Denial of Service (DoS) Attack Detection: Performance Comparison of Supervised Machine Learning Algorithms

Zhuolin Li[1], Hao Zhang[1], Hossain Shahriar[2,3], Michael Whitman[3], Dan Lo[1], Kai Qian[1]
[1]*Department of Computer Science,* [2]*Department of Information Technology,* [3]*Institute for Cybersecurity Workforce Development*
Kennesaw State University, USA
{zli29, hzhang13}@students.kennesaw.edu
{hshahria, mwhitman, dlo2, kqian}@kennesaw.edu

Fan Wu
Department of Computer Science
Tuskegee University, USA
fwu@tuskegee.edu

***Abstract* - Denial of Service (DoS) is one of the common attempts in security hacking for making computation resources unavailable or to impair geographical networks. In this paper, we detect Denial of Service (DoS) attack from publicly available datasets using Logistic regression, Naive Bayes algorithm and artificial neural networks. The results from our experiments indicate that the accuracy, ROC curve and balanced accuracy of artificial neural network were higher than Naive Bayes algorithm and logistic regression for slightly imbalanced distribution dataset.**

Keywords: Denial of Service, Cybersecurity, Naive Bayes, Artificial Neural Network, Logistic Regression.

I. INTRODUCTION

Denial of Service (DoS) attacks issue large number of requests directed to victim servers, which seem normal traffic but leading to the denial of access to services and resources for legitimate users. Existing network layer defense approaches such as firewall and Intrusion Detection System (IDS) are not applicable for detecting DoS attacks. A number of bots are available in the market that can automate DDoS attacks such as Dirtjumper [17]. Worst, DoS as a service is now currently available to mount attacks on legitimate entities [31]. Unfortunately, real-world DoS attacks are much more complex and almost always distributed in nature [24]. The early DOS attacks targeted on Yahoo!, eBay, and CNN in 2000. DoS attacks have been mounted against various websites such as Sony Play Station [12] and bitcoin [11].

The strategy of DOS attacks may be volumetric attacks which catch the bandwidth of the target server flooded with very high bits per second; Protocol-based attacks which capture the resources of the target server flooded with very high packages per second [23]. In recent years, with the rapid development of information technology and the progress of machine learning methods, researchers are committed to applying some machine learning methods to address cybersecurity challenges. Machine learning helps to find the normal traffic pattern and analyzes such attack patterns in network usage. In this paper, we apply three machine learning algorithms namely Naïve Bayes, Regression Analysis and Aritificial Neural Network (ANN) to differentiate normal network traffic from malicious DoS traffic.

Naive Bayes Theorem is used in Naive Bayes classifier. The probabilities for each class in the given dataset is predicted and the highest probability is the prediction. This process is called Maximum A Posteriori (MAP). We assume everything is independent given the class label. Logistic regression is a classical classifier of supervised learning, which is widely used in data mining, diseases diagnosis and economic prediction. The output of logistic regression can predict the probability of a class. The advantage of logistic regression is that it has low time complexity and high interpretability. Regression analysis is a strong statistical method designed to explain the relationship between one or more variables of which is the dependent variable and another one is independent variable by using a mathematical formula [1]. Artificial Neural Networks (ANN) are the abstraction, simplification, and simulation of the human brain [2]. ANNs interpret sensory data through a machine that perceives, marks or clusters raw input. The patterns they recognize are digital, contained in vectors, and all real-world data, whether images, sounds, text or time series, must be converted into vectors [7].

The classification algorithm can be train from the training set and build into a model. Then the model is used to identify new samples. In the field of machine learning, No Free Lunch (NFL) theory [9] proves that all learning algorithms perform equally well on average across all possible data sets. The experimental results in our experiment show that there is no one algorithm can retain the greatest performance in all datasets. Therefore, it is an appropriate choice to select different algorithms according to different datasets.

