An Improved Intrusion Detection System for SDN using Multi-Stage Optimized Deep Forest Classifier

Saritha Reddy A [1†], **Ramasubba Reddy B**[2††], and **Suresh Babu A**[3†††]
sarithaanchuri@gmail.com rsreddyphd@gmail.com asureshjntu@gmail.com
Research Scholar, JNTUA, AP, India. Professor, SVEC, AP, India. Professor, JNTUACE, India

Abstract
Nowadays, research in deep learning leveraged automated computing and networking paradigm evidenced rapid contributions in terms of Software Defined Networking (SDN) and its diverse security applications while handling cybercrimes. SDN plays a vital role in sniffing information related to network usage in large-scale data centers that simultaneously support an improved algorithm design for automated detection of network intrusions. Despite its security protocols, SDN is considered contradictory towards DDoS attacks (Distributed Denial of Service). Several research studies developed machine learning-based network intrusion detection systems addressing detection and mitigation of DDoS attacks in SDN-based networks due to dynamic changes in various features and behavioral patterns. Addressing this problem, this research study focuses on effectively designing a multistage hybrid and intelligent deep learning classifier based on modified deep forest classification to detect DDoS attacks in SDN networks. Experimental results depict that the performance accuracy of the proposed classifier is improved when evaluated with standard parameters.

Key words:
Software Defined Networking (SDN), Network Intrusion Detection Systems (NIDS), DDoS attacks, Deep forest Classification.

1. Introduction

As an increasingly complicated system, the current Internet architecture has been in place for three decades. Legacy Internet cannot keep up with the ever-changing needs of new apps because it lacks adaptability. Network services may be configured and deployed with unparalleled scalability and flexibility using Software Defined Networking (SDN) [1]. Separating the control plane from the data plane allows for better flexibility and control of the traffic flows in the network. SDNs use the OpenFlow [2] protocol to collect network information in real-time since their architecture is flow-based. However, as shown in [3], the SDN design also brings significant security challenges relating to the control plane, the control-data interface, and the control application interface. It's only been recent that SDN security has risen to the front of people's minds as a significant worry (For instance, see [4] and [5] and references therein). In terms of network security, an intrusion detection system (IDS) plays a critical role. IDS Anomaly seeks to find data that differ from a model's expected behavior. Anomaly-based IDS techniques include artificial neural networks (ANN), support vector machines (SVM), and Bayesian networks. There is a significant False Alarm Rate (FAR) and computational cost associated with these strategies, as noted in [6]. Deep Learning (DL) has superseded traditional machine learning approaches, a novel methodology that achieves improved accuracy. In resource-restricted networks like SDNs, DL has a strong argument for its flexibility since it can analyze raw data and learn high-level characteristics independently.

SDN is a promising architecture that isolates the control function from the forwarding hardware and provides today's data centres more flexibility and programmability. Three layers make up the SDN architecture: a control plane (controller), a foreground and a back-ground, as depicted in Fig. 4. In the data plane, packets are sent via one or more switches. The OpenFlow protocol is used by the vast majority of commercially available switches today. SDN switches are also referred to as OpenFlow-enabled switches because of this. According to the software operating on the application plane, the control plane has a variety of controllers that turn these switches into intelligent devices like routers, IDS, and firewalls. Southbound API (often known as OpenFlow) is responsible for connecting the control plane to SDN switches. Use of the northbound interface allows communication between applications and the controller.

A network operating system (NOS) determines the operation of SDN switches through the Control Plane. It has total control over the network. Network traffic may be studied more effectively by using a central network controller that has access to all of the network's traffic data. Controllers like as NOX and Ryu Onix are among the most popular SDN options.

Because it oversees and regulates the whole network, the centralised controller becomes the primary point of attack. To overwhelm the control plane, an attacker might saturate its computing and communication resources. Unmatched

Manuscript received April 5, 2022
Manuscript revised April 20, 2022
https://doi.org/10.22937/IJCSNS.2022.22.4.44

packets are processed and flow rules are installed into the SDN switch when the controller reaches saturation of its resources. Resources like memory, CPU and buffer are used by the controller as a result of this As a result, the controller takes a long time to process valid requests due to the high volume of flooded requests. This causes the whole network to slow down significantly. Limitation of data-control channel bandwidth: After receiving a new packet, the switch sends the header to the controller and saves the payload in the buffer. When a switch is swamped with requests, the buffer becomes overflowing and the switch fails to function. In order to avoid overloading the communication channel, it then begins transmitting entire packets back to the controller. Genuine users see a delay since it uses up all of the channel's bandwidth.

The Internet now offers a wide range of valuable services because of advancements in Internet technology. There are several security dangers, though. Network Intrusion Detection Systems (NIDS) uses two types of detection methods: signature-based and anomaly-based. When just the attack signature (pattern) is known, signature-based detection (also known as abuse detection) might be beneficial. Anomaly-based detection, on the other hand, may be used to both known and unknown assaults. The idea of "traffic identification" is also used by NIDS, i.e., extracting valuable information from the captured traffic flow and then categorizing the recorded traffic as either normal or attack using a previously taught machine learning algorithm Network infections, eavesdropping, and malicious

Assaults are on the rise. Therefore, network security has shifted to the forefront of public discussion and government priorities. Intrusion detection, on the other hand, can effectively deal with these issues. Network information security relies heavily on intrusion detection. Internet commerce is growing exponentially, and as a result, there is increasing complexity in network behavior characteristics, making intrusion detection more difficult [2], [3]. A critical difficulty that cannot be avoided is recognizing various types of harmful network traffic, particularly spontaneous hostile network activity.

In reality, network traffic may be split into regular and abnormal (normal and malicious traffics). Aside from that, there are five different kinds of network traffic: regular; dos; root to local; user to root; and probe (Probing attacks). As a result, detecting intrusions may be viewed as a challenge of categorization. The accuracy of intrusion detection may be significantly enhanced by enhancing the efficiency of classifiers inefficiently recognizing hostile traffic. In intrusion detection, approaches like machine learning [4,5,6,7,8,9] are frequently employed to spot malicious traffic. On the other hand, these approaches belong to the shallow learning category and often emphasize feature selection and engineering. With low identification accuracy and a high rate of false alarms, they struggle with feature selection and cannot successfully tackle the enormous intrusion data classification challenge.

Deep learning-based approaches for intrusion detection have been suggested one after the other in recent years. The authors in [10] present a mal-ware traffic categorization approach using a convolutional neural network using traffic data as an image. Classifiers do not require any additional input data because this approach uses the original traffic as the input data. According to [11], Recurrent Neural Networks (RNN) can detect network traffic behavior by representing it as a series of changing states over time. Classifying incursion traffic is made easier using an LSTM network, which the authors demonstrate in [12]. According to the findings of the experiments, the LSTM algorithm is capable of discovering all of the attack types buried in the training set.

