Journal of Information Technology Management

Print ISSN: 2008-5893

Online ISSN: 2423-5059

# Implementation of Intrusion detection and prevention with Deep Learning in Cloud Computing

**Doddi Srilatha***

*Corresponding Author, School of Computing and Information Technology, REVA University, Bengaluru, India-560064; Department of CSE, Sreenidhi Institute of Science and Technology, Hyderabad, India-501301. E-mail: doddisrilatha@gmail.com

**N.Thillaiarasu**

School of Computing and Information Technology, REVA University, Bengaluru, India-560064. E-mail: thillai888@gmail.com

## Abstract

An administrator is employed to identify network security breaches in their organizations by using a Network Intrusion Detection and Prevention System (NIDPS), which is presented in this paper that can detect and preventing a wide range of well-known network attacks. It is now more important than ever to recognize different cyber-attacks and network abnormalities that build an effective intrusion detection system plays a crucial role in today's security. NSL-KDD benchmark data set is extensively used in literature, although it was created over a decade ago and will not reflect current network traffic and low-footprint attacks. Canadian Institute of Cyber security introduced a new data set, the CICIDS2017 network data set, which solved the NSL-KDD problem. With our approach, we can apply a variety of machine learning techniques like linear regression, Random Forest and ID3. The efficient IDPS is indeed implemented and tested in a network environment utilizing several machine learning methods. A model that simulates an IDS-IPS system by predicting whether a stream of network data is malicious or benign is our objective. An Enhanced ID3 is proposed in this study to identify abnormalities in network activity and classify them. For benchmark purposes, we also develop an auto encoder network, PCA, and K-Means Clustering. On CICIDS2017, a standard dataset for network intrusion, we apply Self-Taught Learning (STL), which is a deep learning approach. To compare, we looked at things like memory, Recall, Accuracy, and Precision.

**Keywords:** IDPS (Intrusion Detection and Prevention System), Network Security.

Journal of Information Technology Management, 2023, Vol. 15, Special Issue, pp. 1-18
Published by University of Tehran, Faculty of Management
doi: https://doi.org/ 10.22059/jitm.2022.89407
Article Type: Research Paper

Received: February 26, 2022
Received in revised form: May 07, 2022
Accepted: August 18, 2022
Published online: October 19, 2022

## Introduction

The Internet has risen to prominence as a means of connecting people throughout the world. Using the Internet, people from all over the globe can interact and share information. The emergence of cloud computing technology, enables the enterprise organization and individual to rapid acquisition and deployment of computing resources in the form of service, has brought great convenience to users, so the cloud computing quickly around the world get extensive application and rapid development. However, with the popularization of cloud computing and diversification of network attack means, the corresponding security problem of cloud computing also rises rapidly, which has become a key factor restricting the development of cloud computing. When user utilizes the Internet, we don't know who's hacking into the system. They may cause network services to be down for a lengthy period, or they can cause them to be sluggish or unavailable for a short time. Port Scan (or probe) and Denial of Service (DoS) attacks are the two most common forms of network attacks. For example, an HTTP flood or UDP flood may bring a system to a standstill by DDoS attacks. Probe attacks are used to find open ports in a computer victim's system. Users and network administrators need to be aware of these threats before they may harm the network. To keep their computer networks safe from intruders and hackers, many businesses currently use security measures like firewalls. There are many forms of network attacks, and a firewall system alone is not sufficient to protect a network from all of them. For example, a firewall can't prevent a Denial of Service attack on open ports that are necessary for network services. An IDPS might be utilized in conjunction with a firewall to further safeguard network services against unauthorized access.

As a result, the acronyms for both intrusion detection and prevention systems are often mashed together. Instead of only looking for current or impending breaches in a network's security policies, intrusion prevention systems actively strive to prevent them from happening. So we cannot have IPS without IDS, which are meant to warn a business of ongoing cyber threats and perhaps react to them automatically. Although these systems are not all the same, they all have different goals and methods of accomplishing them. An organization's particular environment and business requirements should guide the selection of IDPS. It's also crucial to remember that not all solutions for intrusion protection systems are "either/or" choices. IDPS systems that are both host-based and network-based, or multiple network-level IDS systems operating side by side, may be necessary to provide full threat detection and prevention. IDS have been developed using various machine learning approaches, such as ANN (Artificial Neural Networks) & SVM (Support vector machines). Normal traffic may be distinguished from abnormal traffic using these IDS classifiers (Sabhnani M & Serpen G, 2003). To improve classification results, many IDSs conduct a feature selection operation by extracting a subset of important features from the traffic dataset. Through the reduction of duplicate features and noises, feature selection serves to eliminate the chance of inaccurate training. Deep learning algorithms for audio, image, and

speech processing have recently been effectively utilized. To obtain a best feature representation from a large volume of unlabeled data, these approaches try to apply these learned features to a small amount of labeled data in the classification. A variety of distributions may be used to generate the labeled and unlabeled data (Nazarzadeoghaz N et al., 2020). They must, however, relate to one another. Network services may be effectively guarded against malicious activity or other network threats using the Intrusion Detection and Prevention System (IDPS) described in this paper (Garbis J & Chapman J W, 2021).

Our network-based IDPS, which use machine learning algorithms to recognize online network data, is described in this study. Our detection and classification strategy may use a variety of well-known machine learning techniques. Probing & denial-of-service attacks are classified utilizing just a few network traffic data attributes, whereas typical network activity is categorized using the same data (Muthukumaran V et al., 2021). To identify intrusions more quickly and automatically, our IDPS can considerably minimize the time it takes.

Organizations and individuals may now store and process their data on the cloud without having to worry about purchasing, maintaining, and managing their own hardware. For large-scale cloud computing implementations, security is a major problem. As a security measure, intrusion detection and prevention (IDP) may be used (Al-Shourbaji I & Al-Janabi S, 2017). IDP approaches have been extensively explored and assessed for their strengths and shortcomings in cloud computing security in this article. In compared to classic IDP approaches, hypervisor-based and distributed IDS have showed promising security characteristics in cloud computing environments (Maithili, K et al., 2018).

