic instances are created using the following equation:

$$fx_{syn} = fx_i + (\widehat{fx}_i - fx_i) \cdot \gamma \quad (6)$$

Where, \widehat{fx}_i is randomly selected instance nearest neighbour to instances fx_i which is calculated using the k-nearest neighbours (KNN) approach, γ is the random variable between [0,1].

This step is iterated n number of times for each fx_i an instance of M set, where $n = \frac{\beta}{100}$, and β oversampling percentage required dataset balancing.

In ENN, all the instances of the majority class are eliminated whose KNN prediction value is different from the values of the majority class. Let fx_i be the instance such that $fx_i \in N$ (where N is the total number of instances in the dataset) and has neighbours from different classes are higher, this fx_i instance will be discarded. The working steps of ENN are explained below:

The ENN works according to the steps below:

1. Compute K closest neighbour of fx_i using KNN such that $fx_i \in N$
2. The instance fx_i will be discarded if the count of its nearest neighbours from another class is higher;
3. Repeat this process for every majority class instance such that each instance is a subset of N.

The ENN approach eliminates noisy and borderline instances, resulting in a better decision model.

Classification Technique

When the feature space expands, current machine learning-based IDS methods' accuracy suffers significantly. In this paper, a deep learning-based IDS using DLFFNN is presented.

DLFFNN for Intrusion Detection

Deep neural networks (DNNs) have become the method of choice for solving complex problems for various applications. Artificial neurons (AN) are the foundation of a DNN and are based on the brain's organic neurons. The AN determines and transmits the data summed at its input. Each AN employs an activation function for each output to improve approximation and learnability. This is done to make the model more reflective of the actual world, which does not follow a linear path. The activation

function can be sigmoid (σ_{sig}), Rectified Linear Unit (ReLU) ($Rf(x)$) and hyperbolic tangent ($\tanh(x)$). The mathematical form of each activation function is expressed using the following equation:

$$\sigma_{sig} = \frac{1}{1 + e^{-x}} \quad (7)$$

$$Rf(x) = \max(0, x) \quad (8)$$

$$\tanh(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \quad (9)$$

Figure 2 demonstrates deep learning-based feed-forward neural network architecture.

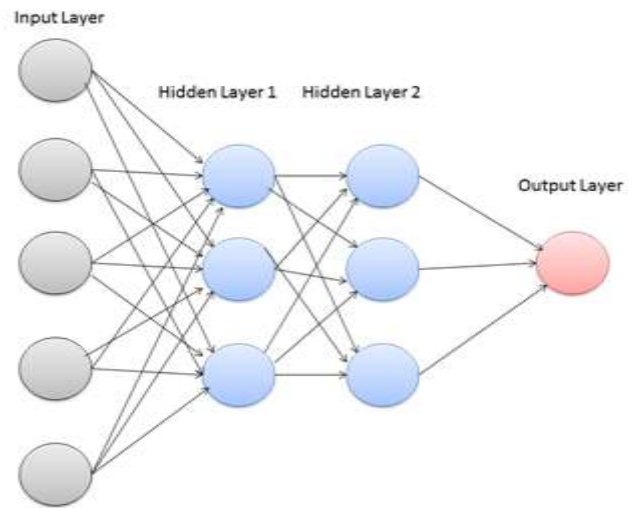


Fig. 2. Architecture of deep learning-based feed-forward neural network (DLFFNN)

The training algorithm for the DLFFNN is described in Algorithm 3. The following three main stages are involved in training DLFFNN:

1. The forward transmission of information.
2. The calculated error is back-propagated.
3. Bringing the weights and biases up to date.

Given an s-sample instances of training set, $\{(fx_1, fy_1), \dots, (fx_s, fy_s)\}$ and λ learning-rate. The DLFFNN is trained using back - propagation learning method, which is followed by a stochastic gradient descent (SDG) approach for updating the weights (Wt) and biases (bs). Furthermore, the gap between the desired and the actual output is computed using cost function defined by following equation:

$$Cost(Wt, bs; m, n) = \frac{1}{2} \|n - op\|^2 \quad (10)$$

Algorithm 3: Deep Learning-based Feed-forward Neural Network (DLFFNN)



Input: Weight (Wt), Bias (bs), Feature Set fr_i (Full feature set $Fr_{Normalised}$ or Reduced Feature set Fr_{ranked})

Output: Updated Weight (Wt), Bias (bs)

1. Forward and Propagate Feature set fr_i through N input layers $\mathcal{L} = \ell_1, \ell_2, \dots, \ell_N$ using $\mathfrak{S}^{\mathcal{L}+1} = Wt^{\mathcal{L}}\alpha^{\mathcal{L}} + bs^{\mathcal{L}}$ and $\alpha^{\mathcal{L}+1} = \mathcal{F}(\mathfrak{S}^{\mathcal{L}+1})$ where \mathcal{F} is ReLU calculated as $\mathcal{F}(\mathfrak{S}) = \max(0, \mathfrak{S})$
2. For every i^{th} output unit error term \mathcal{E} is computed using the following equation:

$$\mathcal{E}_i^N = \frac{d}{d(\mathfrak{S}_i^N)} \frac{1}{2} \|n - op\|^2 = -(n_i - \alpha_i^N) \cdot \mathcal{F}'(\mathfrak{S}_i^N)$$

3. Compute error term for every i^{th} node in every hidden unit present in layer $\mathcal{L} = N - 1, N - 2, \dots, 2, 1$ using following equation:

$$\mathcal{E}_i^{\mathcal{L}} = \sum_{j=1}^{N+1} Wt_{ji}^{\mathcal{L}} \mathcal{E}_j^{\mathcal{L}+1} \cdot \mathcal{F}'(\mathfrak{S}_i^{\mathcal{L}})$$

4. For every training sample computes partial derivative w.r.t. weight (Wt) and bias (bs) using the following equation:

$$\frac{d}{dWt_{ij}^{\mathcal{L}}} Cost(Wt, bs; m, n) = \alpha_j^{\mathcal{L}} \mathcal{E}_i^{\mathcal{L}+1}$$

$$\frac{d}{dbs_i^{\mathcal{L}}} Cost(Wt, bs; m, n) = \mathcal{E}_i^{\mathcal{L}+1}$$

5. Finally, weight (Wt) and bias (bs) are updated using the following equation:

$$Wt_{ij}^{\mathcal{L}} = Wt_{ij}^{\mathcal{L}} - \lambda \alpha_j^{\mathcal{L}} \mathcal{E}_i^{\mathcal{L}+1}$$

$$bs_i^{\mathcal{L}} = bs_i^{\mathcal{L}} - \lambda \mathcal{E}_i^{\mathcal{L}+1}$$

Where λ is the learning rate [0, 1]

4. Building Model

The prominent role of an IDS is to find and categorize the network traffic pattern as normal or intrusive based on the significant feature selected in FES. This step comprises three steps; training, validation and testing. During the training step, the DLFFNN model is trained and optimizes the features using training datasets that include both normal and malicious traffic data and their associated target class labels. The performance of the proposed DLFFNN model is assessed during the validation phase. For performance assessment, the k-fold cross-validation method is used. In this method, a dataset is arbitrarily split into k equal portions. At every iteration, one portion is chosen for data validation, while k-1 portion is used as the training dataset. In this work, k=10 is set as this value provides minimal variance, low bias, minimum overfitting and excellent error approximation. In the testing step, the DLFFNN model is permitted to identify target class labels

from traffic data available in the test sample data, and its performance is assessed.