The challenge of applying ML algorithms is the Imbalanced datasets. An imbalanced dataset refers to the problem with classification problems where the distributions of classes are not equal. Imbalanced datasets are common in our daily life, such as DoS detection. Majority class refers to the data that has a large proportion in the examples. Usually the goal for imbalanced dataset is to detect the rare but important case [6]. We apply Naive Bayes, Logistic Regression and Neural Network classifier to denial of service attack with imbalanced dataset. Balanced accuracy is calculated as the average of the proportion corrects of each class individually. In this paper, we use the balance accuracy to analyze the performance of the three supervised algorithms on an imbalanced dataset and present some initial results on a publicly available dataset.

The rest of this article is organized as follows. Section II briefly introduces the related literature work. Section III introduces the DoS dataset we used in our experiment. Section IV discusses the classifier techniques. Section V provides the research results. Finally, Section VI concludes the paper.

II. RELATED WORKS

Yu et al. [25] suggest mitigating DoS attacks, specifically session flooding, by using trust management. The authors measured four specific parameters of trust for each user after every established connection. Measurements included short-term trust, long-term trust, negative trust and misuse trust. All measures were combined to generate an overarching trust value that is used to determine if the user's next request should be accepted or not. After evaluation, they concluded that their lightweight mechanism produced a negligible amount of computational cost and an acceptable overhead bandwidth based on the typical number of user sessions [25].

Tempesta is a framework developed by Krizhanovsky [26]. This framework is made up of a combined caching HTTP server and firewall. Among the objectives in this study, the author specifically outlines five goals. The framework should provide full access to all layers of the OSI model to allow for traffic classification, modification and systems for filtering. Another goal is to integrate the framework as a component of a Linux TCP/IP stack in hopes of handling short-term connections often used in Distributed DoS (DDoS) attacks. The framework should be closely intertwined with Linux security and netfilter subsystems. This is for classifying and blocking botnets and managing dynamic rules. Tempesta also mitigates web service overloads (common DDoS attacks) by using a reverse-proxy functionality. Lastly the author emphasizes the need to train classification algorithms and back-end servers on what normal HTTP messages or requests must contain so they are interpreted correctly by devices.

In another study, Zolotukhin et al. present a method to detect several types of DoS attacks in a timely fashion [27]. In general, the focus is aimed at detecting application layer DoS attacks that use encryption protocols. The researchers utilize statistics to apply an anomaly-based detection approach to extracted network packets. None of the packets were decrypted, which is the main point. The authors want to detect and analyze attacks without decrypting the network-level traffic specifically between a client and a web server. Conversations between clients and web servers are divided into clusters to create a model of a normal user's behavior. Each conversation is characterized by four main parameters including the source IP address, the source port, the destination IP address and the destination port. The distribution of the conversations into clusters is examined using the stacked auto-encoder (one of many deep learning algorithms) and deviations are marked as anomalies. Their method was able to lower false positive rates to less than two percent [27].

Based on previous research the authors developed, a lightweight DDoS detection mechanism for web servers [28]. They used Transductive Confidence Machines for K-Nearest Neighbors (TCM-KNN) and selection methods based on genetic algorithm. The goal of the present research is to propose a more efficient instance selection method that works better for a real network situation and they call the Extend Fuzzy C-Means (E-FCM) algorithm. By utilizing the algorithm, the researchers reduced the time their mechanism takes by a factor of 4.2. However, the reduction in time comes at the cost of increasing the false positive rate.

In order to monitor application layer DoS attacks against well-known websites, the authors of one study introduced a scheme based on document popularity. The authors suggest using an access matrix to discover any existing spatial-temporal pattern of a typical surge in the number of webpage visitors. To abstract the matrix the researchers analyzed both principal and independent components. A model is developed to detect DDoS attacks based on the entropy of the document's popularity. Their approach includes multidimensional data processing, the advantages of the Hidden Semi-Markov Model, computational complexity and the self-adapting scheme, among several parameters [29].

Flash events and denial-of-service (DoS) attacks both result in degraded web services. A flash event is a sudden peak in the number of visitors to a certain webpage and is considered to be normal, but flash events can keep webpages from functioning completely. The authors of the study suggest the use of enhanced content distribution networks (CDNs) to protect web sites. Through the use of traffic patterns, file reference characteristics and client characteristics, flash events are separated from DoS attacks. Authors of the study state that only about 80 percent of webpages are accessed during any one flash event and the enhanced CDN can help alleviate that percentage. Implementing an enhanced CDN distinguishes flash events from DoS attacks on a server [30].