Our society has grown more reliant on technology in the past decade. News, stock prices, email, and even online shopping are all delivered over computer networks (Raina R et al., 2007). These systems must be protected against a variety of risks to their integrity and availability. Computer systems may be attacked by anybody with the desire and capacity to do so, whether they amateur hackers, competing companies, terrorists, or even foreign governments (Labib K & Vemuri R, 2002). Because of this, information security is vital to the overall health and well-being of society. Concerns about the need for robust information security systems utilising firewalls, intrusion detection software (IDP), encryption, and authentication have grown as the use of wired and wireless communication networks, Web application software, cloud computing, and the Internet has rapidly expanded and widening adoption of electronic data processing & business has increased (Puttini R S et al., 2003).

## Literature Review

Some researchers in the literature evaluate intrusion detection systems based on a variety of methodologies and classification algorithms. Most prior studies evaluated their IDS performance using the 10-year-old offline KDD99 dataset, which has 41 features. Multilayer

perceptrons are trained using an upgraded resilient back propagation training technique in (Amini M et al., 2006). The weight update equation now includes an optimum or ideal learning factor to speed up convergence. An anomaly intrusion detection dataset called NSL-KDD was used in the performance and assessments. A 94.7% detection rate was achieved by the developed system, which was able to identify data in a timely and accurate manner throughout the testing (Naoum R S et al., 2012). For intrusion detection in, the authors proposed a method in which instead of considering all the possible aspects of an attack and spending so much time on investigating them, only the most important one is selected and intrusion detection done with the help of Neural Networks (NN). DTR (Detection Rate) for a several range of attacks is improved by this framework compared to Intrusion identification frameworks that use all the features and choose features using hereditary computation and MLP-NN as the classifier for classification (Jasim Y A, 2018, pp 49-65). The suggested framework is able to recognize interruptions with more accuracy, notably for R2L, U2R, and DoS attacks.

Intrusion detection systems, which have become more important for protecting computers from hackers that are increasingly damaging, were reported in employing neural networks as a tool in the intrusion detection system (Dastanpour A et al., 2016). The neural network's ability to provide a diagnosis quickly is a major factor in its development. Since these protocols (TCP, UDP) enable communication to flow via the network, a variety of attack features have been evaluated for both the standard and odd packets. To train the neural network, the findings of these analyses were applied to conventional and uncommon packet structures and patterns. The study employed the Standard Back propagation Algorithm, which is one of several neural network learning techniques (Albahar M Aet al., 2020). The Resilient Back propagation Algorithm was employed by the researcher to increase the network's intelligence, speed, and capacity to classify. This system's output can distinguish between normal packets and abnormal packets which classifies them into 5 groups with an efficiency of up to 100% of the specified packets (Ng A & Autoencoder S, 2011).

Certain features of intrusion detection systems, such as adaptability, fault tolerance, and high computing speed, are available. So, the development of an effective intrusion detection model is essential for enhancing detection rates and reducing false alarms. In (Mehmood Y et al., 2013), the authors employed the following approaches to identify attacks on intrusion detection systems. There is a significant risk of network communication being hacked as a result of its widespread use and accessibility, as well as its complexity (Saikumar K et al., 2022). The attacks can be stopped by looking for anomalies in the data that is being transferred and processed while communicating. A system's security depends on the ability to detect abnormalities. According to (Moloja D & Mpekoa N, 2017), machine learning is critical in detecting anomalies and intrusions in network communication. Several effective artificial neural network models rely on the word "regularisation," which is an important part of training machine learning models since it causes regularisation in the model training

(Muthukumaran V et al. 2021). In order to classify and detect network communication efficiency abnormalities, this approach is then combined with an ANN. Regularization serves as a deterrent to learning a more complex or flexible model. To put it another way, the machine learning model is capable of coping with a wide range of data sets. NSL-KDD, CIDDS-001, and UNSWNB15 (External and Internal Server Data) were all utilised in the development process. Regularization algorithms based on the L1 and L2 norms have been examined to see whether the new regularizer achieves higher True Positive Rates (TPRs) and accuracy. An important ability to identify intrusions is shown by the suggested regularizer.

Today's, supervised learning is extremely constrained, despite its many applications. In particular, the algorithm's input characteristics x must still be specified by hand in the vast majority of cases. A supervised learning algorithm may perform well once a decent feature representation is provided. Researchers who've spent years and years of their careers laboriously hand-engineering vision, audio, or text characteristics may today be found in fields like computer vision, audio processing, and natural language processing. It's hard not to question whether we can do better than what's being done here in terms of feature engineering. Hand-engineering methods don't scale well to new challenges, and it would be wonderful if we could have algorithms that could automatically develop even better feature representations than the hand-engineered ones (Chen L et al., 2020).

A-optimality criteria and L-optimality criterion with a specified L matrix are used to estimate the asymptotic distribution of the generic sub-sampling estimator. A two-stage adaptive approach is used to overcome this problem since the optimum sub-sampling probabilities are dependent on the unknowns.

In this, however, its open and dispersed design, which is susceptible to attackers, is a key roadblock to its success. The most prevalent method for detecting assaults on the cloud is the Intrusion Detection System (IDS). An overview of various cloud incursions is presented in this work. In the next section, we examine some of the current cloud-based intrusion detection systems (IDS) in terms of their kind, location and detection time. An evaluation of each technique's restrictions is also provided, allowing the researcher to determine whether or not it meets the security needs of the cloud computing system in question. To deal with cloud security threats, we recommend deploying an IDS with a variety of detection techniques.

In (Kiran Kumar M et al., 2022) New technologies and solutions were not conceivable a few years ago because of the growth of ICT. Mobile voting is one of these emerging technologies, which allows voters to utilise their mobile devices to cast their ballots. In recent years, M-voting has been receiving a lot of attention throughout the globe because of the universality of mobile phones and the ease and convenience they bring to their users. Electronic voting has several benefits over paper ballots. Voting through a rapidly expanding gadget like a mobile phone, on the other hand, introduces a slew of new security concerns. A cloud intrusion detection and prevention system, presented in this research, was built to improve mobile phone security during voting. Results from simulations show that the security

system is efficient, dependable, and secure. Until date, no study has combined an intrusion detection and prevention system, cloud computing, and M-voting into a single system, making this report an important addition to the body of knowledge.

