JOURNAL OF BASIC SCIENCE AND ENGINEERING

DETECTION OF CYBER ATTACK IN NETWORK USING MACHINE LEARNING TECHNIQUE

[1]Govindu Hymavathi, [2]K.Chandra Prasad
[1]PG Scholar, Department of CSE, Srinivasa Institute of technology and science, Kadapa.
[2]Assistant professor, Department of CSE, Srinivasa Institute of technology and science, Kadapa.
[1]govinduhymavathi10794@gmail.com, [2]chandu.sp2009@gmail.com

ABSTRACT
In imbalanced network traffic, malicious cyber-attacks can often hide in large amounts of normal data. It exhibits a high degree of stealth and obfuscation in cyberspace, making it difficult for Network Intrusion Detection System (NIDS) to ensure the accuracy and timeliness of detection. This paper researches machine learning and deep learning for intrusion detection in imbalanced network traffic. It proposes a novel Difficult Set Sampling Technique (DSSTE) algorithm to tackle the class imbalance problem. First, use the Edited Nearest Neighbor (ENN) algorithm to divide the imbalanced training set into the difficult set and the easy set. Next, use the K-Means algorithm to compress the majority samples in the difficult set to reduce the majority. Zoom in and out the minority samples' continuous attributes in the difficult set synthesize new samples to increase the minority number. Finally, the easy set, the compressed set of majorities in the difficult, and the minority in the difficult set are combined with its augmentation samples to make up a new training set. The algorithm reduces the imbalance of the original training set and provides targeted data augment for the minority class that needs to learn. It enables the classifier to learn the differences in the training stage better and improve classification performance. To verify the proposed method, we conduct experiments on the classic intrusion dataset NSL-KDD and the newer and comprehensive intrusion dataset CSE-CIC-IDS2018. We use classical classification models: random forest(RF), Support Vector Machine(SVM), XGBoost, MLP AlexNet, Mini-VGGNet. We compare the other 24 methods; the experimental results demonstrate that our proposed DSSTE algorithm outperforms the other method.
Keywords: Network Traffic, NSL-KDD, Difficult Set Sampling Technique (DSSTE), Support Vector Machine (SVM).

I. INTRODUCTION

The internet is providing various convenient services for people. However, we are also facing various security threats. Network viruses, eavesdropping and malicious attacks are on the rise, causing network security to become the focus of attention of the society and government departments. Fortunately, these problems can be well solved via intrusion detection. Intrusion detection plays an important part in ensuring network information security. However, with the explosive growth of internet business, traffic types in the network are increasing day by day, and network behavior characteristics are becoming increasingly complex, which brings great

challenges to intrusion detection [1], [2]. How to identify various malicious network traffics, especially unexpected malicious network traffics, is a key problem that cannot be avoided.

By improving the performance of classifiers in effectively identifying malicious traffics, intrusion detection accuracy can be largely improved. Machine learning methods [3]–[8] have been widely used in intrusion detection to identify malicious traffic. However, these methods belong to shallow learning and often emphasize feature engineering and selection. They have difficulty in features selection and cannot effectively solve the massive intrusion data classification problem, which leads to low recognition accuracy and high false alarm rate. In recent years, intrusion detection methods based on deep learning have been proposed successively. In [9], the authors propose a mal-ware traffic classification method based on convolutional neural network with traffic data as image. This method does not need manual design features, and directly takes the original traffic as the input data to the classifier. In [10], the authors provide an analysis of the viability of Recurrent Neural Networks (RNN) to detect the behavior of network traffic by modeling it as a sequence of states that change over time. In [11], the authors verify the performance of Long Short-term memory (LSTM) network in classifying intrusion traffics. Experimental results show that LSTM can learn all the attack classes hidden in the training data. All the above methods treat the entire network traffic as a whole consisting of a sequence of traffic bytes. They don't make full use of domain knowledge of network traffics. For example, CNN converts continuous network traffic into images for processing, which is equivalent to treating traffics as independent and ignore the internal relations of network traffics. Firstly, network traffic is a hierarchical structure. Specifically, network traffic is a traffic unit composed of multiple data packets. Data packet is a traffic unit composed of multiple bytes. Secondly, traffic features in the same and different packets are significantly different. Sequential features between different packets need to be extracted independently. In other words, not all traffic features are equally important for traffic classification in the process of extracting features on a certain network traffic.

