The Effect of Hyperparameter Optimization on the Estimation of Performance Metrics in Network Traffic Prediction using the Gradient Boosting Machine Model

Machoke Mwita
School of Computational and Communication Science and Engineering, Department of Information Technology Development and Management (ITDM), The Nelson Mandela African Institution of Science and Technology, Tanzania
machokem@nm-aist.ac.tz (corresponding author)

Jimmy Mbelwa
University of Dar es Salaam, Tanzania
jimmymbelwa@gmail.com

Johnson Agbinya
School of Information Technology and Engineering, Melbourne Institute of Technology, Australia
jagbinya@mit.edu.au

Anael Elikana Sam
School of Computational and Communication Science and Engineering (CoCSE), Department of Communication Science and Engineering (CoSE), The Nelson Mandela African Institution of Science and Technology, Tanzania
anael.sam@nm-aist.ac.tz

Received: 7 December 2022 | Revised: 10 February 2023 | Accepted: 18 February 2023

Licensed under a CC-BY 4.0 license | Copyright (c) by the authors | DOI: https://doi.org/10.48084/etasr.5548

ABSTRACT

Information and Communication Technology (ICT) has changed the way we communicate and access information, resulting in the high generation of heterogeneous data. The amount of network traffic generated constantly increases in velocity, veracity, and volume as we enter the era of big data. Network traffic classification and intrusion detection are very important for the early detection and identification of unnecessary network traffic. The Machine Learning (ML) approach has recently entered the center stage in network traffic accurate classification. However, in most cases, it does not apply model hyperparameter optimization. In this study, gradient boosting machine prediction was used with different hyperparameter optimization configurations, such as interaction depth, tree number, learning rate, and sampling. Data were collected through an experimental setup by using the Sophos firewall and Cisco router data loggers. Data analysis was conducted with R software version 4.2.0 with Rstudio Integrated Development Environment. The dataset was split into two partitions, where 70% was used for training the model and 30% for testing. At a learning rate of 0.1, interaction depth of 14, and tree number of 2500, the model estimated the highest performance metrics with an accuracy of 0.93 and R of 0.87 compared to 0.90 and 0.85 before model optimization. The same configuration attained the minimum classification error of 0.07 than 0.10 before model optimization. After model tweaking, a method was developed for achieving improved accuracy, R square, mean decrease in Gini coefficients for more than 8 features, lower classification error, root mean square error, logarithmic loss, and mean square error in the model.

Keywords-network traffic; machine learning; big data; data loggers; feature selection; gradient boosting machine prediction

I. INTRODUCTION

To provide secure and successful communications in any organization at the local, national, and global levels, network management of both incoming and outgoing traffic is critical. The Internet, Wide Area Networks (WANs), Local Area Networks (LANs), e-Commerce, and other online services are widely used in businesses for communication and transaction purposes. Network connectivity is essential for the efficient operation of firms or the supply of daily services to their clientele [1]. Network management tasks, including classification and monitoring of network traffic are crucial for Quality of Service (QoS) control, anomaly detection, and other activities related to network troubleshooting [2]. Voice, data, and multimedia services currently can be merged into one application/software thanks to the advancements in the telecommunications and application/software industries. Several protocols have been developed to support file transfer within the network such as User Datagram Protocol (UDP) and Transmission Control Protocol (TCP) through the Internet. During the early stage of the Internet, static ports (SSH, HTTP, FTP, SMTP, etc.) were used by network security experts for the classification and mapping of network traffic [3]. The Internet has changed the development of an application because they can work without using the assigned port numbers e.g. in peer-to-peer (P2P), gaming, and other online applications [4]. Due to the diversification of software applications that require a different level of bandwidth usage, it is very crucial to be able to identify applications that consume more resources in the network, breach network security, and generate unnecessary traffic. Prediction of network traffic in advance can assist organizations in improving their service provision, both internally and externally by changing the network design or improving areas with weaknesses [5]. This can be done by implementing new network architectures or by deploying equipment with higher capacity.

The advancement of gadgets used in Information and Communication Technology (ICT), particularly smartphones, poses a new challenge to information access and sharing. Currently, most services, such as social media networks and banking are accessed via smartphones [6]. Accessing the services through smartphones generates traffic which may breach the security of the organization and lower QoS provision. With the growth of network complexity, network specialists and engineers employ a variety of strategies and alternatives to identify and optimize network traffic. Machine Learning (ML) is one of them. ML is a branch of Artificial Intelligence (AI) that enables computers to learn from previous data and improve through experience [7]. There are 4 categories of ML algorithms, namely supervised, unsupervised, semi-supervised, and reinforcement. The ML approach has been applied in several fields such as finance, healthcare, ICT (i.e. data anomaly detection, information security, etc.), image recognition, fraud detection, and others, while some companies like Facebook, Google, Apple, Netflix, and Amazon use ML in their daily business operations.