As you can see, all of the approaches discussed above look at the traffic on the network as a whole as a series of bytes. They don't take full advantage of network traffic domain knowledge. It is analogous to handling traffic as if it were an unrelated entity, like CNN does, and ignores the internal relations of the network traffic. In the first place, network traffic is organized into levels. A network traffic unit is a collection of data packets traveling over a network. A data packet is a packet of data made up of a number of bytes. In the second place, traffic characteristics inside the same and distinct packets change considerably. The separate extraction of sequential elements from distinct packets is required. Put another way, not all traffic aspects are equally significant for traffic categorization when extracting features from particular network traffic.

Recent advances in ensemble-based models include a mixture of multiple ensemble-based models, including random forests (RFs) and the stacking, developed by Zhou and Feng [13] and known as the Deep Forest (DF) or gcForest. As in a multi-layer neuronal network structure, there are several levels in gcForest, but each layer has numerous RFs instead of neurons. To put it another way, one may think of the gsForest as an ensemble of decision tree ensembles. Zhou and Feng note out that gcForest is far more straightforward to train than deep neural networks, which involve a considerable deal of hyperparameter tweaking and vast amounts of training data in order to perform well.

The proposed deep forest classifier includes a novel modification of the screening mechanism for confidence based on the adaptive weighting of every training instance

at each cascade level based on its mean class vector at the previous level. the Deep Forest Adaptive Weighted (AWDF). For applying weights, there are two methods to choose from. This is the first method, in which weighted cases are randomly selected for use in training trees. Because of this, the number of "active" instances decreases as one moves up the forest hierarchy. Weights may be used to establish a splitting rule for training the decision trees in a second approach. AWDF outperforms in numerical experiments, according to the findings.

In summary, the major contributions of the study include designing a unique multistage optimized Deep Learning-based NIDS system reducing computing complexity while improving detection accuracy. Investigate the influence of various feature selection strategies on the NIDS detection performance and time complexity (training and testing). Hyper-parameter optimization approaches and their impact on NIDS detection performance are proposed and investigated. We compare the proposed framework's performance to previous research by increasing detection accuracy, a decrease in FAR, and a smaller training sample and feature set.

The paper is organized as follows. A short description of existing research works is given in Section 2. Section 3 provides the detailed description of the proposed mechanism. Dataset details and numerical experiment results are furnished in Section 4. Concluding remarks are provided in Section 5.

2. Literature Study

A network intrusion detection system (IDS) was proposed in the study [14] to identify hostile activity. A recursive feature reduction is done on the CICIDS2017 dataset before the suggested IDS is evaluated using random forest. This is followed by applying a Deep Multilayer Perceptron Model (DMLP) to the chosen features, with an accuracy of 91%. This model has two steps: sparse AutoEncoder (AE) for unsupervised feature learning and softmax regression classifier trained on the obtained training data. N. Shone et al. presented this self-taught learning model. They used their model on the NSL-KDD dataset and could attain an accuracy of above 98%.

The feature pattern graph model presented by Xiao et al. (2019) [15] collects features from TCP, UDP, and ICMP data. Entropy-based on ports per IP, Tuan et al. (2019) [16] researched important aspects to identify malicious traffic, then utilised the KNN algorithm to detect and discard the traffic. K-Nearest Neighbors (KNN) algorithms were tweaked by Xu et al. (2019) [17] to increase detection precision and efficiency. The controller, in accordance with (Mehr and Ramamurthy, 2019) [18], pulls several fields from the packet in messages (SIP, DIP, Sport, Dport). Entropy is computed using these variables, and the model is trained using both legal and malicious traffic. Flow packets, flow bytes, and the pace of flow entries are used by the SVM to determine if incoming traffic is legitimate or malicious. Using this strategy, the consequences of a DDoS assault are decreased by 36%. There are two parts to the Safeguard Scheme (SGS) proposed by Wang et al. (2019) [19]: malicious traffic detection in OpenFlow switches and a defensive mechanism in the control plane. In order to identify malicious traffic, switches analyse packet properties and apply the Back Propagation Neural Network to see whether there is a malicious flow present (BPNN). Notifications are sent only if the defence module receives an alarm message. By remapping controllers, the defence module may alleviate the pressure on the controllers. Using the KNN based machine learning algorithm,

SDN time series analysis was offered by Fouladi et al. (2020) [20] as a DDoS protection approach. To identify a rapid shift in network traffic, the proposed technique predicts the forthcoming traffic characteristics (number of unique source IP addresses (USIP) and destination IP addresses (UDIP))... Another research used six machine learning models to construct a low-volume DDoS protection system (SVM, J48, Random Forest, Random Tree, REP Tree, MLP). The suggested approach is tested using the CIC DoS 2017 dataset and achieves a 95% detection rate. For both low and large volume assaults, Dehkordi and colleagues (2020) [21] integrated the entropy-based technique with machine learning methods. By choosing the appropriate time period, the detection rate may be maximised. A 99.85 percent success rate is achieved compared to existing DDoS protection methods, which are already in use. For effective detection of DDoS assaults, researchers used the generalised entropy (GE) and information-distance metric to reduce duplicate traffic aspects. They used the SNORT intrusion detection system to gather network traffic in order to decrease controller overhead. It is hoped that this strategy would help enhance the accuracy of deep learning classifiers such as Convolutional Neural Networks (CNNs) and Stacked Auto Encoders (SAEs).

Entropy-based DDoS defences with little computing cost were suggested by Mishra et al. (2021)[22] . Flow rate, entropy, and count thresholds have been set to zero in the proposed technique to begin analysing the data. Initial comparison is made with the initial threshold value of packet flow rate. Switch flow table data are used to calculate entropy after surpassing the threshold. The count value is increased when the calculated entropy falls below the threshold. When the attack count reaches a certain level, an attack alarm message is created. An IP-blocking controller

receives and stores information from switches such as DPID and port numbers during the mitigation phase. In a DDoS assault, Shohani et al. (2021) [23] discovered that the attackers target random hosts rather than a single host. There are four steps to the strategy that has been presented. Initially, a network sleuth examines the communication that is both benign and malicious. Second, a controller uses a statistical model to estimate the amount of flow table misses in OpenFlow switches. Lastly, an exponential weighted moving average (EWMA) linear regression is employed to estimate the threshold of table misses at periodic intervals.

Convolutional neural networks are used in [24, 25] to develop a new network intrusion detection model (CNNs). The CNN model enhances class accuracy when used with small numbers while simultaneously reducing the false alarm rate (FAR). Authors [26] utilized a method like this to reduce the dimensionality of a dataset. They used the KDD CUP and UNB ISCX datasets for PCA tests with Random forest and C4.5 classifier methods. Ten main components were used to compare classification accuracy to 41 features using a classifier called the C4.5.