II. BACKGROUND WORK

In the research of network intrusion detection based on machine learning, scholars mainly distinguish normal network traffic from abnormal network traffic by dimensionality reduction, clustering, and classification, to realize the identity fiction of malicious attacks [10], [11].

Pervez proposed a new method for feature selection and classification merging of multi-class NSL-KDD Cup99 dataset using Support Vector Machine (SVM) and discussed the classification accuracy of classifiers under different dimension features [12].

Torres et al. [16] first converted network traffic characteristics into a series of characters and then used Recurrent Neural Network (RNN) to learn their temporal characteristics, which were further used to detect malicious network traffic.

Shiraz studied some new technologies to improve CANN intrusion detection methods' classification performance and evaluated their performance on the NSL-KDD Cup99 dataset [13]. He used the K Farthest Neighbor (KFN) and the K Nearest Neighbor (KNN) to classify the data

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

and used the Second Nearest Neighbor (SNN) of the data when the nearest and farthest neighbors have the same class label.

Wang et al. [17] proposed a malicious software traffic classification algorithm based on Convolutional Neural Network(CNN). By mapping the traffic characteristics to pixels, the network traffic image is generated, and the image is used as the input of the CNN to realize traffic classification.

Staudemeyer and Shamsinejad [13] proposed an intrusion detection algorithm based on Long Short-Term Memory (LSTM), which detects DoS attacks and probe attacks with unique time series in the KDD Cup99 dataset.

III. SYSTEM ANALYSIS

i) Existing System

In existing methods analysis of the viability of Decision tree, SVM were used to detect the behavior of network traffic by modeling it as a sequence of states that change over time. In existing methods verify the performance of methods network in classifying intrusion traffics. Experimental results show that accuracy is 73 percent.

Deep Learning as an essential subfield of machine learning, deep learning has shown excellent performance in Computer Vision (CV) [7], Natural Language Processing (NLP) [8]. Intrusion detection technology based on deep learning has been widely studied in academia and industry. The method of deep learning is to mine the potential features of high-dimensional data through training models and convert network traffic anomaly detection problems into classification problems [9]. By training a large number of data samples, adaptive learning of the difference between normal behavior and abnormal behavior effectively enhances the real-time performance of intrusion processing.

ii) Proposed System

We propose an end-to-end deep learning model cyber-attack detection that is composed of CNN, MLP, Decision tree, Random Forest methods. Linear Regression can well solve the problem of cyber-attack detection and provide a new research method for attack detection. We compare the performance of existing methods with traditional deep learning methods, the cyber-attack model can extract information from each packet. By making full use of the structure information of network traffic, this model can capture features more comprehensively. We evaluate our proposed network with a real NSL-KDD dataset. We use the classic NSL-KDD and the up-to-date CSECIC-IDS2018 as benchmark datasets and conduct detailed analysis and data cleaning. (2) This work proposes a machine learning algorithm, reducing the majority samples and augmenting the minority samples in the difficult set, tackling the class imbalance problem in intrusion detection so that the classifier learns the differences better in training.

IV. SYSTEM DESIGN

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

i) Algorithms

a) Random Forest

The Random Forest classification model is made up of several decision trees. In simple terms, it combines the results from numerous decision trees to reach a single result. The main difference between decision trees and random forests is that decision trees consider all the possible feature splits, however, random forests will only select a subset of those features.

