

Intrusion Detection using Machine learning

Manasa U¹ and Dr. Kuzhalivaimozhi S²

¹⁻²Department of Information Science and Engineering, The National Institute of Engineering, Mysore Karnataka, India.
Email: 2021scn_manasa_a@nie.ac.in, kuzhali_mozhi @nie.ac.in

Abstract— The fast development in the use of computer networks raises concerns about network availability, integrity, and confidentiality. This requires network managers to use various types of intrusion detection systems (IDS) to monitor network traffic for unauthorized and malicious activity. An intrusion is a malicious breach of security policy. As a result, an intrusion detection system monitors network traffic flowing through computer systems to look for malicious actions and recognized dangers, providing alarms when it detects them. Mining approaches can be highly useful in the development of an intrusion detection system. In order to detect intrusions, network traffic might be categorized as normal or anomalous. After examining over twenty-five studies, we chose the top three classification algorithms in our paper: Logistic Regression, Naive Bayes, and SVM. This study compares the top three classification algorithms based on their performance criteria to determine the best suited method available.

Index Terms— Intrusion, Cyber-attacks, machine learning.

I. INTRODUCTION

The rising usage of computers and accessibility of access to the internet has expanded the number of techniques to attack and fool a system. In simple terms, intrusion is the criminal act of invading, seizing, or taking possession of another's property (in this case, the computer system) [1]. Although network-based systems can be protected by a variety of security methods such as information encryption, access control, and intrusion prevention, there are still numerous unreported intrusions. Firewalls, for example, cannot protect internal attacks. As a result, intrusion detection systems (IDSs) are critical to network security.

The widespread use of the internet in recent years has resulted in an exponential increase in the volume of information transmitted among various devices, as well as the number of novel methods of network assault [2]. Traditional approaches, such as firewalls, which focus on data filtering, may not be suitable for detecting all sorts of assaults in real time. Intrusion detection systems (IDS) based on machine learning algorithms are particularly useful in effectively processing the enormous volume of data for recognizing any malicious activity for effective handling and prompt identification of these types of assaults. IDS based on machine learning are used to monitor all network events for malicious behavior [3].

An intrusion detection system (IDS) is used to protect data integrity, confidentiality, and system availability from malicious attacks. In general, intrusion detection does not involve intrusion prevention. There are two types of intrusion detection techniques: anomaly detection and abuse detection. Anomaly detection detects assaults by detecting patterns that differ from usual behaviour [4]. Misuse detection identifies attacks by analysing patterns collected from known invasions. While accuracy is a must for an IDS, extensibility and adaptability are equally vital in today's network computing environment. The main issue with current intrusion detection systems is that they do not generalise to identify zero-day attacks or attacks with unknown signatures [5]. Anomaly detection

systems are adaptive in nature, meaning that they can deal with new attacks but cannot identify the precise sort of attack [6-8]. As a result, numerous supervised and unsupervised algorithms for intrusion detection have been developed.

The paper is divided into four pieces, the first of which is an introduction to the article's topic. The second portion discusses the summary of the research articles, which mentions several ML techniques utilised for building IDS. The third section describes the proposed work's approach, which includes the use of classification algorithms and datasets. Finally, the fourth and fifth sections presented the debate, conclusion, and considerations for future research effort in ML-based intrusion detection systems.

II. RELATED WORK

A comparison of all rule-based and some function-based classifiers in order to forecast their performance in terms of accuracy, sensitivity, specificity, time, and error [9]. Introduced a three-tier intrusion detection system design that includes a blacklist that filters out known assaults from traffic and a whitelist that detects regular traffic. A multi-class SVM classifier was used to classify the remainder traffics, which were discovered as anomalies by the whitelist. Wei et al [10] demonstrated the importance of each feature in the KDD '99 intrusion detection dataset for class classification.

Gao et al. [11] proposed an ensemble machine learning intrusion detection system. They extracted features using the Principal Component Analysis approach. Their ensemble technique combines Decision Tree, Random Forest, KNN, DNN, and Multitree after various testing on the NSL-KDD datasets. The ensemble algorithm's outputs are determined by a majority vote utilizing weights for each method to improve accuracy. They acquired an accuracy of 85.2%, which is higher than the accuracy attained if only one method was used. However, their model is inefficient when analyzing attacks in small numbers.

To detect unknown threats, Jabez et al. [12] developed a system that employs an outlier detection approach. The outlier identification method identifies data points that are apart from grouped points. The neighborhood outlier factor is used in this method to find points that are not close to each other. They ran their solution through its paces on the KDDcup99 datasets. Furthermore, the key benefit of their method is its execution time, which is substantially faster than other alternatives such as back propagation neural networks, which demand a large amount of computer resources.