In ICT data security, ML can be used to classify network traffic from different sources such as Internet of Things (IoT) devices, wireless sensors, LAN, and WAN and identify threats or malicious activities. Different ML methods or techniques have been applied in the classification of network traffic to supplement existing network management tools, for instance, the k-nearest neighbors algorithm (k-NN), Support Vector Machine (SVM), and clustering. Although ML has advanced to the forefront of effectively identifying network traffic, it rarely employs model hyperparameter optimization procedures in this process. Network traffic classification and optimization are very crucial to attain the maximum usage of the communication network infrastructure. This will assist an organization to avoid investing a lot of money without getting value for money or a return on investment. Understanding network traffic flow is very important for both ICT experts and the organization for planning human and financial resource alignment.

In [8], the XGBoost algorithm was used in the classification of network packet trace capture data. Four performance metrics, namely Accuracy, Precision, F1 Score, and Recall were used and the predicted mean accuracy per class was about 99.5%. A Deep Packet Inspection (DPI) study using a Convolution Neural Network (CNN) to classify encrypted packets by using a deep learning approach was conducted in [4]. Authors in [1] showed that a port-based approach can be used to classify flow-based traffic with some limitations, especially in applications that can be accessed using P2P software such as uTorrent, LimeWire, Winny, eDonkey, WinMX, and Transmission. Using adaptive network traffic can be estimated by using probability distributions such as Pareto, log-normal distribution, exponential distribution, Poisson, etc. [9]. However, due to the non-stationary nature or properties of the network traffic, only a non-stochastic mathematical model was applied [10], and a part of the stochastic model known as GBM was used and compared to the adaptive approach.

There are 3 common methods used in classifying network traffic, namely payload-based, flow-based and port-based. Port-based assumes the network traffic uses common port numbers through either UDP or TCP. Network traffic packets in the port-based [11] are inspected by using the source and destination ports of the Internet Protocol (IP) of the package (HTTP(80), SMTP(25), FTP(20, 21), etc.). Another technique that is used to identify malicious activities is the use of DPI, however, it can work only in the encrypted packets and when the packets are not encrypted this approach doesn't work [8]. Authors in [12] showed that the accuracy of the model can be improved by using distance and similarities from the clusters. ML can be used to address some of the drawbacks of DPI and port-based package inspection techniques [13]. Classification of network traffic is done in different ways but the most common one is by estimating performance metrics such as accuracy, area under the curve, sensitivity, kappa values, etc. [14].

Standard metrics like Root Mean Square Error (RMSE) and Mean Square Error (MSE) of deep learning classification techniques kNN and SVM have low performance [15]. In [16], GBM was used for attack detection in traffic flows. A linear regression model was used and the deep learning techniques achieved higher performance compared to the linear regression model.

In this paper, we used GBM which is an ensemble supervised learning technique to classify packet flows, because, after a thorough literature review and to the best of our knowledge, there are no studies related to GBM hyperparameter tuning in network traffic prediction. The model was optimized by using different hyperparameter configurations, such as interaction depth, number of trees, learning rate, sampling, etc. to estimate the prediction performance metrics Accuracy, RMSE, MSE, Logarithmic Loss (Log loss), and R^2. The performance metrics were compared with the default GBM performance metrics.

II. MATERIALS AND METHODS

Network traffic data were collected through an experiment set by using Cisco Netflow (flow viewer) and Sophos firewall. Cisco 4000 Series and Sophos XGS 3100 1U Model were used for the experiment setup (Figure 1). Data for both incoming and outgoing network traffic were recorded in log files. The data were downloaded and stored on a computer in csv and txt formats. A detailed process from data collection up to model evaluations is shown in Figure 1.

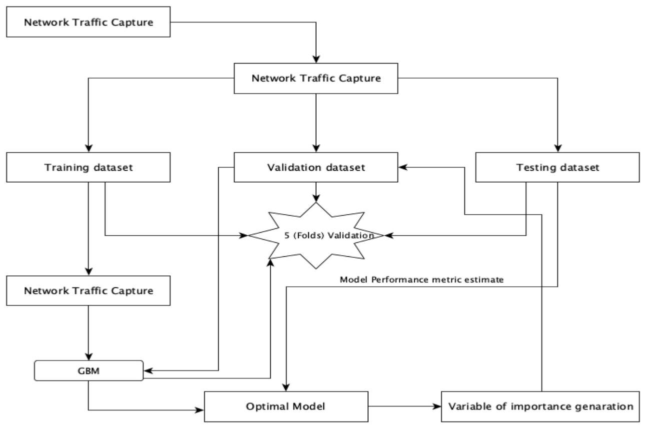

Fig. 1. Data collection up to model comparison flows.