5. Model Evaluation

IDS plays a vital role in ensuring the privacy and security concerns of the network system. The evaluation of IDS performance is based on the correct identification and classification of network flow traffic into normal or abnormal. Also, for accurate IDS, identifying threats is not enough; it should have a minimum or no false alarm rate. Therefore, four evaluation parameters, including accuracy (ACY), recall (RE), precision (PRE), and F1-Score (FS), are employed for the performance evaluation of the proposed DLFFNN model. The data may be split into four groups after classification: Correct Positive (CP), Incorrect Positive (IP), Correct Negative (CN), and Incorrect Negative (IN) (FN). The following equation represents the formula for four evaluation metric parameters:

$$Accuracy(ACY) = \frac{CP + CN}{CP + CN + IP + IN} \quad (11)$$

$$Recall(RE) = \frac{CP}{CP + IN} \quad (12)$$

$$Precision(PRE) = \frac{CP}{CP + IP} \quad (13)$$

$$F1 - Score(FS) = 2 \times \frac{PRE \times RE}{PRE + RE} \quad (14)$$

Results and Evaluation

The evaluation of the proposed work for binary and multi-class classification is presented in this section. The three datasets are considered in this work, namely UNSW-NB15, NSL-KDD and CICIDS2017. The experimental testbed is executed with Python-based Spark on Amazon's Elastic Map Reduce (EMR) cloud services. A cluster framework is employed in which a cluster of 10 nodes is used for this testbed, and each node has **a configuration of** Intel Xeon E5 2676 v3 @ 2.4 GHz processor speed with the memory of 64 GB. MLib (Machine learning) library based on Apache Spark is used for implementing standard machine learning algorithms, and Keras9 library is used for implementing the proposed deep learning model. For the classification number of nodes at the input, layer is selected based on the entire feature set or reduced feature set of each dataset, and the output layer consist of two nodes for binary classification that classify the network traffic data in two categories as 'Normal or Attack', while K nodes in



multi-classification where the value is chosen based on number of attacks classified for each dataset. The hidden nodes at hidden layer are chosen based on the trial-and-error approach.

1. Dataset Description and Modeling

This section presents the details of three datasets, as well as feature set modeling based on KMC-IG feature reduction method. After application of KMC-IG, feature set of each dataset is reduced. The 16 features are selected from NSL-KDD, 13 from UNSW-NB15 and 39 from CICIDS2017. For proposed DLFFNN and KMC-IG techniques, performance evaluation, each dataset is split into three parts: training (70%), validation (15%) and testing (15%). The performance is evaluated on the basis of full and reduced dataset for both binary classification and multi-class classification. The details of each dataset, reduced feature set and data distribution for training, testing and validation phase is mentioned in the following subsection.

■ NSL-KDD Dataset

The Defense Advanced Research Projects Agency (DARPA) produced the first IDSs dataset in 1998, dubbed the DARPA 1998 dataset. The KDD99 dataset was developed as a result of the DARPA 1998 dataset. Thereafter, KDD99 dataset becomes most popular dataset used for IDSs. However, redundant instances in this dataset may highly impact over the performance of classification techniques by favoring normal instance and, as a result, impair their ability to identify anomalies. To address this issue, a new dataset named NSL-KDD had been built in which duplicate instances had been removed.

Table 2. NSL-KDD Reduced Feature set

NSL-KDD Reduced Feature set (16)			
protocol_type	serror_rate	same_srv_rate	dst_host_serror_rate
flag	srv_serror_rate	dst_host_count	dst_host_srv_serror_rate
logged_in	rerror_rate	dst_host_srv_count	dst_host_rerror_rate
count	srv_rerror_rate	dst_host_same_srv_rate	dst_host_srv_rerror_rate

Tavallae et al. first built and introduced this new NSL-KDD dataset in 2009. After cleaning and removing duplicate instances, the total records listed in NSL-KDD are 257673. Like KDD99, the NSL-KDD dataset also comprised 41 feature

attributes and network traffic data of each record in NSL-KDD are classified into normal and four specific attack types, including Probe, DoS, U2R and R2L. After processing the KDD99, NSL-KDD is still biased towards normal instances as it contains a minimum number of attack instances compared to normal instances. (Tavallae et al., 2009) The reduced feature set used in this work for NSL-KDD dataset is shown in Table 2. The distributions of the dataset for the complete feature set and reduced feature set are shown in Table 3.

Table 3. NSL-KDD data distribution for training, validation and testing phase

Attack Type	NSL-KDD-Full Feature Set			NSL-KDD-Reduced and Balanced Feature Set		
	Training Set	Validation Test	Testing Set	Training Set	Validation Test	Testing Set
N=Normal	53937	11558	11558	13498	3123	3123
D=DoS	37370	8008	8008	9480	3729	3729
P=Probe	9855	2112	2112	3066	875	875
R=R2L	2717	582	582	1617	354	354
U=U2R	83	18	18	56	9	9
Total	103962	22278	22278	27717	8090	8090

■ UNSW-NB15 Dataset

The Australian Centre developed this dataset for Cyber Security (ACCS) lead researcher in Australia in 2015. It includes contemporary and sophisticated data representations of cyber threats (Sonule et al., 2020). The dataset mixes everyday activities with synthetic current attack behaviour to replicate how the attack tactics are growing continuously. The UNSW-NB15 dataset describes nine attack class types, including Generic (G), Exploits (E), Fuzzers (F), DoS(D), Reconnaissance(R), Analysis(A), Backdoor(B), Shellcode(SC), and Worms(W). The dataset includes a diverse set of 49 feature characteristics derived from a payload and network header field to classify network traffic as benign or malicious accurately. The UNSW-NB15 dataset is provided in four CSV files, including Two million and 540,044 link records. The dataset was split into several parts and built to be customized by end users. After partitioning, configuring, and eliminating six features from the whole dataset, there are only 43 features left. The reduced feature used in this work for the UNSW-NB15 dataset is shown in Table 4.



