Hybridized Decision Tree methods for Detecting Generic Attack on Ciphertext

Yazan Ahmad Alsariera[1†],

Department of Computer Science, Collage of Science, Northern Border University,
Arar 73222, Kingdom of Saudi Arabia

Summary
The surge in generic attacks execution against cipher text on the computer network has led to the continuous advancement of the mechanisms to protect information integrity and confidentiality. The implementation of explicit decision tree machine learning algorithm is reported to accurately classifier generic attacks better than some multi-classification algorithms as the multi-classification method suffers from detection oversight. However, there is a need to improve the accuracy and reduce the false alarm rate. Therefore, this study aims to improve generic attack classification by implementing two hybridized decision tree algorithms namely Naïve Bayes Decision tree (NBTree) and Logistic Model tree (LMT). The proposed hybridized methods were developed using the 10-fold cross-validation technique to avoid overfitting. The generic attack detector produced a 99.8% accuracy, an FPR score of 0.002 and an MCC score of 0.995. The performances of the proposed methods were better than the existing decision tree method. Similarly, the proposed method outperformed multi-classification methods for detecting generic attacks. Hence, it is recommended to implement hybridized decision tree method for detecting generic attacks on a computer network.

Keywords:
Generic attack, machine learning, hybridized decision tree, cybersecurity.

1. Introduction

Nowadays, the transmission of messages is majorly facilitated through various public and private networks. The internet among other platforms for communication is essential in transmitting information from one host to another destination across the globe [1]. The surge in internet users (i.e. individuals and entities alike) is based on it being a relatively cheap, openly accessible and widely acceptable medium for information transmission [2]. The internet, an example of connected devices, is used by some users to transmitting encrypted information. This information is often sent after being symmetrically encrypted.

Symmetric encryption is the process of converting plaintext information into ciphertext which ensures data confidentiality and integrity [3]. Nonetheless, encrypted messages referred to as ciphers are targets of attackers on a computer network. Cybersecurity as the science of ensuring the protection of data, systems, networks and devices involve developing methods, tools, techniques and technologies to this end [4]. Over the years, the attack of ciphertext has become rampant which leads to improving the encryption mechanism [3]. However, this solution does not cover intercepting ciphertexts on the computer networks. Machine learning (ML) methods have proven to be highly effective in the detection or identification of malicious activities or attacks on networks [5]. This is possible by implementing ML classification algorithms as a predictive model for identifying the type of network packet. Most ML-based predictive models for computer network security are much more focused on either predicting between normal and attack [6], [7] or normal and various popular attacks such Denial of Service (DoS), Probing, SQL injection, Man in the Middle, exploit etc [8], [9]. The later predictive models are referred to as multi-classification, which have received much attention over the years.

In the context of this research, existing multi-classification predictive models for discriminating between a normal network packet and other popular attack types are not effective in detecting generic attacks. Because generic attack possesses similar features with other types of attack such as exploit attack and even normal packets [10]. In effect, this led to the case for developing an explicit predictive model for generic attacks. The use of an explicit generic attack predictive model will accurately ensure generic attack perpetrators are stalled of their interception, unlike multi-classification methods which most time suffer an oversight.

One of such existing multi-classification methods that reported the performance of its method for predicting generic attacks, found in the dataset used by our study, is [11]. Study [11] presented a rule-based multi-classification method for various attack types including generic attacks. The multi-classification method achieved an accuracy of 96.7%, a false positive rate (FPR) of 2.01 for detecting the generic attack found in the dataset. More importantly, generic attacks were seen to be mostly misclassified as normal packet and exploit attacks.