Natesan et al. [27] proposed an efficient feature selection and classification to attain an optimum detection rate using a parallel computing model and a nature inspired feature selection approach. In addition, the Map Reduce programming paradigm is utilised for the selection of the ideal subset with the least amount of computing effort. Rough Set Theory (RST) and SVM are both used in IDS. To improve the Wei et al [28] DL-based DBN model, it has been recommended that particle swarm and genetic algorithms should be used together. NSL-KDD was used to test the model's performance. The results showed considerable increases in the detection rates of the U2R and R2L classes. The main disadvantage of the suggested model is that it takes longer to train because of its intricate structure.

Jiang et al.[29] proposed an effective IDS system with a deeper hierarchy by combining CNN with long-range bidirectional short-term memory. To increase the number of marginal samples, a SMOTE is utilized. This helps the algorithm effectively learn the features. The problem of unequal power between the sexes has been resolved. The spatial features were extracted using the CNN, while the temporal functions were extracted using the BiLTSM. Experiment with NSL-KDD datasets. The given approach improves accuracy as well as detection rate. Detection rates for minor data classes have risen marginally, but they remain mediocre compared to other attack classes. Because of the complicated structure, training takes longer. Zhang et al. [30] proposed a multi-layer IDS model based on CNN and gcForest. The group of researchers also presented a novel P-Zigzag approach for translating raw data into two-dimensional grey features. In the initial coarse grit layer, they used a superior CNN model for initial detection. The anomalous classes are then classified into N- 1 class using gcForest (caXGBoost) in the finely grained layer. They used a dataset to integrate the UNSW- NB15 and CIC-IDS2017 datasets. According to the results of the studies, the proposed model has a much higher accuracy and detection rate than single algorithms while also reducing the FAR.

Yu and colleagues [31] suggested an IDS model based on the new concept of DL few-shot learning (FSL). One of the objectives is to use a small number of balanced dataset data to train on. The vital feature is extracted and scaled using DNN and CNN incorporated in the model. NSL-KDD datasets were used to get experimental results that showed model efficiency at respectable detection rates for minority groups. Only 2% of the data was used for training to get outstanding results for the studied data set. Xiao et al. [32] present an efficient CNN-based IDS. The important thing Principle Component Analysis and Adobe AE will be used first for feature extraction. After being converted into a 2-D matrix, the feature set is then fed into the neural network. Experiments with the KDD Cup'99 dataset were carried out. Studies show that it saves significant time during development and testing. The main issue with R2L attack classes is that their detection rates are lower than those of other attack types.

The existing models for intrusion detection have several drawbacks, despite several earlier studies in the literature. It's common knowledge that class imbalance occurs in intrusion detection datasets, although many studies ignore it. Also, rather than following a systematic approach, the size of the training sample is typically chosen at random. They are also constrained by the usage of out-of-date datasets like KDD-CUP99. The optimization of hyper-parameters using several strategies was also studied in specific papers. However, just one method was employed instead. In addition, just a few studies looked at the framework's temporal complexity, an often-overlooked statistic.

3. Methodology

This research presents a multistage DL-based NIDS system that minimizes computational complexity while retaining detection performance. This is accomplished in phases, each with a new set of approaches. Figure 1 depicts the suggested methodology's workflow.

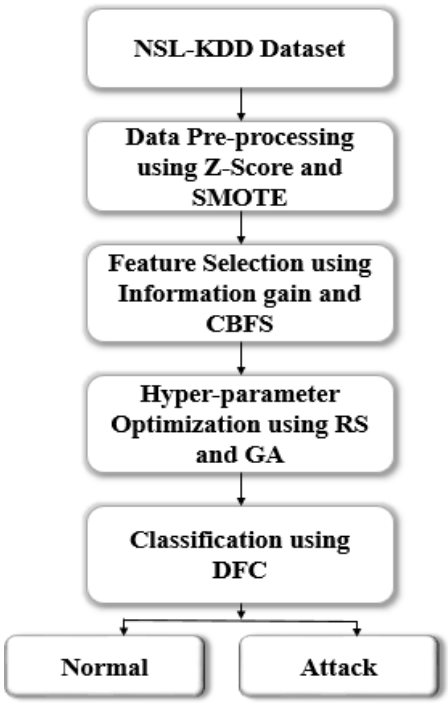

Figure 1: Working Flow of Proposed Methodology

3.1 Data Pre-processing:

In the pre-processing data stage, we will be using the Z-score technique to normalize the data and the SMOTE algorithm to oversample the minority class.

3.1.1 Z-Score Normalization:

Z-score normalization refers to the process of normalizing every value in a dataset such that the mean of all of the values is 0 and the standard deviation is 1.Z-score data normalization is the initial step in the pre-processing data stage. The data must first be encoded with a label encoder to make numerical characteristics out of categorical ones. This is followed by a data normalization process that involves figuring out the normalized value xnorm for every data sample xi as follows:

$$x_{norm} = \frac{x_i - \mu}{\sigma} \quad (1)$$

where μ = Mean vector of an individual feature and μ =Standard deviation

Notably, the Z-score data normalization is carried out since DFC works better with normalized datasets [33].

3.1.2 SMOTE Technique:

Second, the SMOTE technique is used to accomplish minority class oversampling. With this technique, the minority class gets synthesized more often, resulting in a lower class imbalance, hurting the DL classification model's performance [34,35,36]. To increase the performance of the training model, it is critical to executing minority class oversampling, especially for network traffic datasets, which are prone to this problem.

A new minority class instance is generated by analyzing existing examples using the SMOTE technique. The algorithm compiles all instances of the minority class into a single set $Xminority$. For every instance $Xinstance$ within $Xminority$ there generate a new synthetic instance $Xnew$ which is computed as follows.

$Xnew = Xinstance + ran(0,1) * (Xj - Xinstance)$, $j = 1,2, \ldots\ldots, k$ (2)

In which $ran(0,1)$ is considered as a random value within the range [0,1], and Xj is determined by the sampled from the set of k nearest neighbours $\{X1, X2, X3 \ldots\ldots\ldots Xk\}$ of instance $Xinstance$. It should be

Highlighted that SMOTE technique creates new high-quality instances that statistically match samples of the minority class, unlike other oversampling algorithms that reproduce minority class instances.

3.2 Feature selection

To better understand the models' detection performance and temporal complexity, this research analyses two alternative feature selection techniques: information gain-based and correlation-based feature selection. Designing DL models for large-scale systems that generate high-dimensional data is especially significant in this regard [37,38,39].