A detailed description of the GBM input and the classification and optimization steps can be seen in Figure 1. The main steps are:

1. Dataset captured in the experiment.
2. Convert categorical features (labels column) to numerical values (benign, malicious, and outliers).
3. Normalize the dataset by imputing median to all features except the label feature which is categorical with multi labels.
4. Partition the dataset into training and test subsets.
5. Build the GBM classifier model by using the training dataset.
6. Validate GBM by using the test dataset with 5-fold classification based on 3 features (benign, malicious, and outliers).
7. Performance evaluation.
8. Model hyperparameter tuning (optimization).
9. Output: Accuracy, RMSE, MSE, Log loss, and R^2.

A. Data Pre-Processing

Network traffic data that were collected were converted into csv files with 18 columns with 3 feature classes (benign, malicious, and outliers). The PreProces package in R software [16] and Rstudio Integrated Development Environment (IDE) [17] were used to normalize and scale the data. Before data scaling, duplicate data were deleted. After duplication removal and normalization, the data were converted into an H2O package [17] format, so that they can be run in a distributed and low memory management H2O package on top of the R software [18, 19]. The dataset was partitioned into two subsets, 70% model training and 30% for testing. Open-source H20 software which is embedded in R software was used to build both models. H2O uses in-memory cluster computing and distributed systems and supports both supervised and unsupervised ML algorithms [20]. In-memory usage can compress the dataset by using columns to increase the classification speed and time in cluster computing [21]. H20 builds all the models in a sequence where all the data are fully distributed in the memory.

B. Model Development

GBM framework was developed in [22] as an additive model which uses a stage wise decision tree. GBM is a technique used in developing very powerful predictive models in ML [23]. GBM is an ensemble learning model which can work for regression and classification [24]. Ensemble learning merges weak learners into strong learners to improve their performance [25] by using the recursive tree partitioning approach. GBM is a forward learning ensemble technique that uses decision trees. Before model tuning, the model was developed and feature importance (variable importance) was estimated. The importance variable provides the contribution of each variable to model development and its output. The values extracted from the model identify and describe all features that are relevant to the model. GBM aims to optimize a generalized ensemble model [26] as indicated in Figure 1 and in the detailed process in Algorithm 1. GBM always minimizes the loss function so that it can be optimized during model generations. The model was developed by using the H2O GBM framework. H20 GBM framework [21] is a package in R software version 2.4.0 [17] and Rstudio version 2022.02.2 [27] was used as an IDE. This study used R packages eXtreme Gradient Boosting (XGBoost), GBM and Classification and Regression Training (CARET) to implement GBM models [28] by using different parameter configurations (Table I).

```
Algorithm 1: Pseudocode based on [29].
Input D = $\{(x_1, y_1), (x_2, y_2), \ldots\ldots, (x_N, y_N)\}, L(y, O(x))$
Begin
Initialize: $O_o(x) = \mathrm{argmin}_w \sum_{i=1}^{n} L(y_i, w)$
for m = 1:M
$r_{im} = -\frac{L(y_i, O(x_i))}{O(x_i)}$
Train weak learner $C_m(x)$ on training data
```

```
Calculate w: $w_m = argmin_w \sum_{i=1}^{N} L(y_i, O_{m-1} + (x_i) + wC_{m(x_i)})$
Update: $O_{m(x)} = O_{m-1} + w_m C_m(x)$
End for
End
```

TABLE I. PARAMETER CONFIGURATION OF THE PROPOSED APPROACH

Parameter	Values
Algorithm	GBM
Feature class	Categorical
Interaction depth	1 to17
Learning rate	0.01, 0.1, 0.2, 0.3, 0.4
Sample rate	0.4, 0.8, 1.0
Column sample	0.4, 0.6, 1.0
Number of folds	5
Bagging fraction	0.8, 1.0
Number of trees	100, 1000, 2500, 5000

C. Tuning Process

Hyperparameter tuning by using the Grid search method was applied in model development, and a combination of different techniques was used for all values of the hyperparameters provided Figure 2. The evaluation of the model was based on the best performance metrics estimated by using different parameters and architectures. Different values of number of trees were used to evaluate the model by changing the number of trees (*ntrees*) as described in Figure 2. The *ntrees* = {100, 1000, 2500, and 5000} were used with the learning rate values of {0.01, 0.1, 0.3, 0.4}. The interaction depth was set and varied from 1 up to 17 to optimize the model and attain maximum accuracy. Interaction depth (maximum depth) which provides maximum accuracy was selected with its learning rate for model training. The sampling rate values were {0.4, 0.8, and 1.0} during model optimization (Table I).