The distributions of the dataset for the full feature set and reduced feature set are shown in Table 5.

dloss	dmean
sinpkt	ct_state_ttl
swin	ct_dst_ltm
stcpb	ct_src_dport_ltm
	is_sm_ips_ports

Table 4. UNSW-NB 15 Reduced Feature set

UNSW-NB 15 Reduced Feature Set	
Proto	dtcpb
dttl	dwin

Table 5. UNSW-NB 15 data distribution for training, validation and testing phase

Attack Type	UNSW-NB 15-Full Feature Set			UNSW-NB 15-Reduced Feature Set		
	Training Set	Validation Test	Testing Set	Training Set	Validation Test	Testing Set
N=Normal	65100	13950	13950	45570	9765	9765
F=Fuzzers	16972	3637	3637	11881	2546	2546
A=Analysis	1874	402	402	1312	281	281
B=Backdoors	1630	349	349	1141	245	245
D=DoS	11447	2453	2453	8013	1717	1717
E=Exploits	31168	6679	6679	21817	4675	4675
G=Generic	41210	8831	8831	28847	6181	6181
R=Reconnaissance	9791	2098	2098	6854	1469	1469
SC=Shell Code	1058	227	227	740	159	159
W=Worms	122	26	26	85	18	18
Total	180371	38651	38651	126260	27056	27056

▪ **CICIDS2017 Dataset**

When this dataset was published by Sharafaldin et al. in 2018 at the Canadian Institute for Cybersecurity, it fulfilled the 11 critical requirements for generating a reliable feature set. Like the ISCX dataset, this dataset includes real-world examples of legitimate and malignant network traffic. The attack patterns are represented by 78 different characteristics and 78 different labels. All network traffic from Monday through Friday (including recent attacks) was used to create this dataset. On Tuesday and Wednesday, just recent attacks were used. Brute Force, Botnet, Heartbleed and DDoS are only a few of the injection attacks, as are DoS, DDoS and Web Attack. The dataset, which contains network traffic in packet-

based and bidirectional stream formats, was generated in a 5-day simulation scenario. The study analyzed over 78 features for each flow and supplied additional information regarding IP addresses and intrusions. CICIDS-2017 offers a broader variety of attack methods than NSL-KDD and UNSW-NB15, including SSH brute force (BF), Heartbleed (HB), botnet(B), DoS(D), DDoS(D), web, and infiltration (I) attacks. In addition, with almost 3 million data points, CICIDS-2017 can assess IDS performance in large-scale situations (Sharafaldin et al., 2018). The reduced feature used in this work for CICIDS-2017 dataset is shown in Table 6. The distributions of dataset for full feature set and reduced feature set are shown in Table 7.

Table 6. CICIDS2017 Reduced Feature set

CICIDS2017 Reduced Feature Set (39)				
Destination_Port	Bwd_Packet_Length_Std	Packet_Length_Std	Idle_Min	FIN_Flag_Count
Flow_Duration	Fwd_IAT_Max	Packet_Length_Variance	Flow_Packets/s	SYN_Flag_Count
Fwd_Packet_Length_Max	Bwd_IAT_Std	Avg_Fwd_Segment_Size	Flow_IAT_Mean	PSH_Flag_Count
Fwd_Packet_Length_Min	Bwd_IAT_Max	Avg_Bwd_Segment_Size	Flow_IAT_Std	ACK_Flag_Count
Fwd_Packet_Length_Mean	Fwd_PSH_Flags	Init_Win_bytes_backward	Flow_IAT_Max	URG_Flag_Count
Bwd_Packet_Length_Max	Min_Packet_Length	Idle_Mean	Fwd_IAT_Total	Down/Up_Ratio
Bwd_Packet_Length_Min	Max_Packet_Length	Idle_Std	Fwd_IAT_Mean	Average_Packet_Size
Bwd_Packet_Length_Mean	Packet_Length_Mean	Idle_Max	Fwd_IAT_Std	

Table 7. CICIDS2017 data distribution for training, validation and testing phase

Attack Type	CICIDS2017-Full Feature Set			CICIDS2017-Reduced Feature Set		
	Training Set	Validation Test	Testing Set	Training Set	Validation Test	Testing Set
N=Normal	43093	9234	9234	24132	5171	5171
B=Bot	1376	295	295	771	165	165
BF=Brute Force	1055	226	226	591	127	127
DD=DDoS	40694	8720	8720	22789	4883	4883
DGE=DoS Golden-Eye	7205	1544	1544	4035	865	865
DH=DoS Hulk	7340	1573	1573	4111	881	881
DSHT=DoS SlowHttpTest	3849	825	825	2156	462	462
DS=DoS Slowloris	4057	869	869	2272	487	487



FP=FTP patator	5557	1191	1191	3112	667	667
HB=Heart Bleed	8	2	2	4	1	1
I=Infiltration	25	5	5	14	3	3
PS=PortScan	42206	9044	9044	23635	5065	5065
S=SQL	15	3	3	8	2	2
SP=SSH Patator	4128	885	885	2312	495	495
X=XSS	456	98	98	256	55	55
Total	161064	34514	34514	90196	19328	19328

2. Performance Evaluation

The performance of the proposed model is evaluated in two scenarios: Scenario1 represents the performance based on binary classification for the complete and reduced feature set, and scenario 2 represents the performance based on multi-class classification for full and reduced feature features set.

Anomalous. Algorithm 1 is used only to encode and normalize the feature set for full feature set. For a reduced feature set, Algorithm 2 is used after applying Algorithm1. After feature reduction, SMOTE and ENN are applied for balancing the dataset. Table 8, Table 9 and Table 10 represent the confusion matrix for NSL-KDD, UNSW-NB 15 and CICIDS2017.

Scenar io 1: Binary Classification with the Full and Reduced Feature Set

In Binary classification, the model is trained, tested and validated into two categories: Normal and

Table 8. Confusion Matrix for NSL-KDD

Phase	Feature Set					
	Full Feature Set			Reduced Feature Set		
Training	Class	Normal	Anomalous	Class	Normal	Anomalous
	Normal	57997	71	Normal	15598	74
	Anomalous	96	45798	Anomalous	59	11986
Validation	Normal	12551	92	Normal	5681	74
	Anomalous	59	9576	Anomalous	51	2284
Testing	Normal	11467	44	Normal	4187	107
	Anomalous	84	10683	Anomalous	92	3694

Table 9. Confusion Matrix for UNSW-NB 15

Phase	Feature Set					
	Full Feature Set			Reduced Feature Set		
Training	Class	Normal	Anomalous	Class	Normal	Anomalous
	Normal	100074	98	Normal	81041	81
	Anomalous	167	80032	Anomalous	43	45095
Validation	Normal	28762	59	Normal	20967	42
	Anomalous	87	9743	Anomalous	63	5984
Testing	Normal	26122	76	Normal	19865	62
	Anomalous	51	12402	Anomalous	73	7056

Table 10. Confusion Matrix for CICIDS2017

Phase	Feature Set					
	Full Feature Set			Reduced Feature Set		
Training	Class	Normal	Anomalous	Class	Normal	Anomalous
	Normal	94534	68	Normal	62848	94
	Anomalous	41	66421	Anomalous	59	27195
Validation	Normal	24742	43	Normal	12861	79



	Anomalous	81	9648	Anomalous	53	6335
Testing	Normal	23102	39	Normal	13752	57
	Anomalous	61	11312	Anomalous	92	5427

The confusion matrix of each dataset shows a reduction in the classification accuracy rate. For NSL-KDD, UNSW-NB 15 and CICIDS2017, it shows the accuracy of 96.3%, 97.8%, and 96.9% for the full dataset, while 97.8%, 98.1% and 99.3% for the reduced features set. Correct and incorrect classification rate is also deducing from confusion. From the result, it is evident that accuracy after feature reduction is increasing.

classification, four metric parameters, including accuracy (ACY), precision (PRE), Recall (RE), and F1-score (FS), are considered and evaluate the performance for NSL-KDD, UNSW-NB 15 and CICIDS2017. The result of the proposed DLFFNN and KMC-IG method is compared with the existing machine learning approach, including SVM, Naïve Bayes (NB), CNN and ANN. Figures 3-5 show the comparative results before and after feature reduction using the proposed DLFFNN-KMC-IG model and the existing machine learning algorithm.