Security in the cloud has been a major concern in recent years, as the use of cloud computing has increased. Detecting and preventing an attack is preferable than reacting to an assault after it has already occurred. This study offers architecture capable of detecting and preventing security breaches in a distributed cloud computing environment. It employs a single controller to handle all of the instances of IDS that are deployed for each user. This architecture supports both signature-based and learning-based approaches to IDS.

## Methodology

Figure 1 depicts a potential testing and evaluation technique for ML-IDPS models. During the first step, the CIC 2017 dataset is collected from open sources that include eight (8) traffic monitoring sessions, each of which is represented as a CSV file. After data collecting, the final datasets for modeling are constructed. Cleaning and transforming different forms of data, as well as separating datasets, is all part of data processing. The model selection follows this step. The next section outlines the criteria used to pick models. Three machine learning methods are used to train the models: logistic regression, LightGBM, and a suggested Random Forest. K-fold cross-validation is used to retrain the model. Finally, the model is put to the test with data that has never been seen before. A variety of measures are used to assess the outcomes.

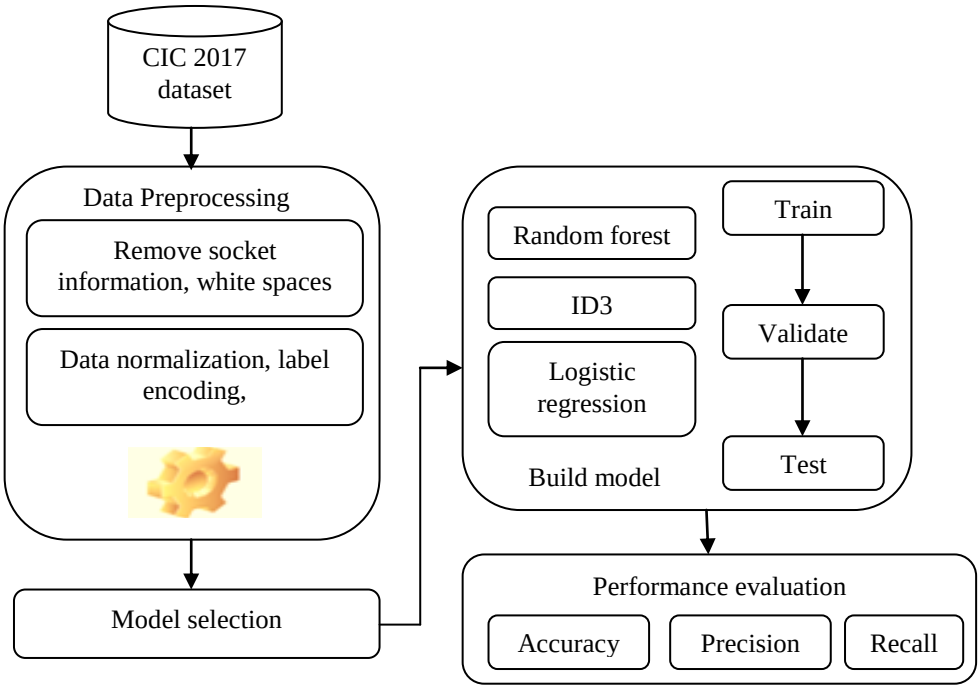


Figure 1. Proposed framework's overall process.

**Dataset:**

It is the most challenging part of IDPS evaluation to choose a suitable dataset. In this part, we'll talk about the IDPS evaluation's primary data collection. Table 1 shows the CICIDS-2017 dataset utilized in this study, which includes these features.

Table 1. Dataset description

| Type of parameter | Specifications |
|---|---|
| Dataset type used | Multi-class |
| Feature count | 80 |
| Number of Classes | 15 |
| Establishment year | 2017 |
| Length of the capture | 5 days |
| Attack model with Infrastructure | 1 Switch, 1 Router 4 PCs |
| Infrastructure victim | 10 PCs, , 1 Firewall, 2 Switches, 3 Servers |

**Data Pre-processing:**

When it comes to reducing the amount of data that may be analyzed without sacrificing any information, data pre-processing is the most common method. IDS computation data is optimized and efficient and false detection rates are reduced and detection rates are increased. Currently, we are using Machine Learning in our studies. ISCX consortium CICIDS-2017 dataset uses CSV data in this format. Machine learning CSV contains eight (8) unique comma-separated traffic monitoring sessions. The file contains "BENIGN" traffic as well as "Attacks," which are characterized as "ABNORMALITY." During normalization, the noise values such as zeros and mean values are replaced by null or infinity symbols. Secondly, a normalization phase is required since CICIDS-2017 contains a considerable quantity of undistributed histogram data for a number of features. The following formulae are used to bring all of the attribute values to the same scale as the normalization technique:

X= −3 value of minimum

$$i = \frac{x - \mu}{\sigma} \quad (1)$$

In equation ( 1) si denotes the standardization

Mean, μ is denotes as,

$$\mu = \frac{1}{N}\sum_{i=1}^{N}(xi) \quad (2)$$

and standard deviation, σ denotes as,

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2} \quad (3)$$

The pre-processing results are a set of 38 features with values in the [-3, 3] range. It was decided to use this range to increase the spread of data and optimize training outcomes. CICIDS 2017 also includes a string name for each class label. To learn the classifier how many classes each tuple belongs to, these values must be encoded into numerical values. These multiclass labels are used just for this method since binary ones already exist in this form.

Feature Importance and Ranking: After scanning and preparing the data, we determine the value of each feature and classify them accordingly to choose a subset for further processing. An intrusion detection model may be built for each feature in the CICIDS 2017 dataset based on the feature's relevance. All non-zero relevant features were retained, except for those in which MDI revealed zero importance because of a large number of different label classes (15).

**Model selection:**

The development stages may also contain other aspects, such as strategic or technical considerations. Machine-learning algorithms may be selected using a range of activities and decisions, as shown in Figure 2. We're focusing on decision-making algorithms in this study. Three commonly used machine learning methods are used to assess the accuracy of general IDS in the CICIDS-2017 dataset. Threefold is the result of our effort. It is important to begin by highlighting the decision-making point in this whole process, as well as outlining each phase in machine learning algorithm evaluation.

