Proceedings of the Faculty of Technical Sciences, Novi Sad

UDC: 007:004.056
DOI: https://doi.org/10.24867/21BE25Ogundare

ISPITIVANJE ALGORITAMA MAŠINSKOG UČENJA ZA PREDVIĐANJE NAPADA NA SOFTVERSKI DEFINISANE MREŽE

INVESTIGATING MACHINE LEARNING ALGORITHMS PERFORMANCE FOR PREDICTING ATTACKS ON SOFTWARE DEFINED NETWORKS

Adedotun Olasunkanmi Ogundare, *Faculty of Technical Sciences, Novi Sad*

Field – SOFTWARE ENGINEERING AND INFORMATION TECHNOLOGIES

Kratak sadržaj – *Sa razvojem tehnologije obrade velikih podataka se razvijaju sistemi detekcije upada i anomalija. Malo istraživanja je sprovedeno u oblasti mreža definisanih softverom. Ovaj rad ispituje upotrebu tehnologija obrade velikih podataka za pronalazak sajber napada u softverski-definisanim mrežama. U radu su iskorišćeni algoritmi mašinskog učenja – stabla odlučivanja, nasumične šume i Naivni Bajes – za detekciju upada nad javno dostupnim skupom podataka. Postignuta preciznost je 96.7%.*

Ključne reči: *Veliki podaci, mašinsko učenje, mreže definisane softverom, sajber napadi, detekcija upada.*

Abstract – *As technology continues to evolve, the application of Big Data Technology is widening for intrusion and anomaly detection. However, not much has been done within the domain of Software Defined Networks (SDN). This paper explores how Big Data Technology can be used to predict cyber-attacks on SDNs by using historical dataset to build Machine Learning models. Using three widely-known supervised machine learning algorithms-Decision Tree, Random Forest and Naïve Bayes - results show average prediction accuracy of 96.7%. several other techniques were employed to evaluate the performance of the models.*

Keywords: *Big Data, Machine Learning, Software Defined Networks, Cyber Attacks, Intrusion Detection.*

1. INTRODUCTION

Modern trends in application development, which require rapid service provisioning, have led to wide adoption of programmable networks like Software Defined Network (SDN) and Cloud Computing. These allow agile applications to be deployed on networks seamlessly without requiring infrastructure overhaul and the resulting capital expenditure.

However, these developments have also introduced new security concerns. Past works noted that despite the benefits brought by the modern computing developments, it has also expanded the frontiers of cyber-attacks [13]. Machine Learning can be used to uncover threats by using models trained of previous datasets to make assumptions, as demonstrated in this work.

REMARK:
This paper is result of the Master thesis supervised by dr. Nikola Luburić.

1.1. SDN Overview

Being the modern approach to networking, Software Defined Network (SDN) abstracts infrastructure from control layer to deliver a dynamic network architecture that supports quick deployment of agile applications. With a 19% CAGR, the global SDN market is growing as more enterprises transition from traditional network to improve productivity [10]. It has also led to an increase in attack footprint. The SDN controller is a potential attack surface that can give hacker a total control of network without the need to hack individual network devices.

1.2. Problem Statement

The daunting task of securing SDN controller lies in filtering malicious nodes from normal ones because they exhibit similar traffic patterns. The conventional approach for defending against attacks is to rely on data validation techniques and apply thresholds which is then used to define rules on SDN controllers [13]. However, this has limitations because it is often subjective and when a single deciding factor has been used, it is unable to detect more than one attack at a time.

1.3. Machine Learning Approach to Attack Detection

Machine Learning can be used to identify patterns in historical data in network traffic statistics. A large scale of network data can be collected in SDN such as security device logs, IP parsing packets, network traffic statistics, system protection logs, flow-tables in OpenFlow switch and so on. Many open-sourced network datasets have been established by global body of research. Using ML models trained from historical data on network traffic, predictions can be made on future traffic.

2. SELECTED RELATED WORKS

Recognizing that the rapid rise in cybercrime has placed software security issues on the spotlight, Rudolf, et al. [11] investigated the performance of machine learning techniques in predicting functions with possible security vulnerabilities in JavaScript programs. Using datasets from public databases, eight machine learning algorithms were used to build prediction models, and static source code metrics were used as predictors. The research found that the best performing algorithm was the KNN which created a model that predicted vulnerable functions with an F-measure of 0.76 (0.91 precision and 0.66 recall), thus providing a viable practical approach to applying machine learning in predicting web vulnerability [11].