Xiao et al. [13] offered a CNN-based method. They began by extracting features using both Principal Component Analysis (PCA) and Auto-Encoder (AE). Auto-Encoder is a dimension reduction technology that removes inconsequential data by using numerous hidden layers of neural networks. They then converted the data dimension from one to two dimensions and fed it into the CNN model to train it. Back propagation methods are used to train and improve the model.

To improve IDS, Khan et al. [14] suggested a two-stage deep learning model (TSDL). In the first stage, they assign a probability value to the traffic and label it as normal or abnormal. They employed a DNN technique for both stages, with a Deep stacked auto-encoder (DSAE) for feature extraction and Soft-max as a classifier, and they used this value as an additional feature to train the classifier in the second step. For multi-class classification issues, soft-max is frequently utilized in a neural network. They put their solution through its paces on the KDDcup99 and UNSW-NB15 datasets.

Kuttranont et al. [17] proposed a KNN-based detection method and parallel computing algorithms that run on a graphics processing unit (GPU). They altered the KNN algorithm's neighbor-selection rule. The normal KNN chooses the top K closest samples as neighbors, but the improved algorithm chooses a preset fraction (say, 50%) of the neighboring samples as neighbors. The proposed method takes into account the unevenness of data distribution and works well with sparse data.

III. METHODOLOGY

Fig 1 depicts the basic procedure of this study. First, the training dataset's data is pre-processed and delivered into the classifiers. The test dataset is used to evaluate the performance of the classifiers once they have been trained. To train the machine learning algorithms, four types of pre-processed data were used: traffic, domain, host, and basic. The outcomes and Discussion section contains a final comparison of all of the experimental outcomes.

The proposed model's output has been addressed in this section, where SVM, LR, and NB were utilized to identify the data as normal or intrusive. The model's performance was evaluated using four different feature subsets from the NSL KDD dataset. Subsequent steps summarize the process and are thoroughly detailed in the sections that follow. The data is pre-processed in the first stage. The second stage is to determine the model's significant feature

subsets. As indicated in Fig 2, the third stage is to use numerous Machine Learning classifiers to train and test the data. Finally, the outcomes of the various parameters are examined.

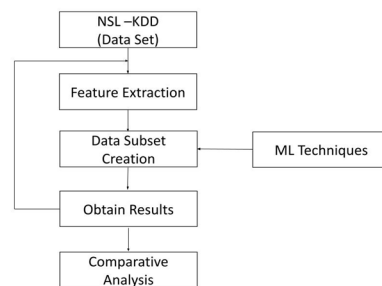


Fig 1 : Flow Chart

A. Dataset

As numerous scholars have pointed out, the KDD'99 dataset has several shortcomings [18-20]. As a result, they proposed the NSL-KDD [21], which condensed the original KDD'99 dataset while making specific modifications. They eliminated redundant and duplicate records from the KDD'99 dataset. The authors also took precautions to avoid model bias by making the amount of records inversely proportional to prediction success. The official dataset is divided into two parts: a training dataset with 125,973 records and a test dataset with 22,544 records. The dataset includes the 41 original features from the KDD'99 dataset.

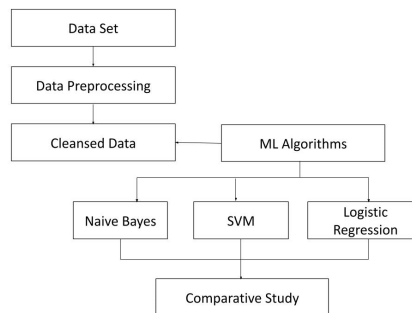


Fig 2 : Architecture

A. Preprocessing

Preprocessing is an important step in the learning process. When the mean and standard deviation of all of the features is compared, there are significant variances between the numeric features, which may result in biased findings when applying machine learning methods [23].

The majority of machine learning algorithms are incapable of processing categorical data features. There are three categorical features in the NSL-KDD dataset: protocol type, service, and flag [24, 25]. These characteristics are critical for categorization. As a result, as part of the data preprocessing stage, we apply one-hot encoding to these features.