**Models Selection for Classification:**

We split the classification process into two parts: (1) the development of the learning method, and (2) the production of the anticipated labels based on the learning approach. These activities are carried out with the help of Scikit-learn, a Python programme for machine learning, data analysis, and computer vision. In cybersecurity, machine learning (ML) techniques, which may be used to identify irregularities with a high success rate, have been shown in figure 2 efficient. Various classification methods were evaluated and trained in this study, including Logistic regression, ID3, and Random forest. Based on the following criteria: (1) a parameter and non-parameter mix; (2) a different model and (3) the common feature of algorithms in previous works, the following algorithm sets are selected:

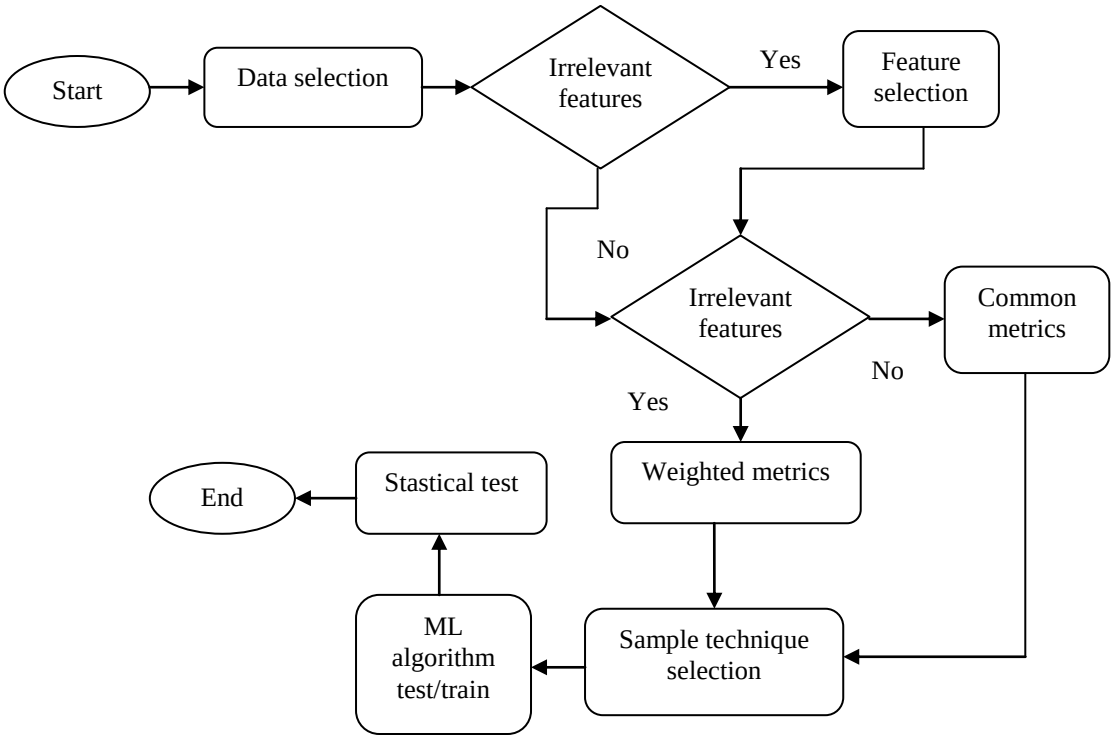


Figure 2. Decision-Making actions using ML Algorithm Selection

**Logistic Regression:**

A sort of predictive analysis known as logistic regression is most effective when dealing with a binary variable. One dependent binary and other non-binary variable are shown to be linked using logistic regression. A logistical function is applied to the training set, and thus predicts the probability of all data points. An algorithm known as the logistic function is used to explain how a classification system grows over time. S-shaped curve can take any real values and convert them to 0 and 1, but never precisely at the boundaries.

**Enhanced ID3 algorithm (Proposed model):**

This work employs the distance metric rule, which states that within a data set, instances with similar properties will exist near other instances with similar properties if it matched them in the same cluster, otherwise in another cluster. It classifies data by spanning a tree structure and asking appropriate requests about the data's properties. Whenever the traversal reaches a leaf node, the data point is classified based on the class in the leaf node, and its pseudo-code is explained in Algorithm 1 in Table 2 through figure 3.

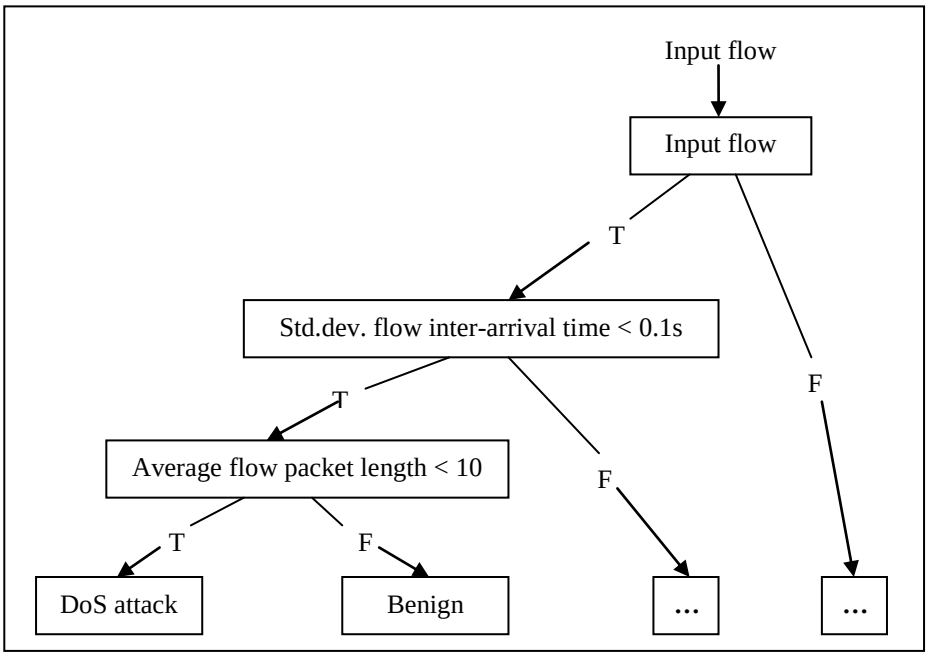


Figure 3. Flow-based traffic categorization decision tree

Table 2. Pseudo code of Algorithm 1for Enhanced ID3 algorithm

**Algorithm 1: Enhanced ID3 algorithm**

**Input:** Features of dataset

**Output:** Decision Tree

1. Load the training data set.
2. If it identified a feature in a dataset, it should be discarded.
3. Split the training set data into subsets based on the distance metric.
4. Determine the distance between (1..... X) Instances in the provided dataset.
5. if (DEud > 55% && DEud 70%)