Marwin, Florian, & Samuel [9] followed the same approach in building machine learning models for predicting Hard Disk drive Failure.

They contend in their work that Self-Monitoring Analysis and Reporting Technology (SMART) had been used hitherto to predict hard drive failures. The SMART approach can determine the current health state but it cannot predict failures. Their research presents four Machine Learning (ML) models for hard disk failure prediction and evaluates the performance of each machine learning model [9].

It must be noted that prior to the work of Marwin, Florian, & Samuel [9], there have been some proposed approaches for managing the shortcoming of SMART. Popular among these approaches is the work of Chaves [4] which attempt to directly apply ML algorithms on SMART to predict time-to-failure in the form of regression. Also, Botezatu, et al. [3] applied classification in determining whether HDDs will fail within a specific time window. While [3] and [4] provide some practical insights, they suffer the limitation of considering HDD failure prediction only as static task by building models on random subsets of available data [3][4][9].

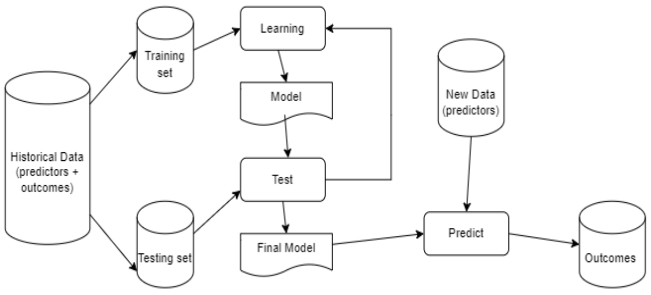
3. PREDICTION EXPERIMENT

A dataset representing historical data on SDN traffic was obtained and downloaded into a directory of the local machine used for this experiment.

Using Apache Spark technology implemented in JVM (Java Virtual Machine), the dataset was used to build classified ML models on Decision Tree, Random Forest and Naïve Bayes algorithm. A pipeline was developed to take the dataset through series of transformation based on Apache Spark Machine Learning workflow, as itemized below:

1. *Loading data from source into RDD (Resilient Distributed Dataset)* — For easy manipulation, dataset was intermittently transformed into data-frames using the Apache SQL library.
2. *Transforming RDD*—Data cleansing, float conversion, centering & scaling was performed to prepare the dataset for ML analysis.
3. *Creating ML Datatypes* —using the principles of dimensionality reduction, the Labeled Point was created as a function of the target variable and the dense vectors.
4. *Splitting training and testing data* –The split was varied at different ratios and performances metrics were recorded at each ratio.
5. *Creating a model*— ML models were built on the training data subset.
6. *Performing predictions* – Predictions were made on testing data subset.
7. *Testing accuracy* –Confusion matrix was generated to perform several accuracy and performance measures.

The workflow is repeated after scaling from a single machine and increasing partitions sequentially across two nodes, three nodes, and four nodes—running in parallel and measuring performance at each case.



3.1. Description of Dataset

The University of Nevada-Reno Intrusion Detection Dataset (UNR-IDD) has been chosen for the experiment. The dataset has 33 columns representing the feature variables (predictors) upon which the dense vectors are formed. It also has a binary label representing the target (outcome) variable which resolves to either of the two possible scenarios, 'normal' or 'attack.' The dataset has a total of 37,411 records out of which 33,638 portends an attack while 3,773 represents normal traffic [7].

In the description of the data collection technique, the developers of the UNR IDD dataset created custom application to collect and log port statistics captured at 5 seconds interval from OpenFlow (OF) switches [7]. The data features captured represents each column in the data set.

3.2. Dimensionality Reduction

In this experiment, three approaches of dimensionality reduction were used successively. At first, hypotheses were used in selecting predictors. Thereafter, correlation coefficient was used to reduce the number of predictors by discarding those that have low correlation with the target variable. Finally, Principal Component Analysis (PCA) was applied with varying constant K. Performance was measured at different values of constant K, and the best scenario was recorded. The same predictors were used to build models for the three selected supervised ML algorithms in order to provide a uniform benchmark for assessing and comparing their performance.