B. Machine learning models

Machine learning techniques were chosen for testing against the NSL-KDD dataset to compare performance for the job of intrusion detection. The probabilistic models used the Gaussian Naive Bayes algorithm; the distance-based models used SVM; and the regression models used logistic regression with L1 and L2 penalties. All of these classifiers were built with the sci-kit learn package and the entire set of parameters.

a. Support vector machine

In SVMs, the goal is to find a maximum-margin separating hyperplane in the n-dimensional feature space [26]. Because the separation hyperplane is determined by a small number of support vectors, SVMs can produce satisfying results even with small-scale training data. SVMs, on the other hand, are sensitive to noise around the hyperplane. SVMs excel at solving linear problems. Kernel functions are commonly employed with nonlinear

data. A kernel function transforms the original space into a new space, allowing the nonlinear data to be separated. Kernel techniques are common in SVMs and other machine learning algorithms.

b. Naïve Bayes

The Nave Bayes algorithm is based on the conditional probability and the attribute independence hypothesis [27]. The Nave Bayes classifier computes the conditional probabilities for several classes for each sample. The sample is assigned to the class with the highest probability. The Nave Bayes method finds the best solution when the attribute independence hypothesis is satisfied. Unfortunately, that hypothesis is difficult to satisfy in practice, and thus the Nave Bayes algorithm performs poorly on attribute-related data.

c. Logistic Regression

The LR is a logarithmic linear model. The LR algorithm uses parametric logistic distribution to compute the probabilities of various classes. An LR model is simple to build, and model training is quick. However, LR struggles with nonlinear data, limiting its application.

IV. RESULTS AND DISCUSSIONS

In the current study, features from NSL-KDD were chosen at random to reduce the dimension of data depending on which model was trained/tested. Based on the results, this strategy performed well in the current model, albeit it may be unfavorable in certain cases. Several classifiers, including SVM, LR, and NB, were trained and tested on 111386 and 37129 instances of NSL-KDD, respectively, and the model performance was evaluated using various parameters. Figures below show a graphical depiction of the results on several feature subsets in the form of an accuracy and confusion matrix.

Fig 3a shows the confusion matrix obtained by Logistic regression model in training various data subset such as Domain, host, traffic, basic and all dataset together. The accuracy obtained by logistic regression for various data subsets are depicted in Table1.

TABLE 1 : LOGISTIC REGRESSION ACCURACY

Data subsets	Accuracy in %
Domain	82.39
Traffic	82.39
Host	91.38
Basic	17.91
All	87.56

Fig 3b shows the confusion matrix obtained by Gaussian Naive Bayes model in training various data subset such as Domain, host, traffic, basic and all dataset together. The accuracy obtained by Gaussian Naive Bayes for various data subsets are depicted in Table 2.

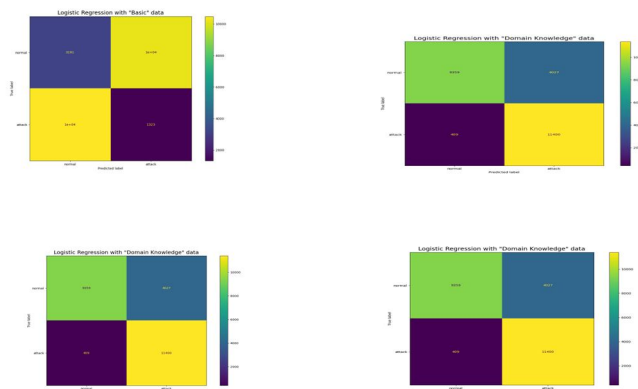


Fig 3a: Confusion matrix Logistic Regression

TABLE II : GAUSSIAN NAÏVE BAYES ACCURACY

Data subsets	Accuracy in %
Domain	48.14
Traffic	86.80
Host	89.43
Basic	53.04
All	90.33

Fig 3c shows the confusion matrix obtained by Support vector machine model in training various data subset such as Domain, host, traffic, basic and all dataset together. The accuracy obtained by support vector machine for various data subsets are depicted in Table 3.

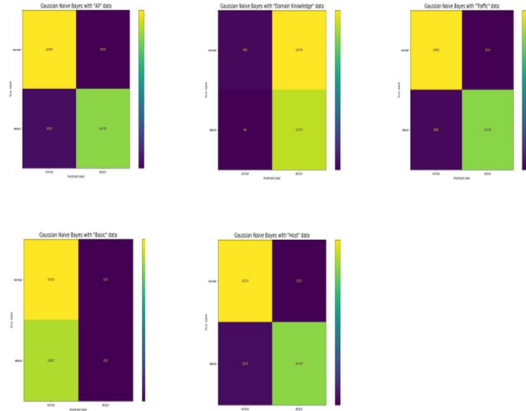


Fig 3b: Confusion matrix Gaussian Naive Bayes

The accuracy comparison of all the three models such as SVM, LR, NB for five categories of data subset has been depicted in Fig 4. Among these three classifiers SVM performed well by giving highest accuracy for all categories of data subset as shown in Table 3.