3. Comparison with Existing Machine Learning Classifiers

For comparative analysis based on binary

Table 11. Comparative Analysis of NSL-KDD Dataset

Algorithms	NSL-KDD Dataset							
	Original Feature Set				Reduced Feature Set			
	Accuracy	Precision	Recall	F-measure	Accuracy	Precision	Recall	F-measure
NB	78.9	77.8	76.3	77.1	86.9	84.2	82.7	83.9
CNN	94.3	92.8	91.8	91.8	96.3	95.3	94.3	92.8
SVM	81.2	78.6	78.1	77.3	82.8	82.5	82.3	81.4
ANN	92.2	91.2	91.1	92.2	93.7	93.1	92.7	94.1
Proposed	97.9	94.9	94.2	95.9	98.9	98.9	96.8	97.8

438

Table 12. Comparative Analysis of CICIDS2017 Dataset

Algorithms	CICIDS2017 Dataset							
	Original Feature Set				Reduced Feature Set			
	Accuracy	Precision	Recall	F-measure	Accuracy	Precision	Recall	F-measure
NB	78.3	76.8	76.3	77.1	96.9	81.2	80.7	80.9
CNN	93.8	92.8	91.8	91.8	96.3	95.3	94.3	92.8
SVM	79.6	78.6	78.1	77.3	83.8	82.5	82.3	81.4
ANN	92.2	91.2	91.1	92.2	94.1	93.1	92.7	94.1
Proposed	96.9	95.9	94.2	95.9	98.8	97.9	96.8	97.8

Table 13. Comparative Analysis of UNSW-NB15 Dataset

Algorithms	UNSW-NB15 Dataset							
	Original Feature Set				Reduced Feature Set			
	Accuracy	Precision	Recall	F-measure	Accuracy	Precision	Recall	F-measure
NB	74.8	73.2	73.8	75.8	79.7	77.7	78.7	80.7



CNN	90.8	90.2	88.9	90.8	95.3	93.1	94.3	96.3
SVM	75.6	73.9	75.6	77.6	80.8	79.2	79.8	81.8
ANN	88.2	88.6	88.2	90.2	93.1	92.4	91.8	94.1
Proposed	95.3	94.8	93.5	95.3	98.2	97.8	97.6	99.1

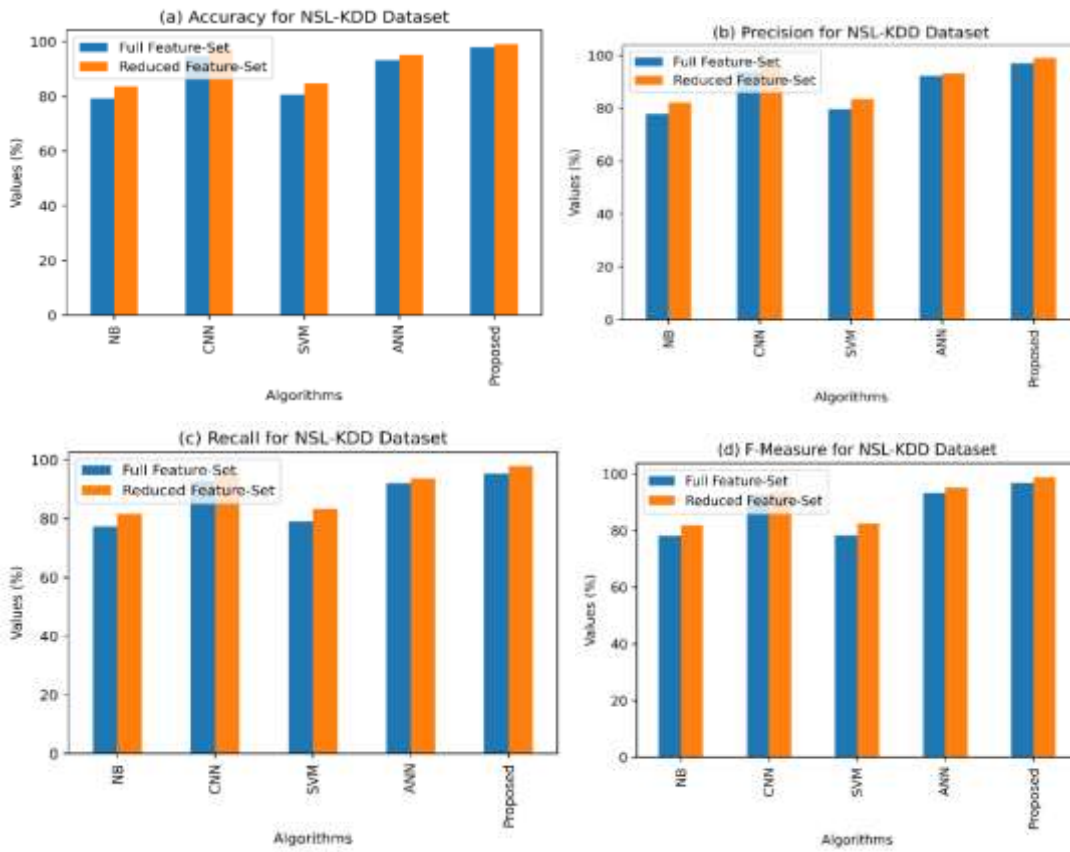
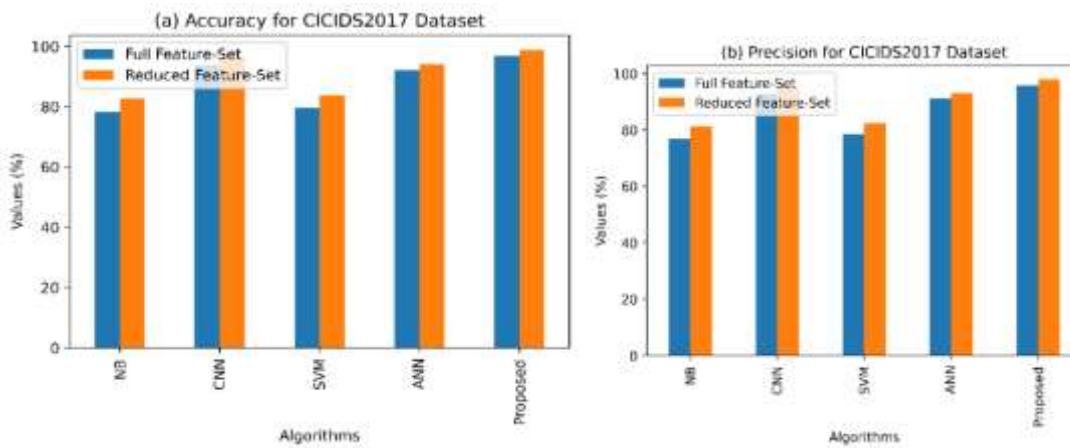