Random Forest algorithm was developed by Breiman, L. [60]. This is an ensemble learning algorithm made up of several DT classifiers, and the output category is determined collectively by these individual trees. When the number of trees in the forest increases, the fallacy in generalization error for forests converges. There are also important benefits of the RF. For example, it can manage high-dimensional data without choosing a feature; trees are independent of each other during the training process, and implementation is fairly simple; however, the training speed is generally fast and, at the same time, the generalization functionality is good enough [4]. Random forest algorithm for machine learning has tree predictions, and based on tree predictions, the RF provides random forest predictions [61]. The Random Forest algorithm model is visualized in Figure.

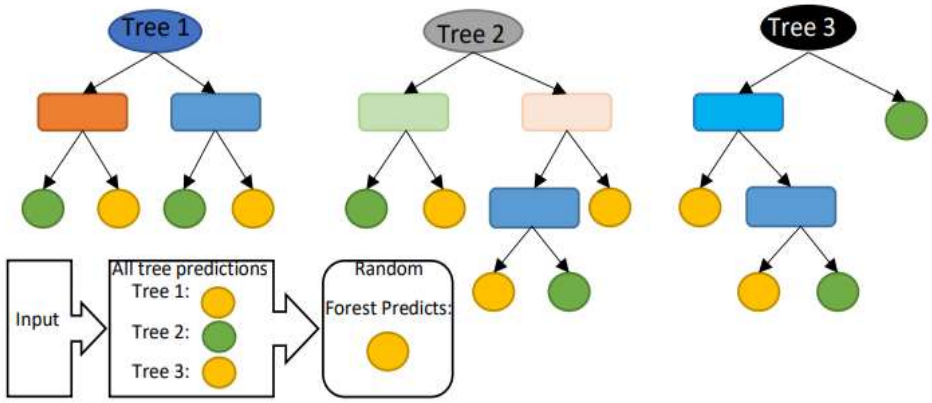

Figure: Random Forest algorithm

b) Logistic Regression (LR)

The adjusted r square by each key contributor was established using hypothesis testing. The factors that were found to be statistically significant after cross tabulations were used to train the multiplex logit model, which was then used to create the equation. In their study, they utilized unstructured historical machine data to train the ML classification algorithms including RF, XGBoost, and LR in predicting the machine failures. Various metrics were analyzed to determine the goodness of fit of the models. These metrics include empirical cross-entropy, area under the receiver operating characteristic curve (AUC), receiver operating characteristic curve itself (ROC), precision-recall

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

curve (PRC), number of false positives (FP), true positives (TP), false negatives (FN), and true negatives (TN) at various decision thresholds, and calibration curves of the estimated probabilities. Based on the results obtained, in terms of ROC, all the algorithms performed significantly better and almost similar. But in terms of decision thresholds, RF and XGBoost perform better than LR. Using a given set of independent variables, linear regression is used to estimate the continuous dependent variations. However, using a given set of independent variables, logistic regression is used to estimate the categorical contingent variations [68]. Graph of the linear regression model and logistics regression model are shown in Figure.

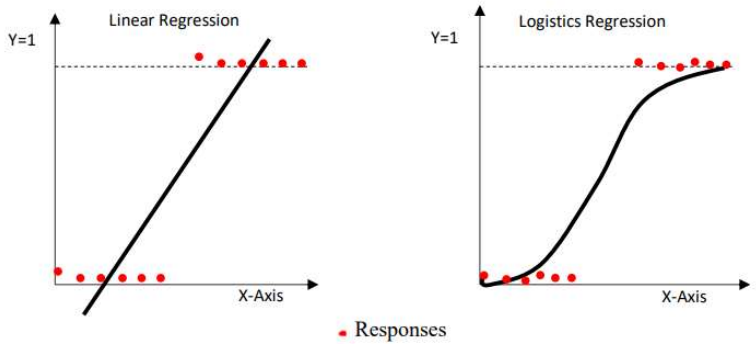

Figure: Logistic Regression

c) Decision Tree (DT)

Decision Tree is a network system composed primarily of nodes and branches, and nodes comprising root nodes and intermediate nodes. The intermediate nodes are used to represent a feature, and the leaf nodes are used to represent a class label [52]. DT can be used for feature selection [57]. DT algorithm is presented in Figure. DT classifiers have gained considerable popularity in a number of areas, such as character identification, medical diagnosis, and voice recognition. More notably, the DT model has the potential to decompose a complicated decision-making mechanism into a series of simplified decisions by recursively splitting covariate space into subspaces, thereby offering a solution that is sensitive to interpretation.