Other researchers have made some comparisons between different algorithms for classification problems [13-16]. Wu Xingdong et al. 2007 [10], C.45, K-Means, compare the top 10 data mining algorithms identified by the IEEE: SVM, Apriori, EM, Pagerank, AdaBoost, kNN, Naive Bayes, and CART. They provide descriptions of these algorithms and discuss their impact. Susi Marianingsih and Fitri Utaminingrum compared two algorithms (Support Vector Machine and Naive Bayes) in classification problem by evaluating their performance using precision, recall, f-measure and accuracy. The result showed that Support Vector Machine classifier accuracy is better than Naive Bayes classifier.

The Naive Bayes classifier is one of the oldest forms and methods used for the classification of binary indexes. The Naive Bayes classifier usually tends to be simple in the way it is being structured and its assumptions are also based on an unrealistic approach, the concept usually outweighs other techniques in terms of its proficiency [18]. Chan et al. [19] proposed revised Naive Bayes classifier to combat spam email attack, where each feature in the Naive Bayes classifier, additional weight based on the number of ham and spam containing the feature is added. The accuracy of the attacked samples of the proposed method is higher than the standard Naive Bayes classifier, especially when the degree of attack is quite large. The work did not focus on applying it for DoS attack detection and comparison of performance among classifiers.

Similarly, Support vector machines [20], content-based filtering employing Naive Bayesian classification, Support Vector Machine, K Nearest Neighbor, Neural Networks [21, 22] have been explored for cybersecurity problems, except denial of service detection.

III. DATASETS

Our denial of service attack dataset is from Datahub. This dataset is a 10% stratified subsample of the data from the 1999 ACM KDD Cup. This dataset used for The Third International Knowledge Discovery and Data Mining Tools Competition, which was held in conjunction with KDD-99 The Fifth International Conference on Knowledge Discovery and Data Mining [5]. The aim is to build a network intrusion detector, a predictive model that can distinguish between "bad" connections (intrusions or attacks) and "good" normal connections.

duration	protocol_ty	service	flag	src_bytes	dst_bytes	land	wrong_frag
0	tcp	http	SF	181	5450	0	0
0	tcp	http	SF	239	486	0	0
0	tcp	http	SF	235	1337	0	0
0	tcp	http	SF	219	1337	0	0
0	tcp	http	SF	217	2032	0	0
0	tcp	http	SF	217	2032	0	0
0	tcp	http	SF	212	1940	0	0
0	tcp	http	SF	159	4087	0	0
0	tcp	http	SF	210	151	0	0
0	tcp	http	SF	212	786	0	0
0	tcp	http	SF	210	624	0	0
0	tcp	http	SF	177	1985	0	0
0	tcp	http	SF	222	773	0	0
0	tcp	http	SF	256	1169	0	0
0	tcp	http	SF	241	259	0	0
0	tcp	http	SF	260	1837	0	0
0	tcp	http	SF	241	261	0	0
0	tcp	http	SF	257	818	0	0
0	tcp	http	SF	233	255	0	0

Figure 1. Denial of Service dataset

Figure 1 shows a snapshot of the dataset used in our work. This dataset includes numerical features and nominal features. 41 features are used to describe the dataset such as duration length (number of seconds) of the connection [5]. We convert the categorical Variable into dummy variables due to scikit-learn does not accept non-numerical features.

And we divide the dataset into training set and test set respectively. The training set of the dataset accounts for 75% of each total sample, and the test set accounts for 25% of each total sample. And then we also test 20% (Testing) / 80% (Training) and 30% (Testing) / 70% (Training).