Fig. 3. Comparative Analysis of Proposed vs. Existing Methods for NSL-KDD Dataset



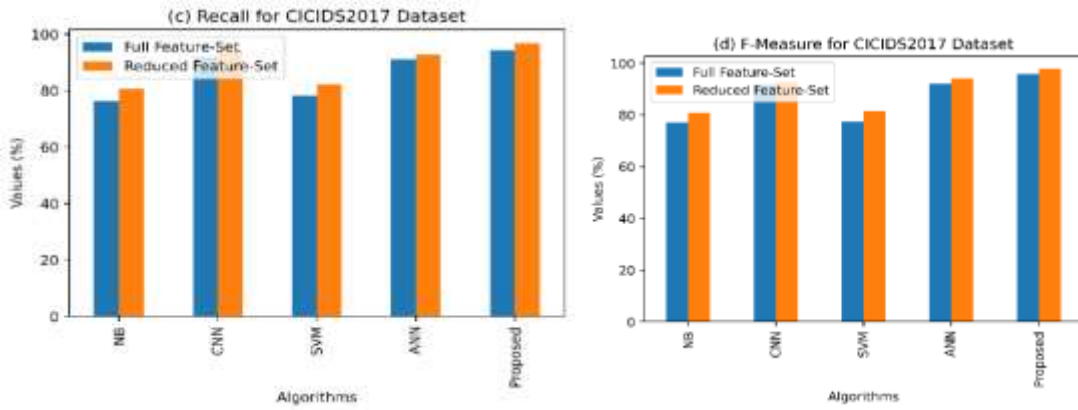


Fig. 4. Comparative Analysis of Proposed vs. Existing Methods for CICIDS2017 Dataset

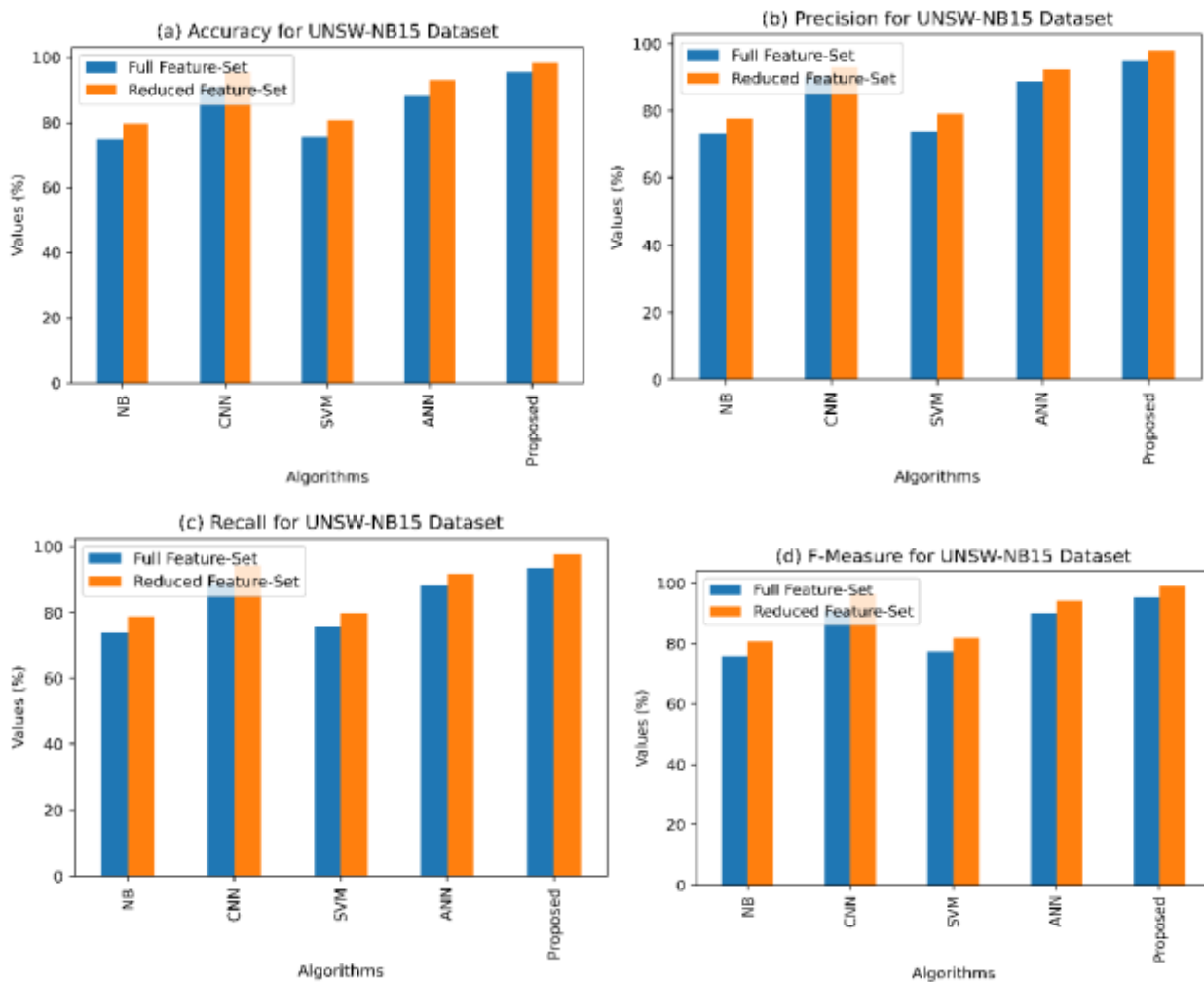


Fig. 5. Comparative Analysis of Proposed vs Existing Methods for UNSW-NB15 Dataset

The result demonstrated in Figures 3, 4, and 5 shows that the proposed DLFFNN-KMC-IG yields the highest Accuracy (ACY), Precision (PRE), Recall (RE), and F1-Score (FC). From the result, it is evident that when it comes to data fitting, proposed DLFFNN models can identify more complicated patterns and reveal the sample instances more accurately with hidden properties than

conventional machine learning models. Therefore, models with deep learning abilities are superior to those with shallow learning. The findings indicate that after using the KMC-IG feature reduction approach, the performance of the existing machine learning classifiers also improves to varying degrees. Based on the metric measures used in this work, it may be concluded that the proposed



method outperforms other conventional machine algorithms worldwide.