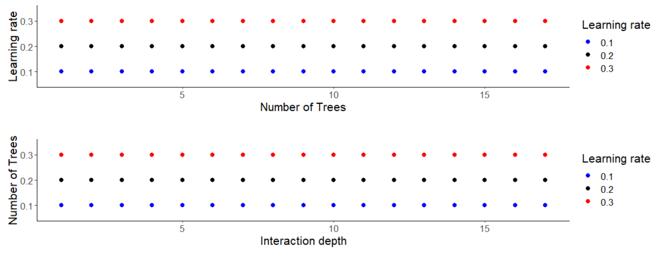

Fig. 2. Graph of model turning parameters.

D. Performance Evaluation

The model was turned using two types of parameter tuning: boosted categories and tree specific parameters [30]. Boosted tuning includes *ntree*, learning rate, and sampling rate, while tree specifics are interaction depth, minimum samples, and feature to split. Five-fold cross-validation technique was used to evaluate the predictive performance of the model. To evaluate the turning parameters in Table I, the following model performance evaluation metrics were used.

1) Accuracy

In multiclass classification, the set of labels predicted for a sample must exactly match the corresponding set of labels in the feature classes. In ML, model accuracy is a common performance metric which considers the features that are correctly classified by the model.

$$\text{Accuracy} = \frac{(\text{TP+TN})}{(\text{TP+FP+TN+FN})} \ldots\ldots\ldots\ldots \qquad (1)$$

where TP, FP, TN, and FN represent True Positive, False Positive, True Negative, and False Negative, respectively.

2) Logarithmic Loss

Log loss is a performance metric used when the input feature is continuous or categorical. In this study we used categorical features, namely benign, malicious, and outliers. Log loss shows how the probability of the predicted values is close to the actual values. When the predicted probability diverges from the true values, the Log loss values get higher. Log loss can be expressed mathematically as:

$$\text{Log loss} = \frac{1}{N}\sum_{i=1}^{N} \sum_{j=1}^{N}\left(wi(y_{i \cdot j} \ln(p_{i,j}))\right) \qquad (2)$$

where *N* is the total number of rows (observations) of our corresponding data frame, *w* is the per-row user-defined weight (default is 1), *C* is the total number of classes (*C*=2 for binary classification), *p* is the predicted value (uncalibrated probability) assigned to a given row (observation), and *y* is the actual target value.

3) Root Mean Squared Error (RMSE)

RMSE is the error between the predicted values and the observed values [31] as indicated below:

$$\text{RMSE} = \frac{1}{N}\sum_{i=1}^{N}(yi - \hat{y}i)^{2} \quad \ldots \qquad (3)$$

where *N* is the total number of rows of the corresponding data frame, *y* is the actual target value, and $\hat{y}$ is the predicted target value.

4) R Squared (R^2)

R^2 is defined as the variance proportional to the dependent variable which is predicted from the independent variable. It is defined by:

$$R^2 = \frac{SS_{reg}}{SS_{tot}} = \frac{\sum_j (y_j - \bar{y})^2}{\sum_i (y_i - \bar{y})^2} \qquad (4)$$

where y_j represents the predicted labels, y_i the true labels, SS_{reg} is the regression Sum of Squares, and SS_{tot} the total Sum of Squares.

III. RESULTS

Several configurations were used during model optimization. The highest accuracy and R^2 values were attained for *ntree* =2500, interaction depth = 14, and learning rate = 0.1 as indicated in Figures 3 and 4. At the same parameter configuration, minimum classification error, RMSE, Log loss, and MSE were also achieved, as indicated in Figure 4. The contribution of an individual attribute in the model was ranked by using relative importance values as shown in Figure 6.

A. Before Model Tuning

Before model tuning, the model was developed and executed with the 5-fold cross-validation technique. The performance metrics were estimated as shown in Figure 3. The Accuracy values ranged from 0.897 to 0.902. Accuracy was maximum at the second fold (0.899). R^2 at the second fold was maximum (0.856) and minimum at the first fold (0.851). The overall estimate for R^2 was about 0.853. There is an inverse relationship between MSE and R^2 as indicated in Figure 2. As the estimated values of MSE increased, there was a decrease in the estimated values of R^2. The values of Accuracy increased until when it reached 0.90 and it stabilized at 0.899 as shown in Figure 3. The interaction depth of 14 was the one that attributed the model to attain maximum Accuracy (Figure 3). The values of R^2 were at increasing trends until they reached 0.86, then started to stabilize and remained constant while the value of RMSE was increasing.