IV. METHODOLOGY

A. Multinomial Naive Bayes

The multinomial Naive Bayes classifier works well for word counts for text classification due to its discrete features. Multinomial Naive Bayes classifier assumes that a corpus of documents is generated by selecting a class [4]. Multinomial Naive Bayes assumes prior probability for polynomial distribution characteristics, namely the following type:

$$P(X_j = x_{jl} \mid Y = C_k) = \frac{x_{jl} + \lambda}{m_k + n\lambda} \quad (1)$$

where $P(X_j = x_{jl} \mid Y = C_k)$ is conditional probability of each value of the j-dimensional feature of the k^{th} category, m_k is the number of samples in the training set with the output of class k. λ is constant value, we use (the default value) 1. Multinomial Naive Bayes' parameters is more than Gaussian Naive Bayes, but also only just three altogether.

B. Logistic Regression

Logistic regression is a linear model for binary classification problems. A linear combination of the product of the independent variable ($x_1, x_2, x_3, \ldots x_n$) and its corresponding weight ($w_1, w_2, w_3, \ldots w_n$) and put these into the sigmoid equation which is used to restrict the output to an interval between 0 and 1. The output is expressed as the probability of the event happen. The sigmoid function is defined in Eq. 2.

$$S(h) = \frac{1}{1 + e^{-h}} \quad (2)$$

Where, h is the linear combination of the product of the independent variable ($x_1, x_2, x_3, \ldots x_n$) and its corresponding weight ($w_1, w_2, w_3, \ldots w_n$), defined in Eq. 3.

$$h(x_i) = \sum_{j=0}^{n} w_j x_{ij} = w_0 x_{i0} + w_1 x_{i1} + w_2 x_{i2} + \ldots + w_n x_{in} \quad (3)$$

This probability can be interpreted as: given these independent features ($x_1, x_2, x_3, \ldots x_n$) multiplied by the weight w, the probability that the sample belongs to category 1 $P(y = 1 \mid \sum_n^{i=0} x_i w_i)$. Through step function, we can obtain the following formula:

f(n)= {1 if S(h) > threshold (4)

0 otherwise

}

where, threshold we usually set it to 0.5.

Many applications do not just want a class label, they want to figure out the probability of belonging to a category. Logistic regression models do a good job for that purpose. The purpose of our logistic regression model is to find the best values of these weights to maximize the probability of our sample data.

Suppose we are given the sample features (x1, x2, x3, … xn), the likelihood function is defined in Eq. 5.

$$l(w) = f(x_1, x_2, x_3, \ldots, x_n \mid w) \quad (5)$$

If these features ($x_1, x_2, x_3, \ldots x_n$) are independent of each other. The likelihood function can be simplified as in Eq. 6.

$$l(w) = \prod_{i=0}^{n} f(x_i \mid w) \quad (6)$$

However, if we have a lot of features, in this case, by multiplying a lot of terms, which are usually very small, the probability function becomes very small. As a result, we should use the log probability function. First, if the probability is very small, it can prevent potential numerical underflow. Second, we convert the product to the sum, which

makes it easier to find the derivative of the function. Third, the log function is monotone, maximizing the value of the probability function is same as maximizing the value of the log probability function. The formula of the log probability function is as follows:

$$l(w) = \log(l(w)) = \sum_{i}^{n} \log(f(x_i \mid w)) \tag{7}$$

So, we can use the probability function to define the weight w as follows:

$$\begin{aligned} L(w) &= \prod_{i=0}^{n} P(y^{(i)} \mid x^{(i)}; w) \\ &= \prod_{i=0}^{n} S(h^{(i)})^{y^{(i)}} (1 - S(h^{(i)}))^{1-y^{(i)}} \end{aligned} \tag{8}$$

And we apply this to log probability function, the formula is defined in Eq. 9.

$$\begin{aligned} l(w) &= \log(L(w)) \\ &= \sum_{i=0}^{n} y^{(i)} \log(S(h^{(i)})) + (1 - y^{(i)}) \log(1 - S(h^{(i)})) \end{aligned} \tag{9}$$

Our goal is to maximize the log probability function and find an optimal weight *w*. We can use the gradient descent algorithm to minimize this function by putting a minus sign in front of the log probability function. The cost function of logistic regression is:

$$J(w) = -\sum_{i=0}^{n} y^{(i)} \log(S(h^{(i)})) + (1 - y^{(i)}) \log(1 - S(h^{(i)})) \tag{10}$$

Where S is sigmoid function (Eq. 2), *n* is the size of samples, *h* is hypothesis function.