The instance is then added to the new set and discarded from the existing dataset if it belongs to the same group.

**Otherwise**

**Do nothing**

6. Repeat steps 4 and 5 for each occurrence until no more matches are discovered.
7. Recursively apply the ID3 algorithm to each subgroup.

If all instances at the target feature are positive, return a single tree root with the label positive.

If all instances at the target feature are positive, return a single tree root with the label negative.

The single node tree root with a label equal to the most frequent value of the target feature in the instances is returned if there are no more predicting features than there are examples.

**Otherwise**

- Compute the entropy of the decision node; if entropy is greater than zero, compute the information gain (IG) for each feature.
- For spitting, each chooses the value with the maximum information gain.
- Apply the procedure recursively until the entropy of each attribute does not reach 0.

**End**

## Random forest algorithm

A supervised learning algorithm, the Random Forest (RF) is a kind of this. Random Forests uses a series of decision trees to categorise data. Individual trees may be constructed using Algorithm 2 in Table 2. Ensemble algorithms benefit from the use of multi-models, as previously stated. Using a random grouping of individual samples, the Random Forest method is used to train every Decision Tree. The majority of votes are cast by individuals in the collection of trees throughout the classification process. The Random Forest is made up of many trees, all of which are identical. The tree develops as long as it is possible to make a tree with a variable x and a variable y. separated from all other trees in the area, it's possible to infer the presence of new growth by looking at the surrounding trees. Tree x is the only option. Each node in a Random Forest is given a random number of children, resulting in a higher degree of accuracy. Root, internal, and/or leaf nodes may all be constructed using this method by randomly picking data characteristics. The root node is the node at the top of the decision tree. There are at least two or more outputs and a single input in the internal node. Leaves or terminal nodes are those that have single input with zero output.

Table 3. Random Forest Pseudo code

| **Algorithm 2 : Random Forest** |
|---|
| **Input**: $(a, b)^K$ |
| $K$ : Instances with L features |
| $n$ : number of labeling instances |
| $I$ : Iteration count |
| **begin** |
| **for** $x \in \{1,\ldots\ldots,X\}$ |
| **do** $d_x \leftarrow 0$ |
| **for** $x \in \{1,\ldots\ldots,X\}$ |
| **do** |
| **for** i $\in \{1,\ldots\ldots,I\}$ |
| **do** |
| $a_1 \leftarrow$ Arbitrary instances from $a^K$ |
| $w \leftarrow n\,(x_1)$ |
| $a_2 \leftarrow$ Arbitrary instances from $a^K$ |
| $a_1\,(x) \leftarrow a_2\,(x)$ |
| **if** $n(\mathrm{a}_1) \neq w$ |
| **then** |
| $d_x \leftarrow d_x + 1$ |

To predict classification results that are more accurate and error-resistant, an RF is built utilizing several different DTs. Majority-based voting is used to train random-built DTs. There are two distinct classification strategies, one that utilizes the whole set of member DTs to build a rule subset and the other that uses a training set of rules to cluster new samples after the training period. As a result, the output is powerful and accurate, and it is resistant to over fitting.

**Self-Taught Learning for better accuracy:**

There are two phases to the self-taught learning process known as Self-Taught Learning (STL). xu, a large collection of unlabeled data, is used to develop an adequate feature representation in the first stage, known as Unsupervised Feature Learning (UFL). This freshly learnt representation is then used to categorise labelled data (xl). Data that is both labelled and unlabeled might come from a number of sources, but there must be a link between them. Figure 3 illustrates STL's architectural flowchart. UFL may be implemented using a variety of methods, including Sparse Autoencoding, Restricted Boltzmann Machines (RBMs), K-Means Clustering, and Gaussian Mixtures.

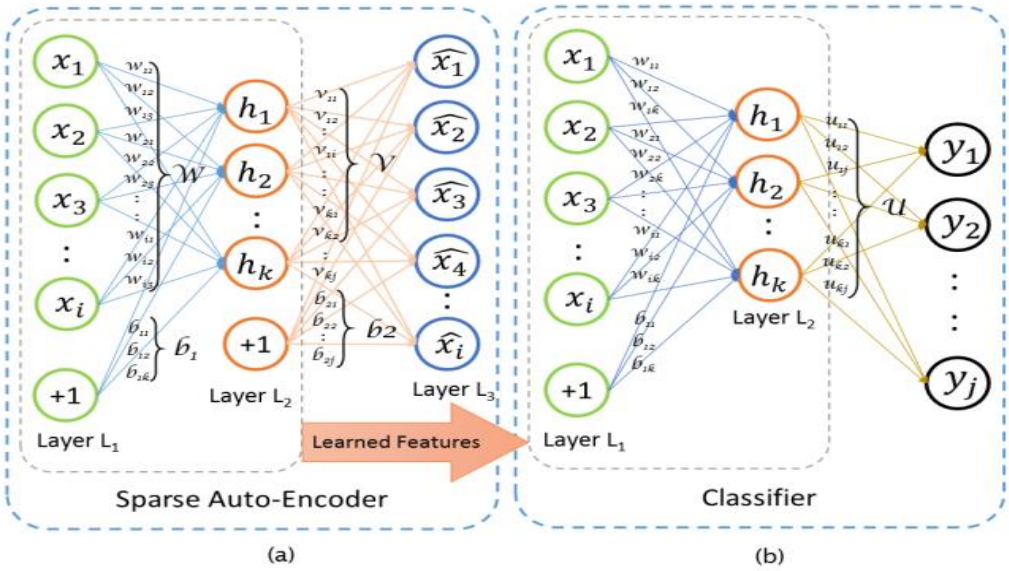


Figure 4. The two-stage process of self-taught learning: a) An unlabeled dataset used for Unsupervised feature learning (UFL). b) Labeled Data classification