Another multi-classification published by a study [12] used three (3) forms of stacked ensemble methods of three (3) machine learning (i.e. Naïve Bayes (NB), k-Nearest Neighbour (kNN) and Decision Tree (DT)) algorithms to

Manuscript received July 5, 2021
Manuscript revised July 20, 2021
https://doi.org/10.22937/IJCSNS.2021.21.7.6

predict various types of network attacks. Although the performance of the stacked ensemble for detecting generic attacks was not revealed, the base models performances were reported. The NB base model published by [12] predicted generic attacks at 92.63% accuracy, while its kNN had 97.91% accuracy and finally, its DT base model achieved 97.93% accuracy. The performances of various multi-classification methods were comparatively analyzed against our published explicit generic attack detector [13] wherein multi-classification methods were outperformed.
Our previously published work [13, 21] was able to detect generic attacks on a network using the decision tree (DT) based ML method but with some limitations. Our previous work, despite its very high accuracy of 99.6%, had an FPR of 0.004 and an MCC score of 0.991. There is a chance of improvement by further reducing the FPR which in turn increase the overall accuracy and the MCC scores. Therefore, this study is motivated to accomplish this feat by conducting empirical research using hybridized DT methods to detect generic attacks.

This study aims to improve the overall performance of decision tree ML-based method for detecting generic attacks aimed at ciphertext and this study will contribute to the body of knowledge by:

1) Advancing the tree-based ML method for detecting generic attacks on a typical computer network by implementing two hybridized decision trees (i.e., Naïve Bayes DT (NBTree) and Logistic Model Tree (LMT)) algorithms.
2) analyze the performance of hybridized decision tree methods against decision tree methods and multi-classification methods, comparatively.

The remaining section of the paper includes Section 2 which explain the development of the dataset and its processing as used in this study. It also contains a subsection that discusses and reveal the implemented hybridized tree algorithms and pseudocode respectively. Finally, section 2 presents the experimental framework and the experimental setup for conducting this empirical research. Section 3 present the empirical results which include the obtained performances of the proposed methods after fitting and evaluation. Also, section 3 discuss the performance results and present a comparative analysis of the proposed method against the existing decision tree method and multi-classification methods. Conclusively, section 4 provided the conclusion of the study and the future work.

2. Method

2.1 Dataset

The dataset used in this study was extracted from the cybersecurity dataset made available by the study of [14]. The original dataset contained different types of attacks one of which is the focus of this study - cryptographic generic attack.

This study develops its dataset by extracting all instances labelled as 'Generic'. This is a sum of 18,871 generic instances. To create a balanced dataset, a set of 18,954 instances labelled as 'normal' were also extracted. The extracted generic and normal instances were joined together to produce the initial dataset.

The initial dataset was cleaned by removing some variables. There were forty-five (45) original variables. The variable 'id' was removed as it does not serve any related purpose to this study. The variable 'attack_cat' was also removed based on its values that are now irrelevant to this study. Finally, the 'class' variable is being cleaned to now contain two values (i.e. Generic and Normal) representing the types of instances contained in the dataset.

Summarily, the dataset used for developing the proposed classification models contained 42 independent variables and a total of 37,825 instances of both normal and generic attacks executing on a network.

2.2 The Hybridized Tree Algorithms

Since the existing DT methods require improvement having reported a FAR value that is arguably high [13] this study considered two (2) hybridized DT algorithms. These algorithms are Naïve Bayes – Decision Tree (i.e., NBTree) and Logistic Model Tree (i.e., LMT). They were selected because they have different characteristics as a type of hybridized classification algorithm and to foster comparative analysis between both hybridized methods
NBTREE is an induced hybridization of DT and NB algorithm containing usual DT nodes of univariate splits and leaves of NB [15]. It produces interpretable models that scale excellently on large datasets and usually outperform either DT or NB independent models. NBTree was developed to inherit the fast induction of the NB classifier and the tree-like nature of DT. The algorithm for NBTree is presented in Figure 1.

Similarly, LMT produce model that is structurally hierarchical which are made up of root, branches and leaves [16]. LMT is a hybrid of the C4.5 DT algorithm and logistic regression function. The C4.5 algorithm inherent in LMT is responsible for splitting using the information gain ratio calculated for each variable from the variable space. Meanwhile, LMT uses its logistic regression at tree nodes to fit functions for the branches and leaves. Oftentimes, LMT prevents overfitting by using the CART algorithm for pruning [17]. The pseudocode for LMT is depicted in Figure 2.

These hybridized decision tree algorithms were implemented as described in the experimental framework depicted in Figure 3.