TABLE III: SVM ACCURACY

Data subsets	Accuracy in %
Domain	82.54
Traffic	88.83
Host	88.83
Basic	80.59

V. CONCLUSION

As the quantity and intensity of network attacks has increased in recent years, intrusion detection systems (IDS) have become an increasingly important component of network security. Because of the vast volumes of security audit data as well as the complex and dynamic aspects of intrusion behaviors, optimizing the performance of intrusion detection systems has become an important open challenge that is attracting increasing attention from

the research community. The Support Vector Machine (SVM) is one of the best machine learning methods for classifying anomalous activity among the several Intrusion detection technologies.

The support vector machine is used in many intrusion detection systems. They are, nevertheless, computationally quite demanding. Dimension reduction strategies are used to extract key features from a given dataset in order to overcome this difficulty. The following two contributions are made by this study. First, this paper includes a review of current trends in intrusion detection using SVM, as well as a study of technologies used by various researchers in this field. Second, it suggests a novel method for selecting the optimal feature for detecting infiltration. In the future, we will aim to improve the work by developing hybrid models.

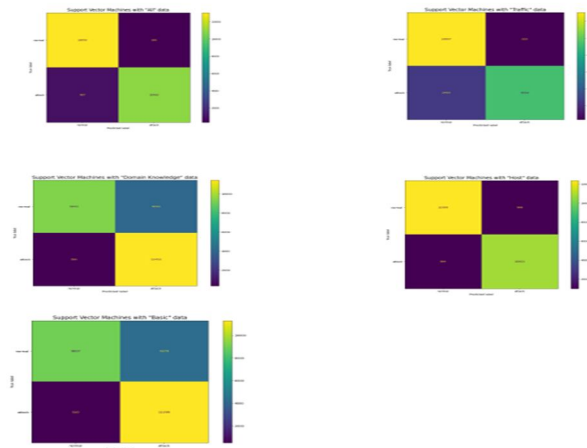


Fig 3c: Confusion matrix of SVM

Fig 3 : Confusion matrix a) Logistic regression b) Naive bayes c) SVM

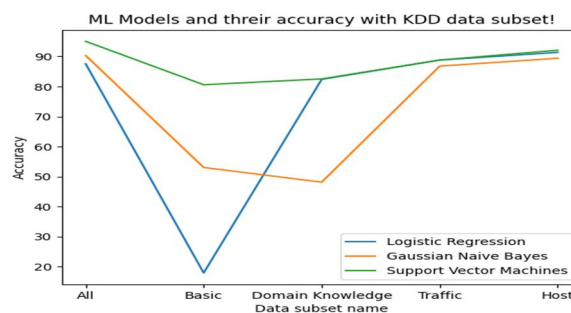


Fig 4 : Accuracy comparison of three models

REFERENCES

- [1] Bhuyan, M.H., Bhattacharyya, D.K., Kalita, J.K.: Towards generating real-life datasets for network intrusion detection. *IJ Network Security*. 17(6), 683– 701 (2015)
- [2] Liao, Hung-Jen, Chun-Hung Richard Lin, Ying-Chih Lin, and Kuang-Yuan Tung. "Intrusion detection system: A comprehensive review." *Journal of Network and Computer Applications* 36, no. 1 (2013): 16-24.
- [3] Khraisat, Ansam, Iqbal Gondal, Peter Vamplew, and Joarder Kamruzzaman. "Survey of intrusion detection systems: techniques, datasets and challenges." *Cybersecurity* 2, no. 1 (2019): 1-22.
- [4] Hoque, Mohammad Sazzadul, Md Abdul Mukit, and Md Abu Naser Bikas. "An implementation of intrusion detection system using genetic algorithm." *arXiv preprint arXiv:1204.1336* (2012).
- [5] Heady, Richard, George Luger, Arthur Maccabe, and Mark Servilla. The architecture of a network level intrusion detection system. No. LA-SUB-93-219. Los Alamos National Lab.(LANL), Los Alamos, NM (United States); New Mexico Univ., Albuquerque, NM (United States). Dept. of Computer Science, 1990.
- [6] Ilgun, Koral. "USTAT: A real-time intrusion detection system for UNIX." In *Proceedings 1993 IEEE Computer Society Symposium on Research in Security and Privacy*, pp. 16-28. IEEE, 1993.
- [7] Wei, P.; Li, Y.; Zhang, Z.; Hu, T.; Li, Z.; Liu, D. An Optimization Method for Intrusion Detection Classification Model Based on Deep Belief Network. *IEEE Access* 2019, 7, 87593–87605.