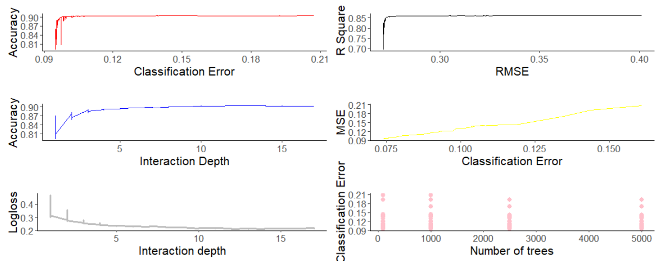

Fig. 3. Graphic representation of the performance metrics before model tuning.

B. After Model Tuning

The second phase of the model was to tune the model by using the parameters indicated in Table I and Figure 1. The values of Accuracy increased from 0.899-0.90 to 0.920-0.95 in the 5-fold cross-validation as indicated in Figure 4. After model tuning, the values of Log loss decreased to about 0.21 from 0.23 before tuning, as shown in Figure 4. The estimates of MSE decreased from 0.08 to 0.07 after model tuning, R^2 increased up to the maximum value of about 0.88 after model tuning from about 0.85 before model tuning. The estimated metrics for RMSE and classification error decreased after model tuning (Figure 2).

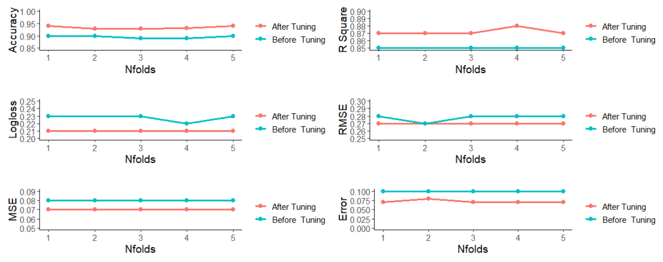

Fig. 4. Performance metrics estimation at 5 – fold cross-validation.

C. Result Comparison before and after Model Tuning

The overall Accuracy before model tuning was about 0.90, however, after model tuning, it was improved up to 0.930 (Figure 5). The value of R^2 after model tuning was higher compared to the one before which varied from 0.85 to 0.87. The values of MSE decreased from 0.08 to 0.07 after hyperparameter tuning. Classification error estimated values decreased as well from about 0.10 before model tuning to 0.08 after model tuning and RMSE values also decreased from 0.28 to about 0.25 (Figure 5). Overall, there is an improvement in all performance metrics after model tuning.

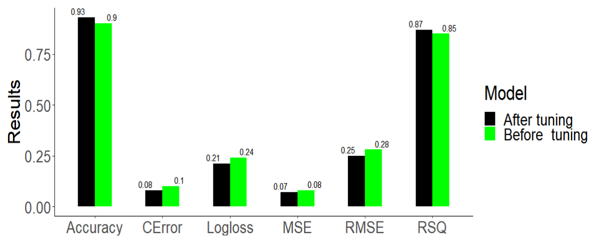

Fig. 5. Graph showing the overall performance metrics before and after tuning.

D. Comparison with Other Studies

The results of this study were compared with other studies as indicated in Table II. The accuracy generated in our study is higher than those reported in [12, 30, 33], while the results of 0.93 [34, 35] and 0.94 [36, 37] are equivalent or above this study's results. The estimated values of R^2 of this study lie between the minimum R^2 as per [33] and below [38]. RMSE was also supported by other studies [16, 30, 39, 40]. MSE from previous studies varied from 0.053 [32] up to 0.4612 [39] while the result of this paper is 0.074 which is supported by other studies as well. The findings of this study are supported by other studies as detailed in Table II.

TABLE II. COMPARISON WITH OTHER STUDIES

	Performance Metrics			
Reference	**Accuracy**	**MSE**	**RMSE**	**R^2**
[39]		0.4612	0.2127	
[38]			13830	0.887
[16]		0.41	0.480	0.836
[12]	0.902	0.280	0.340	
[32]	0.907	0.053	0.229	0.780
[33]	0.746	0.259	0.509	0.769
Current study	0.930	0.074	0.272	0.872

E. Variable of Importance

Features of importance were compared in all phases of the model. Before tuning the model, the overall entropy contributed more than 50% to the model, while after tuning it only provided about 30%. Before model tuning, most of the feature's contributions were below 5% except for Bytes out and Total entropy. The time between the traffic inters into the network and when it ends is about 1%, showing that the contribution of these two features to the model is very minimal even after model tuning. After the model tuning process, there was an improvement in the contribution of individual features to the model. The following features improved up to more than 10% their contribution to the model: Number of packets in, Duration, Bytes out, Total entropy, and Average inter packet time. The contribution of Time start and Time end was about 1% to 2%. Therefore, their contribution to the model is not very important (Figure 6).