Scenario 2: Multi-class Classification with the Full and Reduced Feature Set

In multi-class classification, the model is trained, tested and validated in different attacks based on the dataset. NSL-KDD dataset is classified into four attack classes, and a normal class. UNSW-NB 15 dataset is classified into nine attack classes and a

normal class, while CICIDS2017 dataset is classified into 14 attack classes and a normal class. Like Binary classification, Algorithm 1 is applied to encode and normalize the feature set for full feature set. To reduce the feature set, algorithm 2 is used after the application of algorithm1. After feature reduction, SMOTE and ENN are applied for balancing the dataset. Table 14, Table 15 and Table 16 represent the confusion matrix for NSL-KDD, UNSW-NB 15 and CICIDS2017, respectively.



Table 14. Confusion Matrix for NSL-KDD

Phase	Feature Set											
	Full Feature Set						Reduced Feature Set					
	Class	N	D	P	R	U	Class	N	D	P	R	U
Training	N=Normal	68182	0	0	0	1	N	8756	0	0	0	1
	D=DoS	12	2703	2	8	5	D	0	5343	2	1	1
	P=Probe	0	0	1634	1	0	P	0	0	2877	1	0
	R=R2L	0	0	0	1927	0	R	2	0	0	4128	0
	U=U2R	5	3	0	1	6517	U	0	3	0	1	3929
Validation	N=Normal	9933	1	0	0	1	N	924	0	0	0	1
	D=DoS	12	2711	2	5	0	D	2	723	2	1	4
	P=Probe	0	0	4933	1	0	P	0	0	301	1	0
	R=R2L	0	0	0	3402	0	R	0	0	0	257	0
	U=U2R	3	1	0	1	861	U	1	0	0	1	543
Testing	N=Normal	9853	3	0	0	1	N	911	0	0	0	1
	D=DoS	13	3768	2	1	0	D	2	648	1	2	0
	P=Probe	0	0	3478	1	0	P	1	0	422	1	0
	R=R2L	2	0	0	4136	2	R	0	1	0	231	0
	U=U2R	0	1	0	1	855	U	0	3	0	1	517

Table 15. Confusion Matrix for UNSW-NB 15

Phase	Feature Set																					
	Full Feature Set											Reduced Feature Set										
	Class	N	F	A	B	D	E	G	R	SC	W	Class	N	F	A	B	D	E	G	R	SC	W
Training	N=Normal	61345	0	3	0	1	4	0	0	1	0	N	42342	0	0	0	1	0	0	5	1	0
	F=Fuzzers	1	43421	1	0	0	0	1	0	0	0	F	1	23415	0	1	0	1	1	0	0	4
	A=Analysis	0	2	24210	2	0	0	0	0	0	0	A	0	0	18723	2	0	0	0	0	0	0
	B=Backdoors	0	8	1	12034	0	3	0	1	0	0	B	0	1	1	16639	0	3	0	1	0	0
	D=DoS	0	0	3	0	11331	5	2	2	0	0	D	0	0	3	0	7238	0	2	0	3	0
	E=Exploits	6	8	0	2	7	9011	0	0	0	0	E	4	2	0	2	0	4562	0	0	0	2
	G=Generic	0	0	0	1	0	0	7712	0	0	0	G	0	0	0	1	0	0	3754	4	0	0
	R=Reconnaissance	0	4	0	0	2	1	6521	0	1	R	0	3	0	0	6	2	1	4312	2	1	
	SC=Shell Code	0	0	0	0	0	5	0	0	2964	0	SC	0	0	2	0	0	0	0	0	2964	0
	W=Worms	2	0	1	1	0	3	0	0	3	1734	W	1	0	2	0	3	3	1	1	3	2234
Validation	N=Normal	12321	0	4	0	1	2	0	0	1	0	N	9854	0	0	0	1	0	0	0	1	0
	F=Fuzzers	1	9034	1	0	0	0	1	0	0	2	F	0	6783	0	1	0	0	1	0	0	1
	A=Analysis	0	0	6567	2	0	0	0	0	3	0	A	0	0	4745	2	0	2	0	0	0	0
	B=Backdoors	1	3	1	4040	0	1	0	1	0	0	B	1	1	1	1456	0	0	0	1	0	3
	D=DoS	0	0	3	0	3145	3	2	2	1	0	D	0	0	0	0	1143	0	0	0	3	0
	E=Exploits	4	4	0	2	3	723	0	0	0	1	E	3	0	0	0	0	1091	2	1	0	1
	G=Generic	0	0	0	1	0	0	872	0	1	0	G	0	0	0	3	0	0	321	0	0	0
	R=Reconnaissance	0	1	0	0	0	0	1	645	0	1	R	0	0	0	1	0	0	0	576	2	2
	SC=Shell Code	0	0	0	0	0	6	0	0	283	0	SC	2	0	1	0	0	0	0	0	127	1
	W=Worms	2	0	1	1	0	1	0	1	0	954	W	0	1	5	0	1	0	2	1	2	910
Testing	N=Normal	11321	3	0	1	1	1	0	2	1	1	N	8746	1	2	0	1	0	0	0	3	2
	F=Fuzzers	3	13451	0	0	0	0	1	0	0	0	F	0	7533	0	1	0	0	1	2	0	1
	A=Analysis	0	0	6732	0	0	0	0	0	5	0	A	0	0	5745	0	0	1	0	0	0	0
	B=Backdoors	1	0	0	2145	2	1	0	1	0	0	B	1	1	1	1252	1	0	0	1	0	0
	D=DoS	0	0	2	0	1156	0	2	0	1	0	D	0	0	0	3	1876	3	0	0	1	0
	E=Exploits	0	1	0	1	3	434	0	0	0	1	E	1	4	0	0	0	593	2	1	0	1
	G=Generic	4	0	0	4	0	0	654	3	0	0	G	2	0	2	0	0	0	321	0	2	0
	R=Reconnaissance	0	1	0	0	5	0	1	1346	1	1	R	0	0	0	1	0	0	0	165	2	0
	SC=Shell Code	0	0	0	0	0	0	2	0	365	0	SC	0	2	1	0	0	0	0	0	127	3
	W=Worms	1	0	0	2	0	1	0	3	0	983	W	0	0	0	0	1	0	2	1	2	641