The NBTree Algorithm

Input: a set $T$of labelled instances

Output: a decision-tree with naïve-bayes categorizers at the leaves

1. For each attribute X_i, evaluate the utility, $u(X_i)$,of a split on attribute X_i. For continuous attributes, a threshold is also found at this stage.
2. Let $j\_\arg\max_i(u_i)$, i.e., the attribute with the highest utility
3. If u_j is not significantly better than the utility of the current node, create a Naïve-Bayes classifier for the current node and return
4. Partition T according to the test of X_j. If X_j is continuous, a threshold split is used; if X_j is discrete, a multi-way split is made for all possible values.
5. For each child, call the algorithm recursively on the portion of T that matches the test leading to the child.

Figure 1: NBTree algorithm, extracted from [18].

```
LMT(examples){
    root = new Node()
    alpha = getCARTAlpha(examples)
    root.buildTree(examples, null)
    root.CARTprune(alpha)
}

buildTree(examples, initialLinearModels) {
    numIterations =
       CV_Iterations(examples,initialLinearModels)
    initLogitBoost(initialLinearModels)
    linearModels = copyOf(initialLinearModels)
    for i = 1...numIterations
        logitBoostIteration(linearModels,examples)
    split = findSplit(examples)
    localExamples = split.splitExamples(examples)
    sons = new Nodes[split.numSubsets()]
    for s = 1...sons.length
        sons.buildTree(localExamples[s],nodeModels)
}

CV_Iterations(examples,initialLinearModels) {
    for fold = 1...5
        initLogitBoost(initialLinearModels)
        //split into training/test set
        train = trainCV(fold)
        test = testCV(fold)
        linearModels = copyOf(initialLinearModels)
        for i = 1...200
            logitBoostIteration(linearModels,train)
            logErrors[i] += error(test)
    numIterations = findBestIteration(logErrors)
    return numIterations
}
```

Figure 2: Pseudocode for LMT extracted from [19].

2.3 Performance Assessment

The performances of the classification models developed through the hybridized decision tree algorithm were assessed using these metrics:

Overall Accuracy: reveals the percentage of instances that were correctly classified for all class labels [5]. It is usually calculated as depicted in Eq. 1

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \quad (1)$$

FPR is the score of normal network transmitting ciphertext wrongly classified as generic attacks [20].

$$FPR = \frac{FP}{FP+TN} \quad (2)$$

FNR: is the score of generic attacks on cryptographic messages wrongly classified as normal network [10].

$$FNR = \frac{FN}{FN+TP} \quad (3)$$

MCC: is used to measure the quality of classification methods in classifying each transmitting packet as either normal or a generic attack. It is calculated as depicted in Eq. 4.

$$MCC = \frac{TP\ X\ TN - FP\ X\ FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \quad (4)$$

ROC – Area Under Curve is the receiver operating characteristic curve that graphically indicates the models' ability to discriminate between generic attack and normal instances. It calculated as depicted in Eq. 5.

$$AUC = \int_0^1 \frac{TP}{TP+FN} d\frac{FP}{FP+TN} \quad (5)$$

Other metrics used in assessing the performance of the implemented hybridized methods include True positive rate, True Negative rate, F-measure and kappa value etc.