C. Artificial Neural Networks (ANN)

ANN (Figure 2) includes layers and each layer is made up of nodes. A node is a place for calculation, loosely modeled on a neuron in the human brain that is activated when given enough stimulation. The node combines the input of the data with a set of coefficients or weights that can amplify or weaken the input, thereby assigning importance to the input of the task to be learned by the algorithm [7] (*e.g.*, most useful input to classify data without errors). These inputs weighted products are summed, and then through a node's so-called activation function, to determine whether and to what extent the signal should further influence the final result, such as classification behavior, through the network. If the signal passes, the neuron is activated. The nodal layer is a row of neuron-like switches that turn on and off as input passes through the network. The output of each layer is also the input of the subsequent layers, starting from the initial input layer that receives the data.

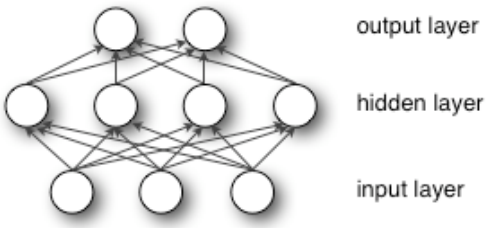

Figure 2. Artificial Neural Network

For our experiment, we have 3 hidden layers with 20 nodes respectively (Figure 2). And the output layer has 2 outputs because our classification problem is binary.

Activation Functions: The activation function determines the output, based on its input. We usually use Relu function (Eq. 11) in our Hidden layer, and the softmax function (Eq. 12) to our output layer.

$$f(x) = \max(0, x) \tag{11}$$

$$f(x) = \frac{e^{x_i}}{\sum_k e^{x_k}} \tag{12}$$

V. EVALUATION

We use the Google open source CoLab collaborative learning platform [8] for analyzing three algorithms on denial of service dataset. We evaluate the dataset with the 3 algorithms. For the denial of service dataset, true positive means that positive examples are correctly assigned to the positive class. In this dataset, it means this website is normal. True negative refers to the negative examples correctly predicted to the negative class. It means this website is attacked by hacker.

False positive means that the algorithm is incorrectly considered negative examples as positive examples. In other words, when a sample is normal, the algorithm mistakenly places the bad connection website in the normal website categories. False negative is defined as positive examples wrongly allocated to negative class. It means the sample website is normal; however, the algorithm misjudged this website is impaired.

The confusion matrix for the dataset is shown in Table 1, 2 and 3 for different split of training and testing datasets. We could judge the first model by its overall accuracy, which works well for most datasets. However, the accuracy might be insufficient to reflect the performance of a model in imbalanced dataset. So, we use balanced accuracy to determine whether an algorithm is a good algorithm.

$$\text{Balanced accuracy} = ((\text{TP}/(\text{TP} + \text{FP})) + (\text{TN}/(\text{TN} + \text{FN})))/2$$

In the above equation, TP is true positive, FP is false positive. TN is True Negative (TN), FN is False Negative (FN). The higher the balance accuracy is, the more the classification is put into the right place. The balanced accuracy analysis is shown in Table 4. Here, we find that Neural Network outperforms Naive Bayes and Logistic Regression.

Table 1: Confusion matrix of Denial of Service dataset (75%training, 25% testing)

Logistic Regression			Naive Bayes			Neural Network		
Predicted:No	Predicted:Yes	N=123505	Predicted: No	Predicted:Yes	N=123505	Predicted:No	Predicted:Yes	N=123505
24281	111	Actual:No	23722	670	Actual:No	24346	46	Actual:No
201	98912	Actual:Yes	1392	97721	Actual:Yes	57	99056	Actual:Yes

Table 2: Confusion matrix of Denial of Service dataset (70%training, 30% testing)