4. RESULTS AND ANALYSIS

The three ML models trained were used to predict attacks on the test data subset. After deriving the confusion matrix, performance is first measured by accuracy given by the equation 1 below:

EQ 1: Equation for Prediction Accuracy

$$= \frac{No. of\ Correctly\ predicted\ attacks\ (TP)}{Total\ Predictions\ (TP + TN + FP + FN)}\ X\ 100$$

Further to that, the *sensitivity* is measured using hit rate and recall with the goal of determining how good the models are in predicting actual 'attacks.'

EQ 2: Equation for Sensitivity

$$= \frac{TP}{TP + FN}\ X\ 100$$

Understanding that accuracy alone does not tell the full story when evaluating the results of prediction, the f-measure was employed.

A threshold level β (Beta) percentage is chosen as minimum probability for varying precision. This helps the author to examine its effect on the prediction accuracy. The training & testing dataset is also split in different proportions in order to examine their effect on prediction accuracy. The Equation 3 below is used to determine the f-measure.

EQ 3: F Beta Measure Equation:

$$= \frac{1}{\beta\, X \frac{1}{precision} + (1-\beta) X \frac{1}{recall}}$$

Table 1 shows the accuracy of the different Machine Learning (ML) algorithms used for this work. It also shows the f-measure with varying β threshold as well as varying training/testing split proportionality. The result reveals that Random Forest has the highest average prediction accuracy—98.404%.

Table 1. Accuracy of the different Machine Learning models on the attack prediction task

Decision Tree Algorithm				
Split	Acc	*B=.5*	*B=1*	*B=1.5*
70:30	96.70	99.72	99.99	100.25
60:40	96.19	99.91	99.95	99.99
50:50	96.33	99.85	99.85	99.86
40:60	96.27	99.80	99.98	100.16
30:70	95.98	99.84	99.88	99.91
AVG	96.29			
Random Forest Algorithm				
Split	Acc	*B=.5*	*B=1*	*B=1.5*
70:30	96.90	95.00	97.56	98.19
60:40	97.77	97.00	91.49	96.77
50:50	98.44	93.41	99.81	98.99
40:60	98.91	91.23	93.93	98.89
30:70	100.00	96.00	96.17	98.87
AVG	98.404			
Naive Bayes Algorithm				
Split	Acc	*B=.5*	*B=1*	*B=1.5*
70:30	91.00	87.20	94.24	94.57
60:40	94.20	94.13	94.96	94.78
50:50	97.30	93.87	93.01	94.99
40:60	97.19	91.16	93.19	95.01
30:70	97.56	91.19	93.19	95.01
AVG	95.45			

4.3. Findings

4.3.1 Effect of split ratio & probability threshold on Prediction Accuracy

The results reveal that in ML algorithm, the split ratio has no significant effect on prediction accuracy. This differs from the study of Rista et al. [12] who found tangible relationship between prediction accuracy and split ratio. The result however shows that threshold has notable effects on the prediction accuracy, thus agreeing with their findings**.**

4.3.2 Execution Performance of ML Algorithms

The performance of the various ML algorithm is further scrutinized by finding the average time it took each algorithm to perform prediction under different scale, as revealed in the Table 2 below.

Table 2 The execution performance of the tested Machine Learning algorithms

ML Algorithms	**Execution Time in Milliseconds (MS)**			
	Single JVM	Two Nodes	Three Nodes	Four Nodes
Decision Tree	*32616*	*32121*	*32121*	*34119*
Random Forest	*31864*	*31985*	*31942*	*33191*
Naïve Bayes	*13652*	*12191*	*12666*	*13011*
Clustering k = 2	27397	26291	26786	26789
Clustering K = 3	27397	23291	26645	26744
Clustering k = 3	27723	26264	26659	26759

The result shows Naïve Bayes has the best performance among the supervised ML algorithms. Measuring performance of ML at different scales extends on common approach of previous works, which mostly measured performance on a single scale—either as single machine or a specific number of clusters. For instance, Emre, et al. [8] measured performance of ML using two nodes which their work was premised upon. The result of performance evaluation done in this research compares results of different ML algorithm when run on single machine, two nodes, three nodes and four nodes.

4.3.3 Effects of Petitioning on Execution Performance

This research reveals that distribution of dataset across partitions and nodes can have effect on execution performance. More partitions (nodes) generally lead to better execution performance. However, this experiment reveals that if the dataset is not large enough to make sense of several partitions, then the performance goal of having multiple partitions may not be achieved due to the delay factors occasioned by cluster management. The principle of Lowest Usable Partitions (LUP), as coined by the author of this work, should be employed in view of the volume of data viz a viz the available computing resource of available nodes.