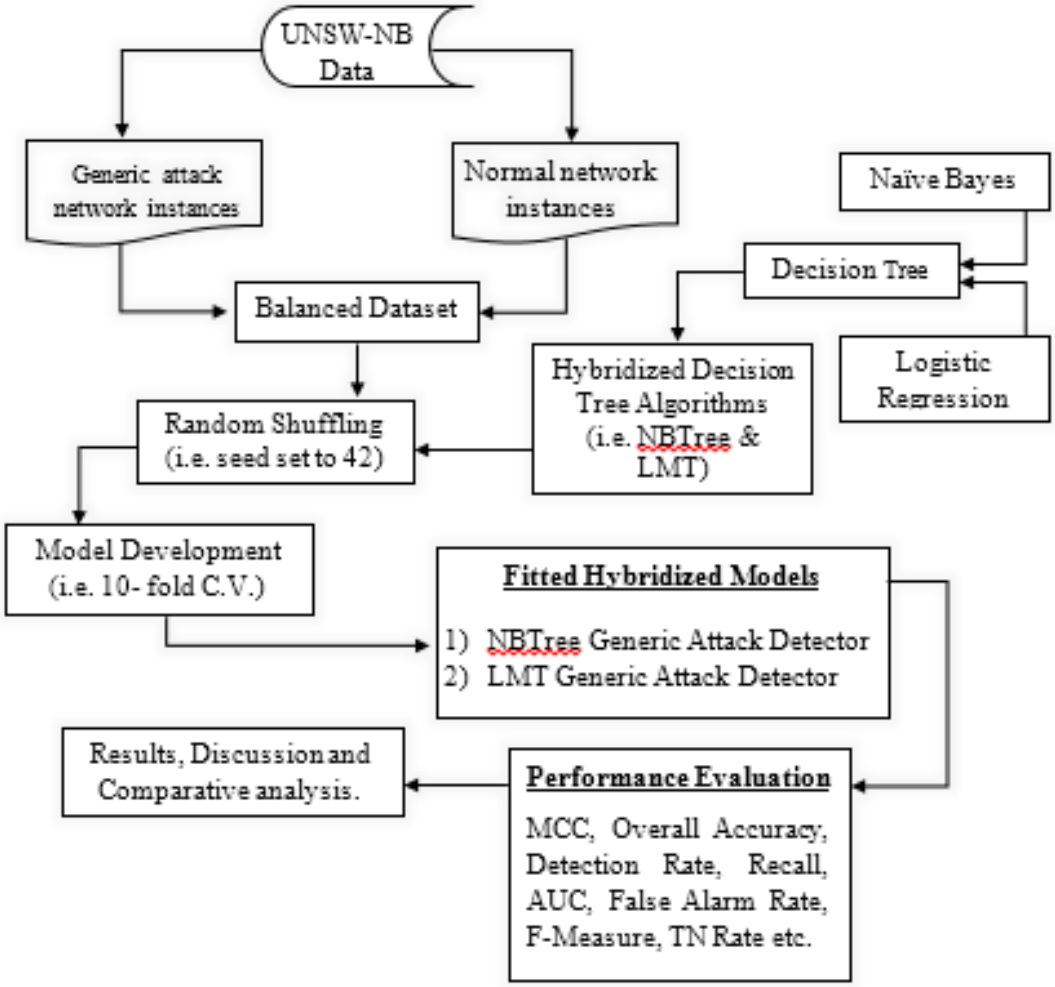

Figure 3: Proposed Experimental Framework

2.4 Experimental Framework and Setup

Figure 3 depicts the detailed activities involved in conducting this empirical study. Firstly, the balanced dataset was developed containing only generic and normal instances from the original data. Secondly, the NBTree and LMT algorithms (results of combining Naïve Bayes and logistic regression with decision tree algorithms, respectively) were implemented with a seed set to 42 to ensure stable model reproduction. Afterwards, hybridized models (i.e. NBTREE and LMT generic attack detectors) were developed by fitting each algorithm on the balanced dataset via 10-fold cross-validation. The 10-fold cross-validation method ensures each partition of the dataset were used for training and testing. This was possible as the dataset was broken into 10 parts, nine of the parts was used for training and the remaining one for testing. This is repeated 10-times until all parts were used for testing. For each iterative testing, the resulting model was evaluated based on some performance evaluation metrics (as depicted in Figure 1). At the end of the iteration, an aggregated performance results were outputted. These performance results are obtained for each fitted hybridized model, they are discussed individually and comparatively analyzed with existing reviewed methods.

The empirical analysis was conducted by using the Waikato Environment for Knowledge Analysis (WEKA) tool. The filter 'RemoveWithValues' was used to extract the normal and generic instances from all other available instances. Under the 'Classify' tab, each hybridized algorithms were selected, the 10-fold cross-validation test options were selected and executed for fitting models.