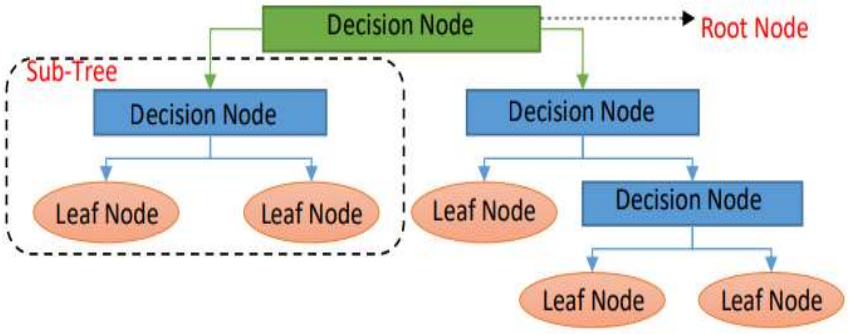

Figure: Decision Tree algorithm

ii) Modules

a) Data Collection

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

There are three symbolic data types in NSL-KDD data features: protocol type, flag and service. We use one-hot encoder mapping these features into binary vectors. One-Hot Processing: NSL-KDD dataset is processed by one-hot method to transform symbolic features into numerical features. For example, the second feature of the NSL-KDD data sample is protocol type. The protocol type has three values: tcp, udp, and icmp. One-hot method is processed into a binary code that can be recognized by a computer, where tcp is [1, 0, 0], udp is [0, 1, 0], and icmp is [0, 0, 1].

b) Pre-Processing

When the dataset is extracted, part of the data contains some noisy data, duplicate values, missing values, infinity values, etc. due to extraction errors or input errors. Therefore, we first perform data preprocessing. The main work is as follows.

(1) Duplicate values: delete the sample's duplicate value, only keep one valid data.

(2) Outliers: in the sample data, the sample size of missing values(Not a Number, NaN) and Infinite values(Inf) is small, so we delete this.

(3) Features delete and transform: In CSE-CIC-IDS2018, we delete features such as ''Timestamp'', ''Destination Address'', ''Source Address'', ''Source Port'', etc. If features ''Init Bwd Win Byts'' and features ''Init Fwd. Win Byts'' have a value of −1, we add two check dimensions. The mark of −1 is 1. Otherwise, it is 0. In NSL-KDD, we use the One Hot encoder to complete this conversion. For example, ''TCP'', ''UDP'' and ''ICMP'' are functions of three protocol types. After One Hot encoding, they become binary vectors (1, 0, 0), (0, 1, 0), (0, 0, 1). The protocol type function can be divided into three categories, including 11 categories for flag function and 70 categories for service function. Therefore, the 41 dimensions initial feature vector becomes 122 dimensions.

(4) Numerical standardization: In order to eliminate the dimensional influence between indicators and accelerate the gradient descent and model convergence, the data is standardized, that is, the method of obtaining Z-Score, so that the average value of each feature becomes 0 and the standard deviation becomes 1, converted to a standard normal distribution, which is related to the overall sample distribution, and each sample point can have an impact on standardization. The standardization formula is as follows, u is the mean of each feature, s is the standard deviation of each feature, and x 0 i is the element corresponding to each column's features.

c) Train-Test Split and Model Fitting

Now divide our dataset into training and testing data. Our objective for doing this split is to assess the performance of our model on unseen data and to determine how well our model has generalized on training data. This is followed by a model fitting which is an essential step in the model building process.

V. TESTING

The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time. Stating formally, we can say, testing is a process of executing a program with intent of finding an error.