3.2.1 Information Gain-based Feature Selection

The information gain-based feature selection (IGBFS) algorithm is the first to be studied. It utilizes information theory ideas like entropy and mutual information, as the name says, to pick out the best traits. The IGBFS uses a feature's quantity of information (in bits) to rank it, and only the most information-dense features are sent to the ML model as part of its feature subset. As a result, [40] is the function for evaluating features.

$$i(S;C) = H(S) - H(C|S) = \sum S \in S_I \sum C_I \in CP(S_I, C_I) \log_n \frac{P(S_I, C_I)}{P(S_I) * P(C_I)} \quad (3)$$

In which $(S; C)$ is considered as mutual information in-between class C, and Subset S where H(S) is considered as entropy as well as the distinct feature set of subset S. $H(S|C)$ is computed as a conditional entropy of an uncertain and discrete subset of the feature set S derived from class C, $P(Si, Ci)$ is calculated as the join probability if the class Ci and the feature having value Si.

3.2.2 Correlation-based Feature Selection

The correlation-based feature selection (CBFS) method is the second feature selection technique under consideration. Its ease of use and ranking attributes according to their association with the target class are often utilized [41,42,43]. If a characteristic is thought to be important by CBFS, it is included in the subset (i.e., if it is highly correlated with or predictive of the class). When employing CBFS, the feature subset evaluation is performed using Pearson's correlation coefficient. In other words, the evaluation function is computed as follows:

$$Merit_s = \frac{k * \overline{r_{cf}}}{\sqrt{k + k(k-1) * \overline{r_f}}} \quad (4)$$

Where $Merit_s$ the merit is related to the context feature subset S, the number of features in the subset S is K, $\overline{rcf}$ is computed as average Pearson class-feature correlation as well as $\overline{rff}$ is computed as a feature- feature Pearson correlation.

3.3 Hyper-parameter Optimization

Random search (RS) and Genetic Algorithm (GA) meta-heuristic algorithms are examined in this study to see which is the most effective for hyper-parameter optimization.

3.3.1 Random Search

In a random search, the objective function is fed random inputs, which are then evaluated to see if any patterns emerge. There are no assumptions about how the objective function is structured, which makes it more effective. Using this approach, non-intuitive solutions may be developed for situations involving a large amount of domain knowledge that might impact or prejudice the optimization process being used. Because algorithms that rely on accurate gradients might fail when searching in noisy or non-smooth parts of the search space, it is possible that random searching is the optimum approach in these situations.

A pseudorandom number generator may be used to create a random sample from a given domain. It is necessary to have a well-defined limit or range for each variable before a random value can be selected and evaluated. It may be more economical to produce a big sample of inputs and then analyse them since generating random samples is computationally easy and does not need much memory. Because each sample is distinct, several evaluations may be performed in parallel to speed up the process. The RS method is the first of several hyper-parameter optimization methods. Heuristic optimization models are the ones that use this technique [36]. Like the grid search algorithm [37, 38], RS experiments with many parameter combinations to find the best one. To put this into a mathematical perspective, consider the following model.

$$\int_{parm}^{max} f(parm) \quad (5)$$

The objective function f should be maximized (usually the model's accuracy), and the collection of tuning parameters is called parameter tuning. While grid search searches through all potential possibilities, the RS technique randomly selects a sample of those to test. This is in contrast to grid search. This means that when there are only a few hyper-parameters to consider, RS outperforms grid search. This technology also makes it possible to undertake parallel optimization, which further reduces the computational complexity.

Meta-heuristic optimization techniques are a subset of hyper-parameter optimization methods. These algorithms' goal is to find or provide an effective solution to an optimization issue [44,45,46,47,48,49]. They are ideal candidates for hyper-parameter optimization because they solve combinatorial optimization issues with decreased computing complexity. Using a technique known as Genetic Algorithm, this research examines well-known meta-heuristics for hyper-parameter optimization (GA). Meta-heuristic algorithms influenced by evolution and natural selection are standard. Another well-known example is this one. It is frequently utilized to find superior solutions to combinatorial optimization issues using biologically inspired procedures such as mutation, crossover, and selection. GA algorithms can effectively explore the solution space by using these operators. The GA method operates as follows in the context of ML hyper-parameter optimization:

a. To begin with, create a new population of chromosomes, which are randomly generated solutions. Hyper-parameter value combinations may be found on every chromosome.
b. Apply a fitness function to each chromosome to determine its level of fitness. When employing the chromosomal vectors, the function is generally the ML model's accuracy.
c. Descend the list of chromosomes in order of relative fitness.
d. Crossover and mutation processes can produce new chromosomes to replace those that are no longer needed.
e. Repetition of steps b) to d) until no improvement in performance is observed, or a stopping threshold is reached.
f. DFC (Deep forest Classifier) receives the ideal characteristics and uses them in the final procedure.

3.4 Modified Deep Forest Classifier (DFC)

The Deep Forest is an ensemble-based decision tree approach that emphasizes building deep models using modules that are non-differentiable. It is built around three primary principles that are considered the reasons behind the rich accomplishments of deep models. The reasons are as follows:

- Layer by Layer processing: It is considered one of the significant factors since, no matter how complex the flat model becomes, the features of layer by layer processing cannot be achieved.
- In-model feature transformation: Basic machine learning models work on the original set of features. However, new features are generated during the learning process of a deep model.
- Appropriate model complexity: Large datasets need complex models, basic machine learning models are limited in terms of complexity. However, it is not the case with deep models.

The overall structural working of the deep forest is separated into two broad parts Cascade Forest Structure & Multi-Grained Scanning. Cascade forest structure ensures layer-by-layer processing, while Multi-grained scanning allows the model to achieve sufficient complexity.

3.4.1 Cascade Forest Structure

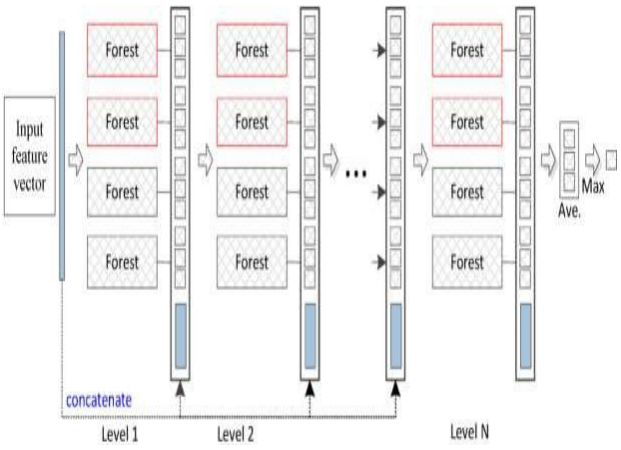

Figure.2. Cascade forest construction

A cascade structure is employed to represent the layer-by-layer processing of raw features. Each layer in the cascade takes input (processed information) from the previous layer and feeds it into the next layer. A layer in the structure can be defined as an ensemble of decision tree forests. It is ensured that diversity is maintained while creating ensembles by including different kinds of forests.