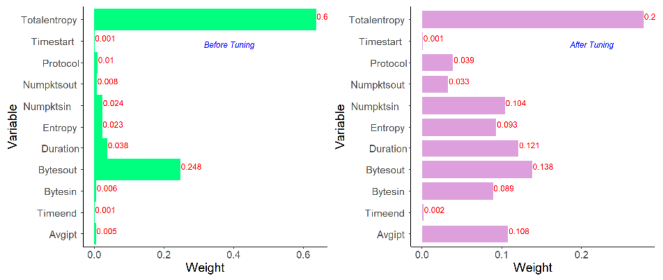

Fig. 6. Graph showing the importance variation before and after tuning.

IV. CONCLUSION, RECOMMENDATIONS, AND FUTURE WORK

The results from this study show that by using hyperparameter optimization (tuning), the Accuracy of the model improved from 0.90 to 0.93. Classification errors were reduced from 0.10 to about 0.07 after model tuning. The estimated values of R^2 were 0.85 before tuning and 0.87 after tuning. Based on these findings, the use of hyperparameter tuning is proposed because it improves several performance metrics. The proposed approach increased the Accuracy, R^2, and Mean of network traffic classification variables and lowered classification error. Furthermore, different machine learning approaches can be used to improve the performance of the model like artificial neural networks, deep learning, and others.

The current study can be extended and used in different conditions in order to extract more variables of importance based on different parameters and experiments. More research is needed in the area of the comparison of the hardware-based approach and machine learning. This study contributes scientific knowledge related to network traffic management and can be used by any organization or Internet service provider to manage network traffic.

FUNDING

The current study was funded by the European Union's Horizon 2020 research and innovation grant number [641918] through the AFRICANBIOSERVICES PROJECT in Tanzania under Tanzanian Wildlife Research Institute (TAWIRI). The authors extend their appreciation to the reviewers for their valuable remarks, Tanzania Communications Regulatory Authority (TCRA), TAWIRI, and Tanzania National Parks (TANAPA) for their support during data collection, and NM-AIST for the conducive working and research environment.

DATA AVAILABILITY STATEMENT

The data used in this study will be available upon request from the corresponding author.

REFERENCES

[1] M. Soysal and E. G. Schmidt, "Machine learning algorithms for accurate flow-based network traffic classification: Evaluation and comparison," *Performance Evaluation*, vol. 67, no. 6, pp. 451–467, Jun. 2010, https://doi.org/10.1016/j.peva.2010.01.001.

[2] J. J. Estevez-Pereira, D. Fernandez, and F. J. Novoa, "Network Anomaly Detection Using Machine Learning Techniques," *Proceedings*, vol. 54, no. 1, 2020, Art. no. 8, https://doi.org/10.3390/proceedings2020054008.

[3] G. Ali, M. Ally Dida, and A. Elikana Sam, "Two-Factor Authentication Scheme for Mobile Money: A Review of Threat Models and Countermeasures," *Future Internet*, vol. 12, no. 10, Oct. 2020, Art. no. 160, https://doi.org/10.3390/fi12100160.

[4] M. Lotfollahi, M. Jafari Siavoshani, R. Shirali Hossein Zade, and M. Saberian, "Deep packet: a novel approach for encrypted traffic classification using deep learning," *Soft Computing*, vol. 24, no. 3, pp. 1999–2012, Feb. 2020, https://doi.org/10.1007/s00500-019-04030-2.

[5] K. Demertzis, K. Tsiknas, D. Takezis, C. Skianis, and L. Iliadis, "Darknet Traffic Big-Data Analysis and Network Management for Real-Time Automating of the Malicious Intent Detection Process by a Weight Agnostic Neural Networks Framework," *Electronics*, vol. 10, no. 7, Jan. 2021, Art. no. 781, https://doi.org/10.3390/electronics10070781.

[6] G. S. Oreku, F. J. Mtenzi, and C. A. Shoniregun, "Traffic classification and packet detections to facilitate networks security," *International Journal of Internet Technology and Secured Transactions*, vol. 3, no. 3, pp. 240–252, Jan. 2011, https://doi.org/10.1504/IJITST.2011.041294.

[7] Q. Bi, K. E. Goodman, J. Kaminsky, and J. Lessler, "What is Machine Learning? A Primer for the Epidemiologist," *American Journal of Epidemiology*, vol. 188, no. 12, pp. 2222–2239, Dec. 2019, https://doi.org/10.1093/aje/kwz189.

[8] I. L. Cherif and A. Kortebi, "On using eXtreme Gradient Boosting (XGBoost) Machine Learning algorithm for Home Network Traffic Classification," in *Wireless Days*, Manchester, UK, Apr. 2019, pp. 1–6, https://doi.org/10.1109/WD.2019.8734193.