Logistic Regression			Naive Bayes			Neural Network		
Predicted: No	Predicted:Yes	N=148206	Predicted:No	Predicted:Yes	N=148206	Predicted:No	Predicted:Yes	N=148206
29218	137	Actual:No	28521	834	Actual:No	29287	68	Actual:No
240	118611	Actual:Yes	1645	117206	Actual:Yes	68	118783	Actual:Yes

Table 3: Confusion matrix of Denial of Service dataset (80%training, 20% testing)

Logistic Regression			Naive Bayes			Neural Network		
Predicted: No	Predicted:Yes	N=98804	Predicted:No	Predicted:Yes	N=98804	Predicted:No	Predicted:Yes	N=98804
19473	102	Actual:No	19252	323	Actual:No	19546	29	Actual:No
155	79074	Actual:Yes	1100	78129	Actual:Yes	63	79166	Actual:Yes

The Receiver Operating Characteristic (ROC) curves in Figures 3, 4, and 5 show the performance of denial of service detection for various training and testing dataset split. Neural networks algorithm performed better than logistic regression and Naive Bayes algorithm. Therefore, Neural Network algorithm can better classify this denial of service dataset.

Table 4: Balanced Accuracy and comparison with Naive Bayes, Logistic Regression, and Neural Network.

Dataset split	**Logistic Regression**	**Naive Bayes**	**Neural Network**
(75%/25%)	0.99671067	0.97924370	0.99876951
(70%/30%)	0.99665683	0.97887415	0.99855692
(80%/20%)	0.99664646	0.98480778	0.99876402

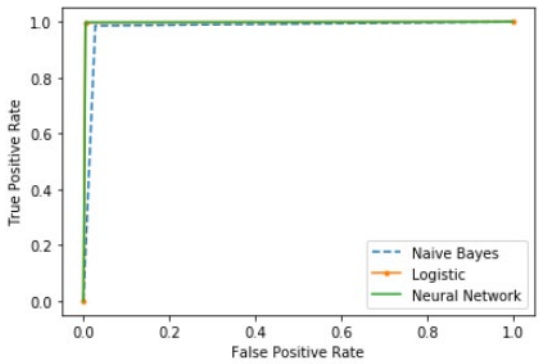

Figure 3: ROC curve for DoS detection (75%training, 25% testing)

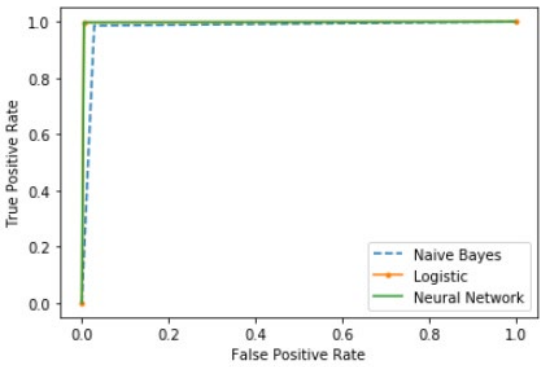

Figure 4: ROC curve for DoS detection (70%training, 30% testing)

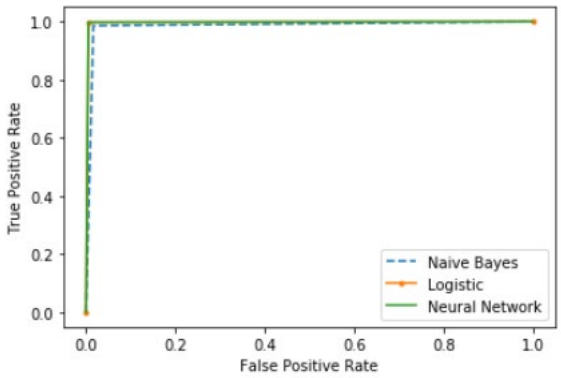

Figure 5: ROC curve for DoS detection (80%training, 20% testing)

Table 5: AUROC comparison with Naive Bayes, Logistic Regression, and Neural Network.

Dataset split	Logistic Regression	Naive Bayes	Neural Network
(75%/25%)	0.979	0.997	0.999
(70%/30%)	0.979	0.997	0.999
(80%/20%)	0.996	0.985	0.999

Table 6: Running time comparison with Naive Bayes, Logistic Regression, and Neural Network.