1. A successful test is one that uncovers an as yet undiscovered error.

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

2. A good test case is one that has probability of finding an error, if it exists.
3. The test is inadequate to detect possibly present errors.
4. The software more or less confirms to the quality and reliable standards.

i) Code testing

This examines the logic of the program. For example, the logic for updating various sample data and with the sample files and directories were tested and verified.

ii) Specification Testing

Executing this specification starting what the program should do and how it should perform under various conditions. Test cases for various situation and combination of conditions in all the modules are tested.

iii) Unit testing

In the unit testing we test each module individually and integrate with the overall system. Unit testing focuses verification efforts on the smallest unit of software design in the module. This is also known as module testing. The module of the system is tested separately. This testing is carried out during programming stage itself. In the testing step each module is found to work satisfactorily as regard to expected output from the module. There are some validation checks for fields also. For example, the validation check is done for varying the user input given by the user which validity of the data entered. It is very easy to find error debut the system.

VI. RESULTS

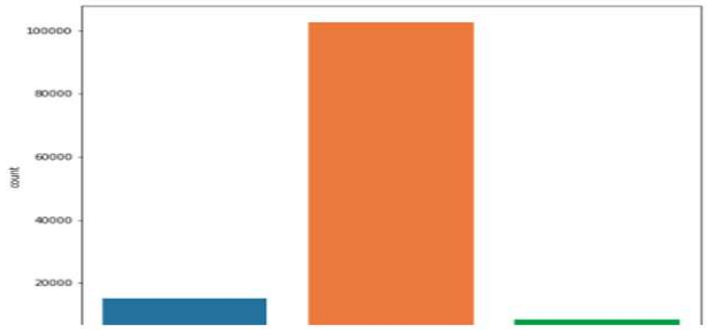

Figure: Protocol type distribution

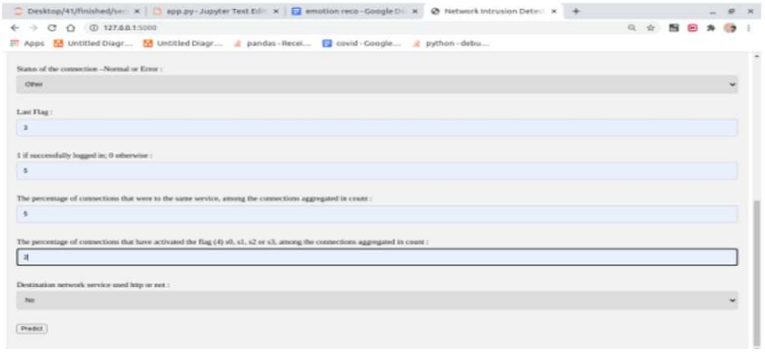

Figure: Data Collection for Analysis

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

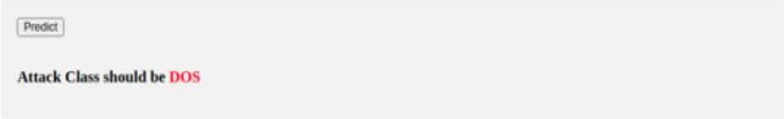

Figure: Predicating The Type of Attack

VII. CONCLUSION

As network intrusion continues to evolve, the pressure on network intrusion detection is also increasing. In particular, the problems caused by imbalanced network traffic make it difficult for intrusion detection systems to predict the distribution of malicious attacks, making cyberspace security face a considerable threat. This paper proposed a novel Difficult Set Sampling Technique, which enables the classification model to strengthen imbalanced network data learning. A targeted increase in the number of minority samples that need to be learned can reduce the imbalance of network traffic and strengthen the minority's learning under challenging samples to improve the classification accuracy. We used six classical classification methods in machine learning and deep learning and combined them with other sampling techniques. Experiments show that our method can accurately determine the samples that need to be expanded in the imbalanced network traffic and improve the attack recognition more effectively. In the experiment, we found that deep learning performance is better than machine learning after sampling the imbalanced training set samples through the MLP algorithm. Although the neural networks strengthen data expression, the current public datasets have already extracted the data features in advance, which is more limited for deep learning to learn the preprocessed features and cannot take advantage of its automatic feature extraction. Therefore, in the next step, we plan to directly use the deep learning model for feature extraction and model training on the original network traffic data, performance the advantages of deep learning in feature extraction, reduce the impact of imbalanced data and achieve more accurate classification.