- [8] Gao, X.; Shan, C.; Hu, C.; Niu, Z.; Liu, Z. An Adaptive Ensemble Machine Learning Model for Intrusion Detection. *IEEE Access* 2019, 7, 82512–82521.
- [9] Jabez, J.; Muthukumar, B. Intrusion Detection System (IDS): Anomaly Detection Using Outlier Detection Approach. *Procedia Comput. Sci.* 2015, 48, 338–346.
- [10] Xiao, Y.; Xing, C.; Zhang, T.; Zhao, Z. An Intrusion Detection Model Based on Feature Reduction and Convolutional Neural Networks. *IEEE Access* 2019, 7, 42210–42219.
- [11] Khan, F.A.; Gumaei, A.; Derhab, A.; Hussain, A. A Novel Two-Stage Deep Learning Model for Efficient Network Intrusion Detection. *IEEE Access* 2019, 7, 30373–30385.
- [12] Meng, W.; Li, W.; Kwok, L.F. Design of intelligent KNN-based alarm filter using knowledge-based alert verification in intrusion detection. *Secur. Commun. Netw.* 2015, 8, 3883–3895.
- [13] Zhang, B.; Yu, Y.; Li, J. Network Intrusion Detection Based on Stacked Sparse Autoencoder and Binary Tree Ensemble Method. In *Proceedings of the 2018 IEEE International Conference on Communications Workshops (ICC Workshops)*, Kansas City, MO, USA, 20–24 May 2018; pp. 1–6.
- [14] Kuttranont, P.; Boonprakob, K.; Phaudphut, C.; Permpol, S.; Aimtongkhamand, P.; KoKaew, U.; Waikham, B.; So-In, C. Parallel KNN and Neighborhood Classification Implementations on GPU for Network Intrusion Detection. *J. Telecommun. Electron. Comput. Eng.* 2017, 9, 29–33.
- [15] Lee, Wenke, and Salvatore J. Stolfo. "A framework for constructing features and models for intrusion detection systems." *ACM transactions on Information and system security (TISSEC)* 3, no. 4 (2000): 227-261.
- [16] Gnanaprasanambikai, L., Munusamy, N.: Data pre-processing and classification for traffic anomaly intrusion detection using NSLKDD dataset. *Cybern Inf Technol.* 18(3), 111– 119 (2018)
- [17] Nasir, Muhammad Hassan, Salman A. Khan, Muhammad Mubashir Khan, and Mahawish Fatima. "Swarm intelligence inspired intrusion detection systems—a systematic literature review." *Computer Networks* (2022): 108708.
- [18] Jiang, Kaiyuan, Wenya Wang, Aili Wang, and Haibin Wu. "Network intrusion detection combined hybrid sampling with deep hierarchical network." *IEEE access* 8 (2020): 32464-32476.
- [19] Rajamohana, S. P., and K. Umamaheswari. "Hybrid approach of improved binary particle swarm optimization and shuffled frog leaping for feature selection." *Computers & Electrical Engineering* 67 (2018): 497-508.
- [20] Abdollahzadeh, Benyamin, and Farhad Soleimanian Gharehchopogh. "A multi-objective optimization algorithm for feature selection problems." *Engineering with Computers* 38, no. Suppl 3 (2022): 1845-1863.
- [21] Eesa, Adel Sabry, Zeynep Orman, and Adnan Mohsin Abdulazeez Brifcani. "A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems." *Expert systems with applications* 42, no. 5 (2015): 2670-2679.
- [22] Aghdam, Mehdi Hosseinzadeh, and Peyman Kabiri. "Feature selection for intrusion detection system using ant colony optimization." *Int. J. Netw. Secur.* 18, no. 3 (2016): 420-432.
- [23] Manzoor, Ishfaq, and Neeraj Kumar. "A feature reduced intrusion detection system using ANN classifier." *Expert Systems with Applications* 88 (2017): 249-257.
- [24] Manzoor, Ishfaq, and Neeraj Kumar. "A feature reduced intrusion detection system using ANN classifier." *Expert Systems with Applications* 88 (2017): 249-257.
- [25] Rajamohana, S. P., and K. Umamaheswari. "Hybrid approach of improved binary particle swarm optimization and shuffled frog leaping for feature selection." *Computers & Electrical Engineering* 67 (2018): 497-508.