[9] S. Ageev, V. Karetnikov, E. Ol'khovik, and A. Privalov, "Adaptive method of detecting traffic anomalies in high-speed multi-service communication networks," *E3S Web of Conferences*, vol. 157, 2020, Art. no. 04027, https://doi.org/10.1051/e3sconf/202015704027.

[10] J. K. Mazima, A. Johnson, E. Manasseh, and S. Kaijage, "Stochastic Modeling Technology for Grain Crops Storage Application : Review," *International Journal of Artificial Intelligence & Applications*, vol. 7, no. 6, pp. 27–42, Nov. 2016, https://doi.org/10.5121/ijaia.2016.7603.

[11] M. Singh, G. Srivastava, and P. Kumar, "Internet Traffic Classification Using Machine Learning," *International Journal of Database Theory and Application*, vol. 9, pp. 45–54, Dec. 2016, https://doi.org/10.14257/ijdta.2016.9.12.05.

[12] R. Samrin and D. Vasumathi, "Hybrid Weighted K-Means Clustering and Artificial Neural Network for an Anomaly-Based Network Intrusion Detection System," *Journal of Intelligent Systems*, vol. 27, no. 2, pp. 135–147, Apr. 2018, https://doi.org/10.1515/jisys-2016-0105.

[13] F. Dehghani, N. Movahhedinia, M. R. Khayyambashi, and S. Kianian, "Real-Time Traffic Classification Based on Statistical and Payload Content Features," in *2nd International Workshop on Intelligent Systems and Applications*, Wuhan, China, Dec. 2010, pp. 1–4, https://doi.org/10.1109/IWISA.2010.5473467.

[14] J. Yang, Y.-X. Wang, Y.-Y. Qiao, X.-X. Zhao, F. Liu, and G. Cheng, "On Evaluating Multi-class Network Traffic Classifiers Based on AUC," *Wireless Personal Communications*, vol. 83, no. 3, pp. 1731–1750, Aug. 2015, https://doi.org/10.1007/s11277-015-2473-4.

[15] A. M. Sadeghzadeh, S. Shiravi, and R. Jalili, "Adversarial Network Traffic: Towards Evaluating the Robustness of Deep-Learning-Based Network Traffic Classification," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1962–1976, Jun. 2021, https://doi.org/10.1109/TNSM.2021.3052888.

[16] A. Gouveia and M. Correia, "Deep Learning for Network Intrusion Detection: An Empirical Assessment," in *Recent Advances in Security, Privacy, and Trust for Internet of Things (IoT) and Cyber-Physical Systems (CPS)*, 1st Edition., Boca Raton, FL, USA: Chapman and Hall, 2020, pp. 191–206.

[17] "R Interface for the 'H2O' Scalable Machine Learning Platform," *H2O*. https://docs.h2o.ai/h2o/latest-stable/h2o-r/docs/index.html.

[18] "R: The R Project for Statistical Computing." https://www.r-project.org/.

[19] I. Satti, A. Elkarim, J. Agbinya, A. Hussein, and I. Satti, "Parallel SVM Based Classification Technique on big data: HPC center in Sudan," *Australian Journal of Basic and Applied Sciences*, vol. 14, pp. 1–14, Apr. 2020, https://doi.org/10.22587/ajbas.2020.14.4.1.

[20] A. Malik *et al.*, "Deep learning versus gradient boosting machine for pan evaporation prediction," *Engineering Applications of Computational Fluid Mechanics*, vol. 16, no. 1, pp. 570–587, Dec. 2022, https://doi.org/10.1080/19942060.2022.2027273.

[21] D. Cook, *Practical Machine Learning with H2O: Powerful, Scalable Techniques for Deep Learning and AI*, 1st ed. O'Reilly Media, 2016.

[22] J. H. Friedman, "Stochastic gradient boosting," *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367–378, Feb. 2002, https://doi.org/10.1016/S0167-9473(01)00065-2.

[23] A. Natekin and A. Knoll, "Gradient Boosting Machines, A Tutorial," *Frontiers in Neurorobotics*, vol. 7, 2013, Art. no. 21, https://doi.org/10.3389/fnbot.2013.00021.

[24] E. A. Freeman, G. G. Moisen, J. W. Coulston, and B. T. Wilson, "Random forests and stochastic gradient boosting for predicting tree canopy cover: comparing tuning processes and model performance," *Canadian Journal of Forest Research*, vol. 46, no. 3, pp. 323–339, Mar. 2016, https://doi.org/10.1139/cjfr-2014-0562.