REFERENCES

[1] D. E. Denning, ‘‘An intrusion-detection model,’’ IEEE Trans. Softw. Eng., vol. SE-13, no. 2, pp. 222–232, Feb. 1987.

[2] N. B. Amor, S. Benferhat, and Z. Elouedi, ‘‘Naive Bayes vs decision trees in intrusion detection systems,’’ in Proc. ACM Symp. Appl. Comput. (SAC), 2004, pp. 420–424.

[3] M. Panda and M. R. Patra, ‘‘Network intrusion detection using Naive Bayes,’’ Int. J. Comput. Sci. Netw. Secur., vol. 7, no. 12, pp. 258–263, 2007.

[4] M. A. M. Hasan, M. Nasser, B. Pal, and S. Ahmad, ‘‘Support vector machine and random forest modeling for intrusion detection system (IDS),’’ J. Intell. Learn. Syst. Appl., vol. 6, no. 1, pp. 45–52, 2014.

[5] N. Japkowicz, ‘‘The class imbalance problem: Significance and strategies,’’ in Proc. Int. Conf. Artif. Intell., vol. 56, 2000, pp. 111–117.

[6] Y. LeCun, Y. Bengio, and G. Hinton, ‘‘Deep learning,’’ Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[7] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, ‘‘Deep learning for visual understanding: A review,’’ Neurocomputing, vol. 187, pp. 27–48, Apr. 2016.
[8] T. Young, D. Hazarika, S. Poria, and E. Cambria, ‘‘Recent trends in deep learning based natural language processing [review article],’’ IEEE Comput. Intell. Mag., vol. 13, no. 3, pp. 55–75, Aug. 2018.
[9] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, ‘‘A deep learning approach to network intrusion detection,’’ IEEE Trans. Emerg. Topics Comput. Intell., vol. 2, no. 1, pp. 41–50, Feb. 2018.
[10] D. A. Cieslak, N. V. Chawla, and A. Striegel, ‘‘Combating imbalance in network intrusion datasets,’’ in Proc. IEEE Int. Conf. Granular Comput., May 2006, pp. 732–737.
[11] B. B. Zarpelo, R. S Miani, C. T. Kawakani, and S. C. de Alvarenga, ‘‘A survey of intrusion detection in Internet of Things,’’ J. Netw. Comput. Appl., vol. 84, pp. 25–37, Apr. 2017.
[12] B. Mukherjee, L. T. Heberlein, and K. N. Levitt, ‘‘Network intrusion detection,’’ IEEE Netw., vol. 8, no. 3, pp. 26–41, May 1994.
[13] S. Kishorwagh, V. K. Pachghare, and S. R. Kolhe, ‘‘Survey on intrusion detection system using machine learning techniques,’’ Int. J. Control Automat., vol. 78, no. 16, pp. 30–37, Sep. 2013.
[14] N. Sultana, N. Chilamkurti, W. Peng, and R. Alhadad, ‘‘Survey on SDN based network intrusion detection system using machine learning approaches,’’ Peer-to-Peer Netw. Appl., vol. 12, no. 2, pp. 493–501, Mar. 2019.
[15] M. Panda, A. Abraham, S. Das, and M. R. Patra, ‘‘Network intrusion detection system: A machine learning approach,’’ Intell. Decis. Technol., vol. 5, no. 4, pp. 347–356, 2011.
[16] W. Li, P. Yi, Y. Wu, L. Pan, and J. Li, ‘‘A new intrusion detection system based on KNN classification algorithm in wireless sensor network,’’ J. Electr. Comput. Eng., vol. 2014, pp. 1–8, Jun. 2014.

Vol. 21, No. 1, (2024)
ISSN: 1005-0930