Table 16. Confusion Matrix for CICIDS2017



Phase	Feature Set																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							
	Full Feature Set													Reduced Feature Set																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
Class	N	B	BF	DD	DGE	DH	DSHT	DS	FP	HB	I	PS	S	SP	X	Class	N	B	BF	DD	DGE	DH	DSHT	DS	FP	HB	I	PS	S	SP	X																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
Training	N=Normal	53071	2	0	4	1	0	3	0	1	0	1	2	0	3	2	N	14625	4	0	1	1	0	1	0	4	6	0	0	1	0	4	B=Bot	1	12576	1	8	1	0	0	1	0	0	5	4	0	0	0	B	6	16587	0	0	1	0	0	1	0	0	0	0	4	0	0	BF=Brute Force	0	2	32456	2	0	4	0	5	0	1	0	1	0	1	0	0	BF	0	2	2658	0	0	4	3	5	9	0	0	0	0	3	0	DD=DDoS	0	6	2	16015	2	0	0	2	0	0	0	2	4	0	8	DD	0	6	2	6214	2	0	0	2	0	0	7	0	0	0	1	DGE=DoS Golden-Eye	4	0	0	3	11200	5	9	0	0	4	0	7	0	9	0	DGE	1	0	0	3	1507	5	2	0	7	2	0	7	0	0	0	DH=DoS Hulk	6	0	0	0	10	8350	0	0	4	0	4	0	2	0	0	DH	0	2	0	0	356	0	0	1	0	0	0	6	1	2	DSHT=DoS SlowHttptest	12	0	8	0	0	1	5327	0	15	1	6	0	0	0	0	DSHT	5	0	8	0	0	1	11264	0	0	0	3	7	0	0	3	DS=DoS Slowloris	0	2	0	0	0	2	0	4110	2	0	0	0	6	0	4	DS	0	0	0	0	2	4	0	3897	0	0	0	9	0	0	0	FP=FTP patator	0	5	0	9	6	6	0	8	9813	0	1	7	7	0	0	FP	0	0	0	9	0	6	0	9	4632	1	0	0	0	3	0	HB=Heart Bleed	0	1	0	0	0	8	0	1	2	1135	0	2	1	0	1	HB	0	1	0	0	0	0	0	3	0	297	2	0	3	0	2	I=Infiltration	3	0	4	0	0	0	5	0	4	0	0	1178	0	0	1	I	0	0	4	0	7	0	5	2	0	1	1687	0	0	2	0	PS=PortScan	7	1	0	1	0	1	1	0	7	1	3	2584	2	0	2	PS	3	0	0	1	0	1	1	1	0	3	0	2	3241	0	0	0	0	S=SQL	1	0	1	0	1	0	2	0	1	0	0	4	1999	3	0	S	0	0	1	0	0	2	5	1	7	1	0	9864	0	0	4	0	SP=SSH Patator	4	0	2	0	0	0	7	4	0	2	4	5	0	89	0	SP	0	4	2	0	9	0	3	1	0	0	0	0	4	13014	0	0	X=XSS	5	0	2	0	3	0	13	2	0	7	0	2	1	0	765	X	9	0	1	0	3	3	1	4	1	0	5	8	0	1	41				
	Validation	N=Normal	13062	4	0	1	1	0	1	0	4	6	0	0	1	0	4	N	8532	1	0	2	1	0	0	2	1	0	2	1	0	1	0	B=Bot	6	6576	0	0	1	0	0	1	0	0	0	0	4	0	0	B	0	2856	0	0	1	0	0	0	0	0	0	0	4	0	0	BF=Brute Force	0	2	1721	0	0	4	3	5	9	0	0	0	0	0	3	0	BF	1	1	451	0	0	2	3	0	2	0	0	0	0	3	2	DD=DDoS	0	6	2	1584	2	0	0	2	0	0	7	0	0	0	1	DD	0	0	0	378	2	1	2	0	0	1	1	0	2	0	0	DGE=DoS Golden-Eye	1	0	0	3	139	5	2	0	7	2	0	7	0	0	0	DGE	0	0	0	1	1354	0	0	0	3	0	0	0	0	0	0	0	DH=DoS Hulk	0	2	0	0	0	1287	0	0	1	0	0	6	1	2	DH	0	2	0	0	0	634	1	7	1	0	4	3	0	1	1	DSHT=DoS SlowHttptest	5	0	8	0	0	1	1932	0	0	0	3	7	0	0	3	DSHT	1	0	1	3	0	4	496	0	0	1	1	0	0	0	0	DS=DoS Slowloris	0	0	0	0	2	4	0	1712	0	0	0	9	0	0	0	DS	0	0	0	0	2	0	0	0	1236	1	0	0	1	3	0	0	FP=FTP patator	0	0	0	0	9	6	0	9	294	1	0	0	0	0	3	FP	7	0	3	0	0	0	5	0	458	2	0	0	0	0	4	HB=Heart Bleed	0	1	0	0	0	0	0	3	0	412	2	0	3	0	2	HB	0	1	0	3	0	2	0	3	0	635	5	0	3	5	2	I=Infiltration	0	0	4	0	7	0	5	2	0	1	521	0	0	0	2	I	0	3	0	0	1	1	0	0	1	0	1	996	3	0	2	0	PS=PortScan	3	0	0	1	0	1	1	0	3	0	2	763	0	0	0	PS	0	0	5	0	0	1	1	0	3	0	2	263	0	0	0	0	S=SQL	0	0	1	0	0	0	2	5	1	7	1	0	0	2146	0	S	0	5	1	2	0	0	0	0	0	1	0	1	0	315	0	0	SP=SSH Patator	0	4	2	0	9	3	1	0	0	0	0	0	0	4	97	0	SP	1	0	0	0	2	0	0	1	0	0	0	0	5	97	5	X=XSS	9	0	1	0	3	3	1	4	1	0	5	8	0	1	0	1956	X	0	0	0	1	0	1	4	0	1	2	1	5	0	0	437
		Testing	N=Normal	11678	1	0	4	0	2	0	3	4	1	0	7	0	2	0	N	7832	1	0	2	1	0	0	2	1	0	2	1	0	1	B=Bot	3	9521	3	0	3	0	5	0	0	0	0	0	0	0	0	B	0	1698	0	0	1	0	0	0	0	0	0	0	4	0	0	BF=Brute Force	0	2	2178	1	0	4	0	0	9	1	0	6	0	0	0	0	BF	1	1	1798	0	0	2	3	0	2	0	0	0	0	3	2	DD=DDoS	0	6	0	1584	1	6	1	2	0	0	2	0	5	0	3	DD	0	0	0	245	2	1	2	0	0	1	1	0	2	0	0	DGE=DoS Golden-Eye	1	0	0	0	1295	0	0	2	7	3	0	7	0	0	0	DGE	0	0	0	1	2194	0	0	0	3	0	0	0	0	0	0	DH=DoS Hulk	0	2	0	3	0	1521	3	0	1	0	0	1	2	2	DH	0	2	0	0	0	1365	1	7	1	0	4	3	0	1	1	DSHT=DoS SlowHttptest	1	0	1	0	0	1	714	6	0	3	5	2	0	0	0	DSHT	1	0	1	3	0	4	316	0	0	1	1	0	0	0	0	DS=DoS Slowloris	0	0	0	2	2	4	0	329	0	0	0	0	2	0	2	DS	0	0	0	0	2	0	0	0	263	1	0	0	1	3	0	0	FP=FTP patator	0	2	2	0	0	6	0	9	1874	5	1	0	0	6	0	FP	7	0	3	0	0	0	5	0	0	1322	2	0	0	0	0	4	HB=Heart Bleed	0	1	0	0	0	5	0	0	3	0	321	0	0	0	0	0	HB	0	1	0	3	0	2	0	3	0	256	5	0	3	5	2	I=Infiltration	3	0	4	0	0	0	5	2	0	1	1432	3	0	0	0	0	I	3	0	0	1	1	0	0	1	0	1	758	3	0	2	0	PS=PortScan	3	0	0	0	2	1	1	0	3	0	2	118	4	0	0	PS	0	0	5	0	0	1	1	0	3	0	2	729	0	0	0	0	S=SQL	0	0	1	8	0	0	2	5	1	7	1	0	1009	6	0	S	0	5	1	2	0	0	0	0	1	0	1	0	210	0	0	SP=SSH Patator	0	4	2	0	0	0	3	1	0	0	0	0	2	97	7	SP	1	0	0	0	2	0	0	1	0	0	0	0	5	87	5	X=XSS	3	1	1	1	4	3	1	2	1	0	4	1	0	0	0	541	X	0	0	0	1	0	1	4	0	1	2	1	5	0	0	65	

The classification accuracy rate is deduced from the confusion matrix of each dataset. For NSL-KDD, UNSW-NB 15 and CICIDS2017, it shows the accuracy of 91.3%, 96.8%, and 96.9 % for full dataset, while 93.8%, 97.5% and 97.3 % for the reduced features set. Correct and incorrect classification rate is also deducing from confusion matrix. From the result, it is evident that accuracy after feature reduction increases.