The working in cascading stage proceeds as follows, for a given case, an approximate class distribution will be generated by each forest. This is done by considering the training examples and fraction of different classes at the terminal or leaf node where the particular instance falls, then averaging all the trees in the same forest. This has also been depicted in Figure. 2. The approximated class distribution so obtained forms a vector of classes with the help of k-fold cross-validation. The vector is then concatenated with the original set of features. The result is then forwarded to the next cascading layer. K-fold cross-validation helps in reducing the risk of overfitting. The number of levels is determined automatically based on the performance of the validation set.

A striking difference in deep forest and other deep models is the ability to adaptively change the model complexity by terminating the amount of training data when tolerable. This provides a considerable advantage when working with datasets of varying sizes.

3.4.2 Multi-Grained Scanning

The cascading forest procedure is enriched with the procedure of multi-grained scanning. The inspiration behind the multi-grained scanning procedure was that deep models are generally well suited and good at handling feature relationships. The whole process is depicted in Figure 3. The sliding windows and feature vectors scan raw features are produced. The feature vectors are either negative or positive based on the extraction from the training sample; they are then used to produce class vectors. A completely random forest is trained using the instances extracted from windows having the same size. The concatenation of generated class vectors obtains transformed features.

The actual label of the training sample is used to assign the instances that are extracted from the windows. Though these assignments can be incorrect, they can be attributed to the flipping output method. Also, feature sampling can be performed if transformed feature vectors are too long. The sliding windows size is varied to obtain grained features vectors that are different.

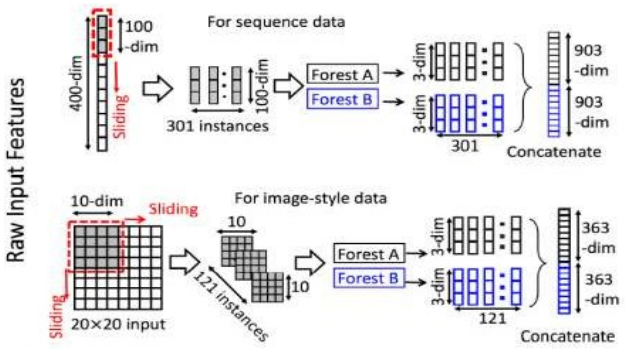

Figure.3. Multi-Grained scanning.

The Deep forest has shown a lot of promise, and its success can be attributed to the following factors:

Fewer hyper-parameters
Data-dependent tuning of model's complexity
Less dependence on GPU

3.5 Security Considerations

The scalable manner uses distributed parallel ML algorithms with several optimization strategies to manage an extensive network and host event volumes. The scalable design also enables a quick and parallel examination of network and host-level actions using the overall graphic processing unit (GPU) processing capacity.

It is a signature-based NIDS system with a multistage optimization of DFC-based NIDS frameworks. A good example of this is that the framework oversamples the minority class in network traffic, which is often the attack class [50]. Observations of known started assaults teach the framework new things. It's important to remember that Since it is taught using a binary classification model, the framework may be used as an anomaly-based NIDS, classifying any unusual activity as an attack. As a module inside a larger security framework/policy, this framework can be used by a person or organization to protect themselves better. This security framework/policy can include other techniques such as firewalls, deep packet inspection, user access control, and user authentication procedures can be included in this security framework/policy [51,52]. This would provide a safe architecture with many layers that can protect user data and information while maintaining privacy and security.

4. Results and Discussions

All tests were conducted on an Ubuntu 14.0.4 LTS with Python. Use Scikit-learn to implement all traditional machine learning algorithms. Using GPU-enabled TensorFlow4, three DNNs were developed with a higher Keras5 framework backend. The GPU was NVidia GK110BGL Tesla K40, and the CPU was configured to run on a 1 Gbps Ethernet network (32 GB RAM, 2 TB hard disk. The following test cases were selected to assess the performance of the proposed and different classical deep learning classifiers on the NSL-KDD dataset. However, the deep forest has less hyperparameters and it can adjust the hyper-parameters automatically during the training process.

4.1 Dataset Description

We considered the widely available and widely used leak detection data sets in earlier work: the NSL-KDD data set [53]. The dataset has standard data, and four different types of attacks include Probe, U2R, R2L, and DoS. There are 42-dimensional features presented in each intrusion record, and it is categorized into a 3-dimensional symbol feature, a traffic type label, and a 38-dimensional digital feature. Table 1 shows the description of the data set.

Table 1: Dataset Description

Category	Train	Test
Normal	77423	9899
DoS	12256	8458
Probe	4897	2211
U2R	65	211
R2L	789	2989
Total	128983	22897

4.2 Performance metrics

The basis truth value is necessary for the evaluation of the various statistical measures. In binary classification, the foundation truth consisted of several connection registers that were normal or attacked. Let L and A be the sum of usual and Attack logs in the test dataset and use the subsequent terms to determine the excellence of the classification model:

- True Positive (TP) - the sum of connection records properly categorized to the Usual class.
- False Negative (FN) - the sum of Attack connection records incorrectly categorized to the Usual connection record.
- True Negative (TN) - the sum of connection records properly categorized to the Attack class.
- False Positive (FP) - the sum of Normal linking records wrongly categorized to the Attack linking record.

The following evaluation metrics are examined based on the above-given terms.

$$Accuracy = \frac{TN+TP}{TN+TP+FN+FP} \times 100 \quad \ldots\ldots\ldots (6)$$

$$f-measure = \frac{2TP}{2TP * FP + FN} \times 100 \ldots\ldots (7)$$

$$prcesion = \frac{TP}{(FP+TP)} \times 100 \ldots\ldots\ldots\ldots\ldots (8)$$

$$Recall = \frac{TP}{FN+FP} \times 100 \ldots\ldots\ldots\ldots\ldots\ldots (9)$$

4.3 Performance of the Proposed Metrics

The proposed evaluation has been segregated into major parts, such as binary classification and Multiclass classification. The binary classification detects the Attack or normal communication. Multiclass classification has

detected the various types of Attack, which is presented in the dataset.

A. Multiclass classification

The detailed results for the classification of the proposed system for multiclass are reported in this section.

Table.2. Comparative analysis of multiclass on Proposed DFC method

Category	Accuracy (%)	Precision (%)	Recall (%)	F-score (%)
Normal	79.08	87.27	94.60	91.47
DoS	82.75	93.16	87.47	83.42
Probe	83.43	75.81	78.62	87.28
U2R	81.33	71.04	76.47	92.94
R2L	87.19	72.32	69.04	81.15

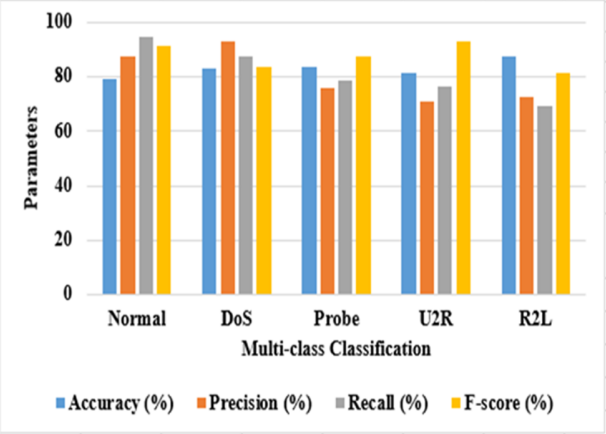

Figure 4: Graphical Representation of Proposed DFC for different categories on NSL-KDD Dataset.