The input, hidden layer, and output layer of a sparse auto encoder are all three levels of a neural network. Hidden layer has K nodes, whereas input and output layers have N nodes. According to Figure 4 (a), target values at the output layer are assigned to the input values, i.e., $\hat{x}_i = x_i$. While trying to learn the approximation of output $\hat{x}_i$ similar to x, the sparse autoencoder network uses the back-propagation algorithm to discover the ideal values for weight matrices [16].

$$hw, b^{(x)} = g\,(Wx + b) \tag{5}$$

$$J = \frac{1}{2m}\sum_{i=1}^{m}\|x_i - \$_i\|^2 + \frac{\lambda}{2}\left(\sum_{k,n} W^2 + \sum_{n,k} V^2 + \sum_{k} b_x^2 + \sum_{n} b_x^2\right) + \beta\sum_{j=1}^{K} KL\left(\rho P\rho\$_j\right)KL\left(\rho P\rho\$_j\right)$$
$$= \rho\, log\frac{\rho}{\rho\$_j} + (1-\rho)log\frac{1-\rho}{1-\rho\$_j} \tag{6}$$

Back-propagation in sparse auto encoder using uses Eq.(6) as the cost function to minimise. M input data sum-of-square error terms are averaged in the first term. In order to avoid over-fitting in training, an extra weight decay term with a weight decay parameter is utilized. Equation (7)'s sparsity penalty component, expressed as the KullbackLeibler (KL) divergence, forces the hidden layer to maintain low average activation values:

$$\mathrm{KL}(\rho \mathrm{P} \rho \$_{\mathrm{j}}) = \rho \log \frac{\rho}{\rho \$_{\mathrm{j}}} + (1-\rho) \log \frac{1-\rho}{1-\rho \$_{\mathrm{j}}} \tag{7}$$

A sparsity constraint parameter called ρ may be anywhere from 0 to 1, & β controls the sparsity penalty term. Classification in the second stage is carried out using a new features representation (a) and the labels vector (y). Classification is done by using a soft-max regression model, as illustrated in Figure 3b.

## Results and Discussion

A cloud-based intrusion detection and prevention system is discussed and analysed in this section. On a 64-bit Windows 10 PC running Windows 7 @ 2 GHz dual-core with 4 GB main RAM, we built and tested an intrusion detection & prevention system in Python. Precision, recall, and accuracy are used to assess a proposed system. CICIDS-2017 is the dataset used in our proposed method.

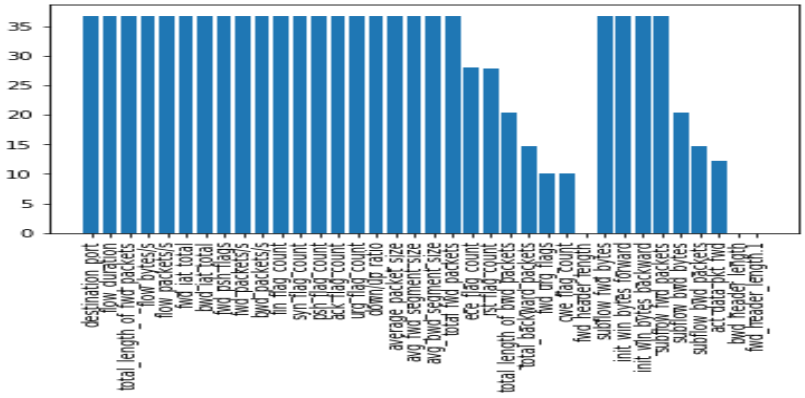


Figure 5. Plotting feature significance

From Figure 5, it is clear that the fwd_header_length,bwd_header_length andfwd_header_length1 is having zero feature significance, so that can be dropped in the feature selection process to reduce the dimensionality.

**Exploring with KMeans & PCA:**

Despite the fact that we've reduced the number of features from 45 to 15, we'd still utilise them all. After that, we'll be able to test their performance by putting them into various models. The unbalanced dataset that results from using 15 features is a major drawback. Clustering and PCA analysis will be used to investigate the data further.

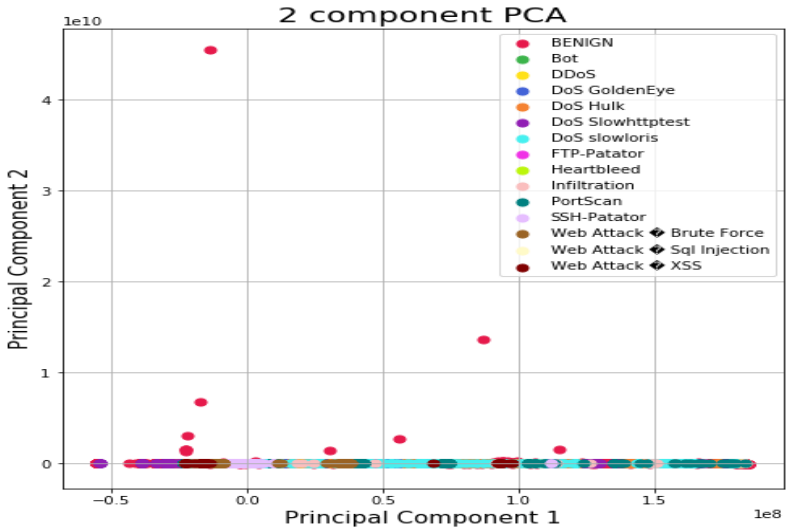


Figure 6. Clustering and PCA analysis

If we look at figure 6, we can see that the kmeans algorithm has a difficult time assigning clusters since there is a lot of overlap. Both infiltration and regular network activity include features that cannot be separated, which may be due to poor feature selection or the inability to distinguish between the two.