[25] M. Machoke, J. Mbelwa, J. Agbinya, and A. E. Sam, "Performance Comparison of Ensemble Learning and Supervised Algorithms in Classifying Multi-label Network Traffic Flow," *Engineering, Technology & Applied Science Research*, vol. 12, no. 3, pp. 8667–8674, Jun. 2022, https://doi.org/10.48084/etasr.4852.

[26] M. Alqahtani, A. Gumaei, H. Mathkour, and M. Maher Ben Ismail, "A Genetic-Based Extreme Gradient Boosting Model for Detecting Intrusions in Wireless Sensor Networks," *Sensors*, vol. 19, no. 20, Jan. 2019, Art. no. 4383, https://doi.org/10.3390/s19204383.

[27] J. J. Allaire, "RStudio: Integrated Development Environment for R," presented at the The R User Conference 2011, Coventry, UK, 2011.

[28] C.-W. Wu, H.-L. Shen, C.-J. Lu, S.-H. Chen, and H.-Y. Chen, "Comparison of Different Machine Learning Classifiers for Glaucoma Diagnosis Based on Spectralis OCT," *Diagnostics*, vol. 11, no. 9, Sep. 2021, Art. no. 1718, https://doi.org/10.3390/diagnostics11091718.

[29] J. H. Friedman, "Greedy Function Approximation: A Gradient Boosting Machine," *The Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001, https://doi.org/10.1214/aos/1013203451.

[30] R. Andersson, *Classification of Video Traffic : An Evaluation of Video Traffic Classification using Random Forests and Gradient Boosted Trees*. Karlstad, Sweden: Karlstad University, 2017.

[31] H. Wan, "Gradient Descent Boosting: Convergence and Algorithm," 2017, [Online]. Available: https://courses.engr.illinois.edu/ece543/sp2017/projects/Haohua%20Wan.pdf.

[32] D. Preethi and N. Khare, "Sparse auto encoder driven support vector regression based deep learning model for predicting network intrusions," *Peer-to-Peer Networking and Applications*, vol. 14, no. 4, pp. 2419–2429, Jul. 2021, https://doi.org/10.1007/s12083-020-00986-3.

[33] C. Pan, Y. Wang, H. Shi, J. Shi, and R. Cai, "Network Traffic Prediction Incorporating Prior Knowledge for an Intelligent Network," *Sensors*, vol. 22, no. 7, Jan. 2022, Art. no. 2674, https://doi.org/10.3390/s22072674.

[34] L.-H. Chang, Tsung-Han Lee, Hung-Chi Chu, and Cheng-Wei Su, "Application-Based Online Traffic Classification with Deep Learning Models on SDN Networks," *Advances in Technology Innovation*, vol. 5, no. 4, pp. 216–229, Jul. 2020, https://doi.org/10.46604/aiti.2020.4286.

[35] R. Dangi, A. Jadhav, G. Choudhary, N. Dragoni, M. K. Mishra, and P. Lalwani, "ML-Based 5G Network Slicing Security: A Comprehensive Survey," *Future Internet*, vol. 14, no. 4, Apr. 2022, Art. no. 116, https://doi.org/10.3390/fi14040116.

[36] Z. A. Qazi, J. Lee, T. Jin, G. Bellala, M. Arndt, and G. Noubir, "Application-awareness in SDN," in *ACM SIGCOMM 2013 conference on SIGCOMM*, New York, NY, USA, Aug. 2013, pp. 487–488, https://doi.org/10.1145/2486001.2491700.

[37] S. S. Alzahrani, "Data Mining Regarding Cyberbullying in the Arabic Language on Instagram Using KNIME and Orange Tools," *Engineering, Technology & Applied Science Research*, vol. 12, no. 5, pp. 9364–9371, Oct. 2022, https://doi.org/10.48084/etasr.5184.

[38] Q. H. Do, T. T. H. Doan, T. V. A. Nguyen, N. T. Duong, and V. V. Linh, "Prediction of Data Traffic in Telecom Networks based on Deep Neural Networks," *Journal of Computer Science*, vol. 16, no. 9, pp. 1268–1277, Sep. 2020, https://doi.org/10.3844/jcssp.2020.1268.1277.

[39] S. Mahajan, R. Harikrishnan, and K. Kotecha, "Prediction of Network Traffic in Wireless Mesh Networks Using Hybrid Deep Learning Model," *IEEE Access*, vol. 10, pp. 7003–7015, 2022, https://doi.org/10.1109/ACCESS.2022.3140646.

[40] K. P. Rusna and V. G. Kalpana, "Using Artificial Neural Networks for the Prediction of the Compressive Strength of Geopolymer Fly Ash," *Engineering, Technology & Applied Science Research*, vol. 12, no. 5, pp. 9120–9125, Oct. 2022, https://doi.org/10.48084/etasr.5185.