While in the normal category, the proposed method achieved 79.08% accuracy, 87.27% precision, 94.60% recall, and 91.47% F1-measure. While comparing with other types of recall experiments, the proposed DFC achieved high performance on the standard category only. The proposed method achieved high precision (i.e. 93.16%) on the DoS category and high F1-measure (i.e.92.94%) only on the U2R category. In other categories like Probe, U2R, R2L, the proposed method achieved nearly 71% to 75% of precision, 69% to 78% of recall, and 81% to 87% of accuracy, where DFC achieved less recall value (i.e.69.04%) on R2L category only.

B. Binary classification

The detailed results for Binary classification of several classical ML and DL classifiers and proposed systems are reported in this section. Figure 5 shows the graphical analysis of the proposed classifier.

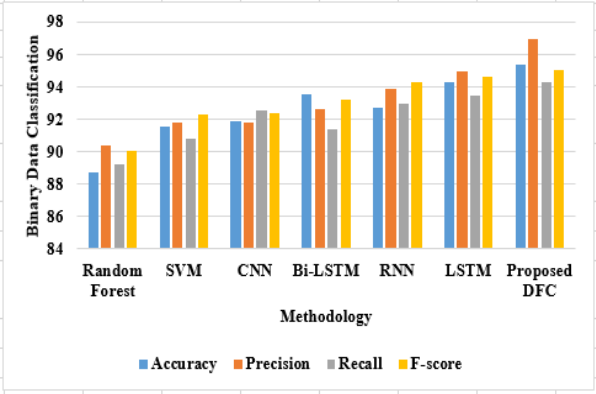

Figure 5: Graphical Representation of Proposed DFC with existing techniques for binary data classification

Table.3. Comparative analysis of binary class on Proposed with various existing algorithms.

Algorithm	Accuracy	Precision	Recall	F-score
Random Forest (RF)	88.70	90.41	89.21	90.02
Support Vector Machine (SVM)	91.50	91.82	90.81	92.27
Convolutional Neural Network (CNN)	91.90	91.78	92.52	92.41
Bi-directional Long Short Term Memory (Bi-LSTM)	93.50	92.63	91.38	93.18
Recurrent Neural Network (RNN)	92.70	93.90	92.93	94.32
LSTM	94.27	94.91	93.47	94.63
Proposed DFC	95.32	96.95	94.24	95.02

From the above table 2, it is proved that the proposed DFC achieved better accuracy (95.32%), precision (96.95%), recall (94.24%), and F-score (95.02%) than existing ML and DL techniques. The existing methods, namely SVM and CNN, achieved nearly 91% to 92% accuracy, precision, recall, and F-score. The other ways, such as Bi-LSTM, RNN, and LSTM, gained almost 92% to 94% of accuracy, precision, recall, and F-score on binary data classification. While compared with all techniques, Random Forest (RF) provides low results in all parameters, i.e. 88.70% accuracy, 90.41% precision, 89.21% recall, and 90.02% F-score.

C. Minimal feature analysis

Optimizing functionality is an essential step to detect intrusion. This is a crucial step towards identifying more correctly the different sorts of attacks. Without optimizing features, a misclassification of assaults may be possible, and the development of a model would take a long time. The methods for selecting functions reduced the training and testing time significantly and enhanced the rate of detecting intrusions. Two experiment trials are performed on limited feature sets on the NSL- KDD to assess the performance of the proposed method and static machine learning classifications. Table 4 provides detailed results. Compared to tests in 4 feature sets, the experiments with 11 and 8 feature sets were good. In addition, experiments with 11 groups of functionalities were successful compared to the eight sets. The performance difference of 11 to 8 minimum set of features is minor.

- Table.4. Comparative analysis of test results using minimal feature sets.

	Accuracy (%)		
Algorithm	**11 features**	**8 features**	**4 features**
RF	88.27	89.67	87.18
SVM	91.82	92.43	88.79
CNN	92.04	92.94	91.41
Bi-LSTM	93.13	93.06	92.07
RNN	94.43	94.16	93.66
LSTM	94.89	94.87	94.09
Proposed DFC	96.90	95.08	94.39

The above table consists of validated techniques with proposed methods for different attacks, namely Normal, DoS, Probe, U2R, and R2L. When the number of features is minimized, the accuracy of the proposed DFC is also

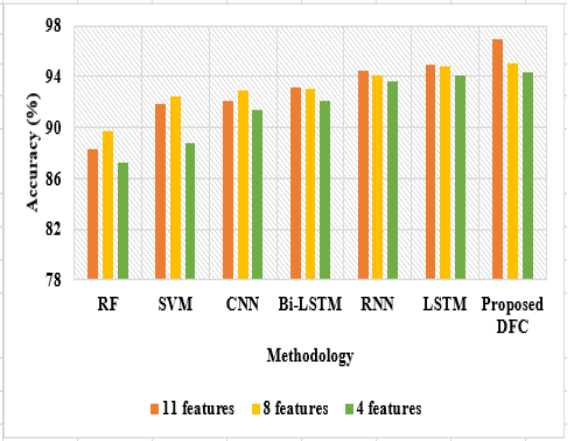

Figure 6: Graphical Representation of proposed DFC with existing classifiers in terms of accuracy while reducing the features set.

minimized. For instance, its accuracy is 96.90%, when the 11 features are reduced, and its accuracy is 95.08%, when eight features are reduced, finally, it reaches 94.39% of accuracy when only four features are reduced. COMPARED WITH EXISTING METHODS, the RF technique achieved low accuracy, i.e., nearly 87% to 89% for all features reduction. When the feature set is 8, the existing methods such as SVM, CNN, Bi-LSTM, RNN, and LSTM achieved 92.43%, 92.94%, 93.06%, 94.16%, and 94.87% of accuracy, but the same techniques gained 88.79%, 91.41%, 92.07%, 93.66%, and 94.09% of accuracy only.