Dataset split	Logistic Regression	Naive Bayes	Neural Network
(75%/25%)	8160ms	551ms	41200ms
(70%/30%)	7310ms	567ms	39300ms
(80%/20%)	8220ms	500ms	44400ms

Table 5 compares the Area Under the ROC (AUROC). It is used for classification analysis to determine which models are used to best predict categories. Here, the true positive rate and false positive rate are plotted. The closer the AUC of the model is to 1, the better. Therefore, the model with higher AUC is better than the model with lower AUC. In our experiment, artificial neural network has the largest value, so it is the best model to detect this dataset.

We compare the runtime performance of the three models. As can be seen from Table 6, artificial neural networks need to spend more time on training, logistic regression is moderate, Naive Bayes spend the shortest time, but the accuracy is not as good as the other two machine learning algorithms.

VI. CONCLUSION

Security threats are evolving and getting more hidden and complicated. Detecting malicious security threats and attacks have become a huge burden to cyberspace. We should apply proactive prevention and early detections of security vulnerabilities and threats rather than patching security holes afterwards. To analyze large amount of data to find out suspicious behaviors, threat patterns, and vulnerabilities and to predict and prevent future cybersecurity threats are a challenge. Machine Learning (ML) is a powerful instrument to take up such challenge.

In this paper, we used dataset to classify denial of service attack by using Naive Bayes algorithm, artificial neural networks and logistic regression and compare their performance. The experimental results show that the neural network algorithm performed better than logistic regression and Naive Bayes algorithm in the dataset with slightly imbalanced distribution.

REFERENCE

[1] AKSU, G., & REYHANLIOGLU KECEOGLU, C. (2019). Comparison of Results Obtained from Logistic Regression, CHAID Analysis and Decision Tree Methods. *Eurasian Journal of Educational Research (EJER)*, (84), 115–134.

[2] Chunhui Bao, Yifei Pu, and Yi Zhang, "Fractional-Order Deep Backpropagation Neural Network," *Computational Intelligence and Neuroscience*, Vol. 2018, Article ID 7361628, 2018.

[3] Karthickveerakumar. (2017). Spam Filter: Identifying spam using, Kaggle.com

[4] Xu, Shuo & Li, Yan & Zheng, Wang. (2017). Bayesian Multinomial Naïve Bayes Classifier to Text Classification. 347-352. 10.1007/978-981-10-5041-1_57.

[5] TunedIT. 1999 ACM KDD Cup, SIGKDD.org

[6] Stiris Kotsiantis, Dimitris Kanellopoulos, Panayiotis Pintelas. "Handling imbalanced datasets: A review". *GESTS International Transactions on Computer Science and Engineering*, Vol.30, 2006

[7] Beginner's Guide to Neural Networks and Deep Learning, Pathmind, skymind.ai/wiki/neural-network.

[8] Google Colaboratory, Accessed from https://colab.research.google.com/notebooks/welcome.ipynb

[9] Wolpert DH. The lack of a priori distinctions between learning algorithms. *Neural Comput*. 1996;8(7):1341–90.

[10] Wu, Xindong, and Vipin Kumar. The Top Ten Algorithms in Data Mining. CRC Press, 2009.

[11] T. Bienkowski, Your Network or Your Bitcoins: Three Rules for Dealing with DDoS Extortion Threats, 2019, https://www.arbornetworks.com/blog/insight/your-network-or-your-bitcoins-three-rules-for-dealing-with-ddos-extortion-threats/

[12] Sony Playstation Hack, 2019, Accessed from http://www.scmagazine.com/sony-psn-downed-hacking-group-claims-ddos-attack/article/463065/

[13] Chen, Jian, et al. "Credit Card Fraud Detection Using Sparse Autoencoder and Generative Adversarial Network." *Proc. of 9th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, Nov. 2018, pp. 1054–1059.