**Performance Metrics:**

For the IDPS evaluation, the feature selection algorithm's selection accuracy and the machine learning algorithm's accuracy are both considered. It is possible to assess IDPS performance in a number of ways. There are three subcategories in this measure: threshold, ranking, and probability metrics. The effectiveness of our technique is assessed using execution measures like as precision, recall, and F1 Score. It is possible to split test data based on how well the classification was performed using a confusion matrix.

### 4.1.1 Confusion Matrix:

The confusion matrix may be understood as a tool for determining whether or not a classifier is effective in distinguishing between tuples belonging to various classes. While the classifier is incorrect in classifying the data, the True-Negative (TN) and True- Positive values give the correct classifier data. In addition to evaluating a dataset's performance, the following measures are employed:

Accuracy is a highly prevalent statistic in classification tasks, and it seeks to represent the fraction of correctly classified samples across the overall classification context:

$$\text{Accuracy} = \frac{\text{Count of predicted correctly}}{\text{Total count of predictions}} \tag{8}$$

This measure is an excellent indication of how well a classifier is doing. Due to uneven datasets, such as the intrusion detection datasets, the majority class is given preference in this

metric (Benign traffic). Consequently, if 90% of traffic was benign and the classifier failed to identify every attack but recognized all other traffic, it would still have a 90% accuracy, which would just mask the model's performance results.

Precision is defined as the proportion of correct positives (TP) to total positives (TP+FP)., i.e.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{9}$$

Intrusion detection uses this ratio to determine how many attacks were accurately anticipated and how many were incorrectly identified. In general, precision is preferable than accuracy since it is not prejudiced to the majority class.

Remembering a true positive (TP) is the percentage of all positive samples (TP + FN) that were properly anticipated.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{10}$$

**Assessment Efficiency:**

In this experiment, we report on the results of a machine learning-based cyber security intrusion detection system. We constructed the model first, then tested it on the remaining 20% of the dataset to see how it performed in the real-world events we encountered throughout the testing process. The outcomes are computed by creating a confusion matrix that provides the false positive rate, false negatives, true positives, and true negatives. Table 3 illustrates the predicted outcomes in terms of precision, recall, & accuracy for each class, either normal or abnormality, based on these factors, to demonstrate the findings at the conclusion of the model.

Table 4. Comparison of testing metrics with three ML models

| Model | Precision in % | Recall in % | Accuracy in % |
|---|---|---|---|
| **Linear Regrssion** | 81.23 | 71.44 | 71.23 |
| **Enhanced ID3 Decision Tree (Our work)** | 99.12 | 99.64 | 99.85 |
| **Random forest** | 98.30 | 98.12 | 98.11 |

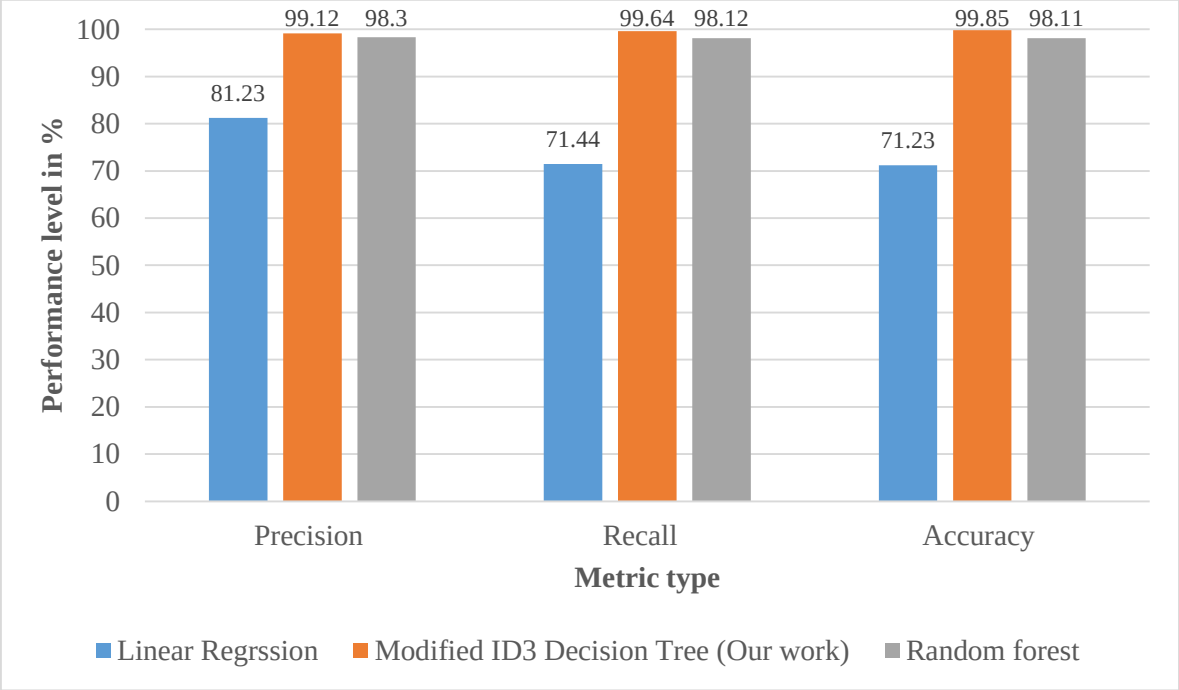


Figure 7. Performance comparision of three testing metrics (Precision, Recall and Accuracy) with three ML models

Figure 7 proves that the proposed ID3 outperforms the existing model's Logistic regression (LR), Random forest (RF) with Precision = 99.12%, Recall = 99.64%, and Accuracy = 99.85%. The rationale behind it is that, in our ID3 model, initially identify the essential features before building the intrusion detection and prevention framework based on the best-selected features. As a consequence, the model's variance and over-fitting issues are reduced, and the model's important traits are taken into account, making it more universal. For previously unknown test-cases, this means the model may enhance prediction results and lower the computational complexity of any data-driven security model that results. As a result of the results presented is too the aforementioned experimental investigation, we can infer that our ID3 model outperforms existing machine learning classification-based techniques when interpreting data.