5. Conclusion

Due to the growing reliance of individuals and businesses on the Internet and their concerns about the security and privacy of their activities, the field of cyber-security has attracted considerable interest from both industry and academia. Increasing resources have been budgeted and deployed to safeguard modern Internet-based networks from malicious assaults. Thus, many NIDS kinds have been put up in the literature. There is still space for improvement in NIDS performance despite the advancements that have been made. High volumes of network traffic data, constantly changing settings, and a variety of attributes acquired as part of training datasets (high dimensional datasets) all contribute to the requirement for real-time intrusion detection and analysis. This can only be done by selecting and optimizing the most appropriate set of DL-based detection models' parameters. That is why the authors of this research advocated for a new, simpler, and more efficient DL-based NIDS system. in terms of both complexity and detection performance. This research initially looked at the effects of oversampling approaches on the training sample size for the models. It

established the smallest training sample size necessary for an efficient intrusion detection system using NSL-KDD.

According to the findings of the experiments, adopting the SMOTE oversampling approach reduces the size of the training datasets. IGBFS and CBFS feature selection strategies have been used in this study, and their effects on feature set size, training sample size, and model detection performance have all been studied. The results of the experiments revealed that the feature selection approaches might lower the size of the feature collection. For future study, other models, such as deep-learning classifiers with learning rate optimization techniques, can be investigated because they excel on non-linear and large datasets.

References

[1] Dong, B., & Wang, X. (2016, June). Comparison deep learning method to traditional methods using for network intrusion detection. In *2016 8th IEEE International Conference on Communication Software and Networks (ICCSN)* (pp. 581-585). IEEE.

[2] Zarpelão, B. B., Miani, R. S., Kawakani, C. T., & de Alvarenga, S. C. (2017). A survey of intrusion detection in Internet of Things. *Journal of Network and Computer Applications*, *84*, 25-37.

[3] Mukherjee, B., Heberlein, L. T., & Levitt, K. N. (1994). Network intrusion detection. *IEEE network*, *8*(3), 26-41.

[4] Wagh, S. K., Pachghare, V. K., & Kolhe, S. R. (2013). Survey on intrusion detection system using machine learning techniques. *International Journal of Computer Applications*, *78*(16).

[5] Sultana, N., Chilamkurti, N., Peng, W., & Alhadad, R. (2019). Survey on SDN based network intrusion detection system using machine learning approaches. *Peer-to-Peer Networking and Applications*, *12*(2), 493-501.

[6] Panda, M., Abraham, A., Das, S., & Patra, M. R. (2011). Network intrusion detection system: A machine learning approach. *Intelligent Decision Technologies*, *5*(4), 347-356.

[7] Li, W., Yi, P., Wu, Y., Pan, L., & Li, J. (2014). A new intrusion detection system based on KNN classification algorithm in wireless sensor network. *Journal of Electrical and Computer Engineering*, *2014*.

[8] Garg, S., & Batra, S. (2017). A novel ensembled technique for anomaly detection. *International Journal of Communication Systems*, *30*(11), e3248.

[9] Kuang, F., Xu, W., & Zhang, S. (2014). A novel hybrid KPCA and SVM with GA model for intrusion detection. *Applied Soft Computing*, *18*, 178-184.

[10] Wang, W., Zhu, M., Zeng, X., Ye, X., & Sheng, Y. (2017, January). Malware traffic classification using convolutional neural network for representation learning. In *2017 International Conference on Information Networking (ICOIN)* (pp. 712-717). IEEE.

[11] Torres, P., Catania, C., Garcia, S., & Garino, C. G. (2016, June). An analysis of recurrent neural networks for botnet detection behavior. In *2016 IEEE biennial congress of Argentina (ARGENCON)* (pp. 1-6). IEEE.

[12] Staudemeyer, R. C., & Omlin, C. W. (2013). ACM press the south African institute for computer scientists and information technologists conference-east London south Africa (2013.10. 07–2013.10. 09) proceedings of the south African institute for computer scientists and information technologists co. In *Proc. South African Inst. Comput. Scientists Inf. Technol. Conf.* (pp. 252-261).

[13] Zhou, Z. H., & Feng, J. (2017). Deep forest: Towards an alternative to deep neural networks. arXiv. *arXiv preprint arXiv:1702.08835*.

[14] Ustebay, S., Turgut, Z., & Aydin, M. A. (2018, December). Intrusion detection system with recursive feature elimination by using random forest and deep learning classifier. In *2018 international congress on big data, deep learning and fighting cyber terrorism (IBIGDELFT)* (pp. 71-76). IEEE.

[15] Xiao, Y., Fan, Z. J., Nayak, A., & Tan, C. X. (2019). Discovery method for distributed denial-of-service attack behavior in SDNs using a feature-pattern graph model. *Frontiers of Information Technology & Electronic Engineering*, *20*(9), 1195-1208.

[16] Tuan, N. N., Hung, P. H., Nghia, N. D., Van Tho, N., Phan, T. V., & Thanh, N. H. (2019, October). A Robust TCP-SYN Flood Mitigation Scheme Using Machine Learning Based on SDN. In 2019 *International Conference on Information and Communication Technology Convergence (ICTC)* (pp. 363-368). IEEE.

[17] Xu, Y., Sun, H., Xiang, F., & Sun, Z. (2019). Efficient DDoS detection based on K-FKNN in software defined networks. *IEEE Access*, *7*, 160536-160545.

[18] Mehr, S. Y., & Ramamurthy, B. (2019, December). An SVM based DDoS attack detection method for Ryu SDN controller. In *Proceedings of the 15th international conference on emerging networking experiments and technologies* (pp. 72-73).

[19] Wang, Y., Hu, T., Tang, G., Xie, J., & Lu, J. (2019). SGS: Safe-guard scheme for protecting control plane against DDoS attacks in software-defined networking. *IEEE Access*, *7*, 34699-34710.

[20] Fouladi, R. F., Ermiş, O., & Anarim, E. (2020). A DDoS attack detection and defense scheme using time-series analysis for SDN. *Journal of Information Security and Applications*, *54*, 102587.

[21] Dehkordi, A. B., Soltanaghaei, M., & Boroujeni, F. Z. (2021). The DDoS attacks detection through machine learning and statistical methods in SDN. *The Journal of Supercomputing*, *77*(3), 2383-2415.

[22] Mishra, A., Gupta, N., & Gupta, B. B. (2021). Defense mechanisms against DDoS attack based on entropy in SDN-cloud using POX controller. *Telecommunication systems*, *77*(1), 47-62.

[23] Shohani, R. B., Mostafavi, S., & Hakami, V. (2021). A Statistical Model for Early Detection of DDoS Attacks on Random Targets in SDN. *Wireless Personal Communications*, 1-22.

[24] 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.

[25] Wu, K., Chen, Z., & Li, W. (2018). A novel intrusion detection model for a massive network using convolutional neural networks. *Ieee Access*, *6*, 50850-50859.

[26] Vasan, K. K., & Surendiran, B. (2016). Dimensionality reduction using principal component analysis for network intrusion detection. *Perspectives in Science*, *8*, 510-512.