4.3.4 Effect of Dataset on Prediction Accuracy

Dataset is a factor that affects prediction accuracy. It was noted in [8] that higher variance in data will always increase the chances of false prediction. The intrusion detection dataset used for this work has relatively low variance, as reported by the developers of the dataset [7]. Hence, the predictions made from models built with the dataset shows reasonably high accuracy. An accurate prediction like that of this experiment can be leveraged by SDN controller to tighten the network security and block potential attacks.

5. CONCLUSION

This paper has leveraged the power of big data by building ML models to predict vulnerable hosts/traffic in SDN network and thereafter measuring accuracy and performance on different metrics. This has helped to identify factors that impact overall accuracy of prediction as well as execution performance. The models were built from historical dataset on network traffic. The research outcome can be used to define security rules for SDN controller to prevent malicious acts on a network. The result of the experiment has shown that ML algorithm can be used by SDN to define security rules based on the accurate predictions made.

LITERATURE

[1] Ashraf, J., & Latif, S. (2014). Handling Instrusion and DDoS attacks in Software Defined Networks using Machine Learning Techniques. *Software Engineering Conference*, (pp. 55-60).

[2] Ali, S., Sivaraman, V., Radford, A., & Jha, S. (2015). A survey of securing networks using software defined networking. *Reliability IEEE Transactions*, 1086-1097

[3] Botezatu, M. (2016). Predicting disk replacement towards reliable data centers. *22nd ACM SIGKDD International Conference on Knowledge Discipline and Data Mining*.

[4] Chaves, I. (2018). Hard disk drive failure prediction method based on bayesian network. *International Joint Conference on Neural Networks.* IEEE

[5] Hu, F., Hao, Q., & Bao, K. (2014). A survey on software deined network and Openflow: From concepts to Implementation. *IEEE communications survey tutorials*, 2181 - 2206.

[6] Kim, H., & Feamster, N. (2013). Improving network management with software defined networking. *IEEE Communications magazine*, 114-119

[7] Das, T., Abu, H., Shukla, R., Sengupta, S., & Arslan, E. (2022). *University of Nevada - Reno Intrusion Detection Dataset (UNR-IDD).* TechRxiv.

[8] Emre, U., Sonali, S.-B., & Rattikorn, H. (2018). Towards Prediction of Security Attacks on Software Defined Networks. *Interntional Conference on Big Data*

[9] Marwin, Z., Florian, E., & Samuel, K. (2021). Machine Learning Model Update Strategies for Hard Disk Drive Failure Prediction. *20th IEEE International Conference on Machine Learning and Applications.* IEEE

[10] Nunes, B., Mendonca, M., Nguyen, N., Obraczka, K., & Turletti, T. (2014). A survey of software-defined networking: past, present and future of programmable networks. *Communications Surveys and Tutorials, IEEE*, 1617-1634.

[11] Rudolf, F., Peter, H., Peter, G., Gabor, A., Denes, B., & Tibor, G. (2019). Challenging Machine Learning algorithms in Predicting Vulnerable Javascript Functions. *7th International Workshop on Realizing Artificial Intelligence synergies in Software Engineering.* Szeged: University of Szeged.

[12] Rista, A., Ajdari, J., & Zenuni, X. (2020). Predicting and analyzing Absenteeism at Workplace Using Machine Learning Algorithms. *43rd International Convention on Information, Communication and Electronic Technology* , 485-490

[13] Saurav, N., Faheem, Z., Zasimer, D., Eric, W., & Baijian, Y. (2016). Predicting Network attack Patterns in SDN using Machine Learning Approach. *IEEE Conference on Network Function Virtualization and Software Defined Networks.* New York

[14] Sommer, V. (2014). *Anomaly Detection in SDN Control Plane.* Munich: Master's thesis, Technical University of Munich

[15] Ullman, S., Poggio, T., Harari, D., Zysman, D., & Seibert, D. (2020). Unsupervised learning - Clustering. *Center for Brains, Minds and Machines*

Short Biography:

Dotun Ogundare from Nigeria, earned his masters degree at the Faculty of Technical Sciences of the University of Novi Sad in October 2022 and subsequently enrolled in his PhD. His current interests include distributed computing, Software Defined Networking, Systems Dynamics, digital culture and technology determinism.