Conclusion

In this work, high dimensionality and the imbalanced nature of the datasets is considered in intrusion detection. For this purpose, a K-means clustering with information gain is proposed, followed by SMOTE and ENN oversampling technique that reduces dimensionality and balances the dataset. First, the work is concentrated on reducing feature dimensionality using KMC-IG, which uses a data mining-based K-means clustering algorithm that forms clusters based on binary and multi-classification categories. After clustering entropy-based information gain approach is used for ranking and reducing feature set. Secondly, SMOTE and ENN oversampling and data cleaning approaches improve intrusion detection. After dimensional reduction and balancing, a deep learning-based feed-forward

neural network is applied for intrusion detection and classification. The performance of the proposed KMC-IG and DLFFNN model is evaluated for the three datasets, including UNSW-NB15, NSL-KDD and CICIDS2017. The proposed method shows the accuracy of 97.9%, 95.3%, and 96.9% for full dataset, while 98.9%, 98.2% and 98.8% for the reduced features set for NSL-KDD, UNSW-NB15 and CICIDS2017 dataset respectively for binary classification. For NSL-KDD, UNSW-NB 15 and CICIDS2017, it shows the accuracy of 91.3%, 96.8%, and 96.9 % for full dataset, while 93.8%, 97.5% and 97.3 % for reduced features set in case of multi-class classification. The experimental results demonstrated that the proposed work improves intrusion detection's performance and outperforms the existing machine learning approach.

References

Alrawashdeh, K., & Purdy, C. (2016). Toward an online anomaly intrusion detection system based on deep learning. In 15th IEEE international conference on machine learning and applications (ICMLA), 195-200.
 Ambusaidi, M.A., He, X., Nanda, P., & Tan, Z. (2016). Building an intrusion detection system using a filter-based feature selection algorithm. IEEE transactions on computers, 65(10), 2986-2998.
 Bamakan, S.M.H., Amiri, B., Mirzabagheri, M., & Shi, Y. (2015). A new intrusion detection approach using PSO based



- multiple criteria linear programming. *Procedia Computer Science*, 55, 231-237.
- Ding, S., & Wang, G. (2017). Research on intrusion detection technology based on deep learning. In 3rd IEEE International Conference on Computer and Communications (ICCC), 1474-1478.
- Farnaaz, N., & Jabbar, M.A. (2016). Random forest modeling for network intrusion detection system. *Procedia Computer Science*, 89, 213-217.
- Hu, J., Yu, X., Qiu, D., & Chen, H.H. (2009). A simple and efficient hidden Markov model scheme for host-based anomaly intrusion detection. *IEEE network*, 23(1), 42-47.
- Jaiganesh, V., Mangayarkarasi, S., & Sumathi, P. (2014). An efficient algorithm for network intrusion detection system. *International Journal of Computer Applications*, 90(12).
- Javaid, A., Niyaz, Q., Sun, W., & Alam, M. (2016). A deep learning approach for network intrusion detection system. *Eai Endorsed Transactions on Security and Safety*, 3(9), e2.
- Kavitha, B., Karthikeyan, S., & Maybell, P.S. (2012). An ensemble design of intrusion detection system for handling uncertainty using Neutrosophic Logic Classifier. *Knowledge-Based Systems*, 28, 88-96.
- Khammassi, C., & Krichen, S. (2017). A GA-LR wrapper approach for feature selection in network intrusion detection. *Computers & security*, 70, 255-277.
- Kumar, G., Kumar, K., & Sachdeva, M. (2010). The use of artificial intelligence based techniques for intrusion detection: a review. *Artificial Intelligence Review*, 34(4), 369-387.
- Ma, T., Wang, F., Cheng, J., Yu, Y., & Chen, X. (2016). A hybrid spectral clustering and deep neural network ensemble algorithm for intrusion detection in sensor networks. *Sensors*, 16(10), 1701.
- Moustafa, N., & Slay, J. (2015). UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In *military communications and information systems conference (MilCIS)*, 1-6.
- Osanaiye, O., Cai, H., Choo, K.K.R., Dehghantanha, A., Xu, Z., & Dlodlo, M. (2016). Ensemble-based multi-filter feature selection method for DDoS detection in cloud computing. *EURASIP Journal on Wireless Communications and Networking*, 2016(1), 1-10.
- Sharafaldin, I., Lashkari, A.H., & Ghorbani, A.A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp*, 1, 108-116.
- Shiravi, A., Shiravi, H., Tavallaee, M., & Ghorbani, A.A. (2012). Toward developing a systematic approach to generate benchmark datasets for intrusion detection. *Computers & security*, 31(3), 357-374.
- Shone, N., Ngoc, T.N., Phai, V.D., & Shi, Q. (2018). A deep learning approach to network intrusion detection. *IEEE transactions on emerging topics in computational intelligence*, 2(1), 41-50.
- Sonule, A. R., Kalla, M., Jain, A., & Chouhan, D.S. (2020). UNSWNB15 Dataset and Machine Learning Based Intrusion Detection Systems. *International Journal of Engineering and Advanced Technology (IJEAT)*, 9(3), 2638-2648.
- Tang, P.S., Tang, X.L., Tao, Z.Y., & Li, J.P. (2014). Research on feature selection algorithm based on mutual information and genetic algorithm. In 11th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP), 403-406.
- Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A.A. (2009). A detailed analysis of the KDD CUP 99 data set. In *IEEE symposium on computational intelligence for security and defense applications*, 1-6.
- Wang, W., Zhu, M., Zeng, X., Ye, X., & Sheng, Y. (2017). Malware traffic classification using convolutional neural network for representation learning. In *International conference on information networking (ICOIN)*, 712-717.
- Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. *IEEE Access*, 5, 21954-21961.
- Zhang, F., & Wang, D. (2013). An effective feature selection approach for network intrusion detection. In *IEEE eighth international conference on networking, architecture and storage*, 307-311.
- Zhang, F., Chan, P. P., Biggio, B., Yeung, D. S., & Roli, F. (2015). Adversarial feature selection against evasion attacks. *IEEE transactions on cybernetics*, 46(3), 766-777.