[14] Nurul Fitriah Rusland, Norfaradilla Wahid, Shahreen Kasim,Hanayanti Hafit, "Analysis for Naive Bayes Algorithm in Email Spam Filter which is among Multi numbers of Datasets," IOP Conference Series: Materials Science and Engineering, 2017, 226, 012091. DOI: 10.1088/1757-899x/226/1/012091

[15] Shams, R & Mercer R., "Classifying spam emails using text and readability features," Proc. of 13th IEEE International Conference on Data Mining, Dallas, TX, 2013, pp. 657-666.

[16] S. R. Gomes et al., "A comparative approach to email classification using Naive Bayes classifier and hidden Markov model," *Proc. of 4th International Conference on Advances in Electrical Engineering (ICAEE)*, Dhaka, 2017, pp. 482-487

[17] C. Wueest, The continued rise of DDoS attacks, Symantec Technical Report, 2019, Accessed from https://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/the-continued-rise-of-ddos-attacks.pdf

[18] K. Larsen, Generalized Naive Bayes Classifiers, *ACM SIGKDD Explorations Newsletter*, Vol.7, No.1, pp. 76-81.

[19] Juayan, P. & Chan, P. (2019). Revised Naïve Bayes classification for bating the attack in filtering, *2013 International Conference on Machine Learning and Cybernetics*, Tianjin, 2013, pp. 610-614.

[20] Dada, E., et al., Machine Learning to email spam filtering where review, approaches and open research problems, 5(6), https://www.sciencedirect.com/science/article/pii/S2405844018353404

[21] V. Christina, S. Karpagavalli, G. Suganya, "Email spam filtering using supervised machine learning techniques," Int. J. Comput. Sci. Eng., 02 (09) (2010), pp. 3126-3129.

[22] J.R. Mendez, F. Díaz, E.L. Iglesias, J.M. Corchado, "A comparative performance study of feature selection methods for the anti-spam filtering domain, Advances in Data Mining," *Applications in Medicine, Web Mining, Marketing, Image and Signal Mining*, Springer Berlin Heidelberg, 2006, pp. 106-120.

[23] Ann Harrison, The Denial-of-Service Aftermath, CNN, Cable News Network, Feb. 2000.

[24] Paffenroth, R. C., and C. Zhou. "Modern Machine Learning for Cyber-Defense and Distributed Denial-of-Service Attacks," *IEEE Engineering Management Review,* Vol. 47, No. 4, Dec. 2019, pp. 80‑85.

[25] Yu, J. et al., "Mitigating application layer distributed denial of service attacks via effective trust management," *IET Communications*, 4(16), 2010. pp. 1952-1962. DOI: 10.1049/iet-com.2009.0809

[26] A. Krizhanovsky, "Tempesta: a framework for HTTP DDoS attacks mitigation," *Proceedings of 24th Annual International Conference on Computer Science and Software Engineering (CASCON)*. IBM Corp., 2014, pp. 148-162.

[27] Mikhail Zolotukhin, Timo Hamalainen, Tero Kokkonen, Jarmo Siltanen, "Increasing Web Service Availability by Detecting Application-Layer DDoS Attacks in Encrypted Traffic," *Proc. of 2016*

23rd International Conference on Telecommunications (ICT), Thessaloniki, Greece.

[28] Yang Li, Tian-Bo Lu, Li Guo, Zhi-Hong Tian, Qin-Wu Nie, "Towards Lightweight and Efficient DDoS Attacks Detection for Web Server," *Proc. of World Wide Web*, 2009, pp. 1139-1140.

[29] Yi Xie and Shun-Zheng Yu, "Monitoring the Application-Layer DDoS Attacks for Popular Websites," *IEEE/ACM Transactions on Networking*, 17(1), Feb 2009, pp. 15-25.

[30] Jaeyeon Jung, Balachander Krishnamurthy, Michael Rabinovich, "Flash Crowds and Denial of Service Attacks: Characterization and Implications for CDNs and Web Sites," *Proc. of World Wide Web*, May 2002, Honolulu, USA, pp. 293-304.

[31] J. Cheng, J. Yin, Y. Liu, Z. Cai, and C. Wu, "DDoS Attack Detection Using IP Address Feature Interaction," *Proc. of 2009 International Conference on Intelligent Networking and Collaborative Systems*, pp. 113-118.