[27] Natesan, P., Rajalaxmi, R. R., Gowrison, G., & Balasubramanie, P. (2017). Hadoop based parallel binary bat algorithm for network intrusion detection. *International Journal of Parallel Programming*, *45*(5), 1194-1213.

[28] Wei, P., Li, Y., Zhang, Z., Hu, T., Li, Z., & Liu, D. (2019). An optimization method for intrusion detection classification model based on deep belief network. *IEEE Access*, *7*, 87593-87605.

[29] Jiang, K., Wang, W., Wang, A., & Wu, H. (2020). Network intrusion detection combined hybrid sampling with deep hierarchical network. *IEEE Access*, *8*, 32464-32476.

[30] Zhang, X., Chen, J., Zhou, Y., Han, L., & Lin, J. (2019). A multiple-layer representation learning model for network-based attack detection. *IEEE Access*, *7*, 91992-92008.

[31] Yu, Y., & Bian, N. (2020). An intrusion detection method using few-shot learning. *IEEE Access*, *8*, 49730-49740.

[32] Xiao, Y., Xing, C., Zhang, T., & Zhao, Z. (2019). An intrusion detection model based on feature reduction and convolutional neural networks. *IEEE Access*, *7*, 42210-42219.

[33] Ali Alheeti, K. M., & McDonald-Maier, K. (2018). Intelligent intrusion detection in external communication systems for autonomous vehicles. *Systems Science & Control Engineering*, *6*(1), 48-56.

[34] Chen, Z., Yan, Q., Han, H., Wang, S., Peng, L., Wang, L., & Yang, B. (2018). Machine learning based mobile malware detection using highly imbalanced network traffic. *Information Sciences*, *433*, 346-364.

[35] Chen, Z., Yan, Q., Han, H., Wang, S., Peng, L., Wang, L., & Yang, B. (2018). Machine learning based mobile malware detection using highly imbalanced network traffic. *Information Sciences*, *433*, 346-364.

[36] Tan, X., Su, S., Huang, Z., Guo, X., Zuo, Z., Sun, X., & Li, L. (2019). Wireless sensor networks intrusion detection based on SMOTE and the random forest algorithm. *Sensors*, *19*(1), 203.

[37] Çatalkaya, M. B., Kalıpsız, O., Aktaş, M. S., & Turgut, U. O. (2018, September). Data feature selection methods on distributed big data processing platforms. In *2018 3rd International Conference on Computer Science and Engineering (UBMK)* (pp. 133-138). IEEE.

[38] Krishna, R. S. B., & Aramudhan, M. (2014, July). Feature selection based on information theory for pattern classification. In *2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)* (pp. 1233-1236). IEEE.

[39] Bonev, B. (2010). *Feature selection based on information theory*. Universidad de Alicante.

[40] Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., & Liu, H. (2017). Feature selection: A data perspective. *ACM Computing Surveys (CSUR)*, *50*(6), 1-45.

[41] Hall, M. A. (1999). Correlation-based feature selection for machine learning.

[42] Moubayed, A., Injadat, M., Shami, A., & Lutfiyya, H. (2018, March). Relationship between student engagement and performance in e-learning environment using association rules. In *2018 IEEE world engineering education conference (EDUNINE)* (pp. 1-6). IEEE.

[43] Koch, P., Wujek, B., Golovidov, O., & Gardner, S. (2017). Automated hyperparameter tuning for effective machine learning. In *proceedings of the SAS Global Forum 2017 Conference* (pp. 1-23). Cary, NC: SAS Institute Inc..

[44] Yang, L., & Shami, A. (2020). On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, *415*, 295-316.

[45] Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of machine learning research*, *13*(2).

[46] Injadat, M., Moubayed, A., Nassif, A. B., & Shami, A. (2020). Systematic ensemble model selection approach for educational data mining. *Knowledge-Based Systems*, *200*, 105992.

[47] Injadat, M., Moubayed, A., Nassif, A. B., & Shami, A. (2020). Multi-split optimized bagging ensemble model selection for multi-class educational data mining. *Applied Intelligence*, *50*(12), 4506-4528.

[48] Bianchi, L., Dorigo, M., Gambardella, L. M., & Gutjahr, W. J. (2009). A survey on metaheuristics for stochastic combinatorial optimization. *Natural Computing*, *8*(2), 239-287.

[49] Cohen, G., Hilario, M., & Geissbuhler, A. (2004, November). Model selection for support vector classifiers via genetic algorithms. An application to medical decision support. In *International Symposium on Biological and Medical Data Analysis* (pp. 200-211). Springer, Berlin, Heidelberg.

[50] Salo, F., Injadat, M., Nassif, A. B., Shami, A., & Essex, A. (2018). Data mining techniques in intrusion detection systems: A systematic literature review. *IEEE Access*, *6*, 56046-56058.

[51] Moubayed, A., Refaey, A., & Shami, A. (2019). Software-defined perimeter (sdp): State of the art secure solution for modern networks. *IEEE network*, *33*(5), 226-233.

[52] Kumar, P., Moubayed, A., Refaey, A., Shami, A., & Koilpillai, J. (2019, April). Performance analysis of sdp for secure internal enterprises. In *2019 IEEE Wireless Communications and Networking Conference (WCNC)* (pp. 1-6). IEEE.

[53] Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009, July). A detailed analysis of the KDD CUP 99 data set. In *2009 IEEE symposium on computational intelligence for security and defense applications* (pp. 1-6). IEEE.

!

Ms. Saritha Anchuri, Research scholar in JNTU college of Engineering Anantapuramu, received M.Tech from Rajeev Gandhi Memorial College of Engineering and Technology, Nandyal. At present she is working as assistant professor in the Department of CSE at Sri Venkateswara Engineering College, Tirupathi. She is having 10 years of teaching experience.

Dr. B. RamaSubbaReddy, Professor of Sri Venkateswara College of Engineering, Tirupathi has received Ph.d from Sri Venkateswara University, Tirupathi. He worked in various prestigious institutions both in Telangana and Andhra Pradesh as Head of the Department and is having more than 20 years of teaching experience. His areas of interest include Data mining, Computer Networks, Network security and cryptography, Machine Learning.

Dr. A. Suresh Babu, Professor, JNTUACEA has received Ph.d from JNTU, Hyderabad, India. Since then he Served as a Head of the Department , Computer Science & Engineering, JNTUACE, Pulivendula , Chairman of UG & PG Board Of Studies , CSE,JNTUACE(Autonomous),Pulivendula, Worked as Addl. Controller of Examinations, JNTUA from Feb 2009 to March 2011, Worked as Deputy Warden for I YEAR Boys Hostel, JNTUACEA. Currently he is working as an Controller of Examinations, JNTUA, Anantapuramu. His research areas include Data mining, Cloud Computing, Big Data Analytics.