## Conclusion

An intrusion detection and prevention system (IDPS) that uses machine learning methods to identify and classify network threats are presented in this paper. During the investigation, it was discovered that the underlying dataset for IDPS detection has to be updated to better identify new attacks. As a result, attackers use a wide range of strategies and technology to execute out these attacks. The goal of the CIC-IDS-2017 dataset was to provide an intrusion detection dataset that included realistic network traffic and current network assaults. In studies including feature subsets, the ID3 classifier outperformed the competition. For ID3-based network intrusion detection systems (NIDS), feature selection (FS) is used to increase their performance on the CICIDS-2017 Dataset. Using our suggested technique, the Accuracy is

99.85%, the Precision is 99.12%, and Recall is 99 percent and the F1-Score is 99.64%. Stacked Auto encoder, a deep belief network version of sparse auto encoder, may be used for unsupervised feature learning to increase speed even more.

## Conflict of interest

The authors declare no potential conflict of interest regarding the publication of this work. In addition, the ethical issues including plagiarism, informed consent, misconduct, data fabrication and, or falsification, double publication and, or submission, and redundancy have been completely witnessed by the authors.

## Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article

## References

Albahar, M. A., Binsawad, M., Almalki, J., El-etriby, S., & Karali, S. (2020). Improving Intrusion Detection System using Artificial Neural Network. International Journal of Advanced Computer Science and Applications, 11(6).

Al-Shourbaji, I., & Al-Janabi, S. (2017). Intrusion Detection and Prevention Systems in Wireless Networks. Kurdistan Journal of Applied Research, 2(3), 267-272.

Amini, M., Jalili, R., & Shahriari, H. R. (2006). RT-UNNID: A practical solution to real-time network-based intrusion detection using unsupervised neural networks. computers & security, 25(6), 459-468.

Chen, L., Xian, M., Liu, J., & Wang, H. (2020). Intrusion detection system in cloud computing environment. In 2020 International Conference on Computer Communication and Network Security (CCNS) (pp. 131-135). IEEE.

Dastanpour, A., Ibrahim, S., & Mashinchi, R. (2016). Effect of genetic algorithm on artificial neural network for intrusion detection system. International Journal of Computer Sciences and Engineering, 4(10), 10-18.

Garbis, J., & Chapman, J. W. (2021). Intrusion Detection and Prevention Systems. In Zero Trust Security (pp. 117-126). Apress, Berkeley, CA.

Improving Network Security Based on Trust-Aware Routing Protocols Using Long Short-Term

Jasim, Y. A. (2018). Improving intrusion detection systems using artificial neural networks. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal, 7(1), 49-65.

Kiran Kumar, M., Kranthi Kumar, S., Kalpana, E., Srikanth, D., & Saikumar, K. (2022). A Novel Implementation of Linux Based Android Platform for Client and Server. In A Fusion of Artificial Intelligence and Internet of Things for Emerging Cyber Systems (pp. 151-170). Springer, Cham.

Labib, K., & Vemuri, R. (2002). NSOM: A real-time network-based intrusion detection system using self-organizing maps. Networks and Security, 21(1).

Maithili, K., Vinothkumar, V., & Latha, P. (2018). Analyzing the Security Mechanisms to Prevent Unauthorized Access in Cloud and Network Security. Journal of Computational and Theoretical Nanoscience, 15(6), 2059–2063. doi:10.1166/jctn.2018.7407

Mehmood, Y., Shibli, M. A., Habiba, U., & Masood, R. (2013). Intrusion detection system in cloud computing: challenges and opportunities. In 2013 2nd national conference on information assurance (NCIA) (pp. 59-66). IEEE.

Memory-Queuing Segment-Routing Algorithms. International Journal of Information

Moloja, D., & Mpekoa, N. (2017). Towards a cloud intrusion detection and prevention system for m-voting in south africa. In 2017 International Conference on Information Society (i-Society) (pp. 34-39). IEEE.

Muthukumaran V., Kumar, V. V., Joseph, R. B., Munirathanam, M., & Jeyakumar, B. (2021). Improving Network Security Based on Trust-Aware Routing Protocols Using Long Short-Term Memory-Queuing Segment-Routing Algorithms. International Journal of Information Technology Project Management, 12(4), 47–60. doi:10.4018/ijitpm.2021100105

Muthukumaran V., Kumar, V. V., Joseph, R. B., Munirathanam, M., & Jeyakumar, B. (2021).

Naoum, R. S., Abid, N. A., & Al-Sultani, Z. N. (2012). An enhanced resilient backpropagation artificial neural network for intrusion detection system. International Journal of Computer Science and Network Security (IJCSNS), 12(3), 11.

Nazarzadeoghaz, N., Khendek, F., & Toeroe, M. (2020). Automated design of network services from network service requirements. In 2020 23rd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN) (pp. 63-70). IEEE.

Puttini, R. S., Marrakchi, Z., & Mé, L. (2003). A Bayesian Classification Model for Real- Time Intrusion Detection. In AIP Conference Proceedings (Vol. 659, No. 1, pp. 150-162). American Institute of Physics.

Raina, R., Battle, A., Lee, H., Packer, B., & Ng, A. Y. (2007). Self-taught learning: transfer learning from unlabeled data. In Proceedings of the 24th international conference on Machine learning (pp. 759-766).

Sabhnani, M., & Serpen, G. (2003). Application of machine learning algorithms to KDD intrusion detection dataset within misuse detection context. In MLMTA (pp. 209-215).

Saikumar, K., Rajesh, V., & Babu, B. S. (2022). Heart Disease Detection Based on Feature Fusion Technique with Augmented Classification Using Deep Learning Technology. Traitement du Signal, 39(1).

Technology Project Management, 12(4), 47–60. doi:10.4018/ijitpm.2021100105

**Bibliographic information of this paper for citing:**

Srilatha D. & Thillaiarasu N. (2023). Implementation of Intrusion detection and prevention with Deep Learning in Cloud Computing. *Journal of Information Technology Management,* 14 (Special Issue), 1-18. https://doi.org/ 10.22059/jitm.2022.89407