LMT parameter was set to cross-validate its number of boosting iteration at every node by using a heuristic that considerably decreases its runtime. It was also set to a minimum of 15 instances at which a node can be considered for splitting. On the other hand, NBTree retains the parameters of a typical decision tree for splitting nodes and

a naïve Bayes method for its leaves. Its parameter was set to accept the instances in a batch of 100. The performance of each generic attack detector was assessed based on the metric mentioned and discussed in prevision subsections.
All experiments were conducted using WEKA version 3.8 installed on a personal computer with an Intel Corei5 processor, 8GB of RAM and 500GB of HDD available for paging.

3. Results and Discussions

The results of assessing the performances of the hybridized DT generic attack detectors are presented first for NBTree and then for LMT models respectively.

The performance of the NBTree generic attack detector was assessed using the information obtained in the confusion matrix presented in Table 1.

Table 1: Confusion Matrix of NBTree model.

	Generic	***Normal***
Generic	18827	44
Normal	42	18912

From Table 1, of the 18,871 generic instances, 44 instances were wrongly classified while only 42 of the 18,954 normal instances were misclassified. This results in a 99.77% overall accuracy. Other obtained performance assessment results of NBTree is presented in Table 2. The NBTree generic attack detector achieved a 0.995 MCC score, FPR and FNR of 0.002, and a precision of 0.998. This performance reveals that the hybridized method produced a predictive model with a higher level of prediction way better than chance.

Table 2: NBTree performance scores

Metric	***Score***
Accuracy	99.7726%
Kappa	0.9955
TP Rate	0.998
FP Rate	0.002
TNR	0.998
FNR	0.002
Precision	0.998
Recall	0.998
F-Measure	0.998
MCC	0.995
ROC	0.999

Similarly, the performance of the LMT attack detector was assessed producing the confusion matrix presented in Table 3. This detector is seen to have misclassified 51 generic attacks instances as normal while misclassifying 52 normal instances as generic attacks achieving a 99.73% accuracy.

Table 3: Confusion Matrix of LMT model.

	Generic	***Normal***
Generic	18829	51
Normal	52	18902

Based on the values in the confusion matrix, the LMT model for generic attack detection had a false positive rate of 0.003, MCC of 0.955 and precision score of 0.997 as depicted in Table 4.

Table 4: LMT performance scores

Metric	***Scores***
Accuracy	99.7277
Kappa	0.9946
TP Rate	0.997
FP Rate	0.003
TNR	0.997
FNR	0.003
Precision	0.997
Recall	0.997
F-Measure	0.997
MCC	0.995
ROC	0.999

Based on the performance assessment results, both of the proposed hybridized generic attack detectors were excellent in identifying generic attack instances. However, the insignificantly overall accuracy increase of the NBTree generic attack detector over the LMT variant can be attributed to the conditional probability induced into the DT by the Naïve Bayes algorithm. Equally, the reduced FPR score can be attributed to the algorithmic nature of NB of

treating variables independently unlike the logistic regression method in LMT.

Comparatively, the existing tree-based method [13] for detecting generic attacks had an FPR value of 0.004 which is higher than the 0.002 FPR value of the proposed hybridized NBTree generic attack detector. Also, this study's proposed methods had better performance assessment scores (i.e., accuracy, MCC, ROC and Precision) than the existing tree-based method. Unsurprisingly, hybridizing the decision method with other algorithms as executed by this study improve the performance of generic attack detectors.

The multi-classification method presented by [11] had a FAR of 2.0 while our study achieved 0.002. The study [11] reported an overall accuracy of 65.21% for detecting generic attacks while our study produced 99% accuracy. Another multi-classification DT method presented by [12] had 75% overall accuracy which is also very low compared to the overall accuracy achieved by this study's proposed hybridized DT methods. Empirically, the proposed methods achieved significantly better performance than the existing methods. The proposed hybridized method achieved better performance than DT methods as well as other multi-classification methods for generic attack detection.

4. Conclusion and Future works

This paper aimed at improving the accuracy and reducing the false positive rate of ML-based methods for detecting cipher text or cipher block generic attacks. Particularly, this study improves the ability of the decision tree ML method to detect generic attacks by implementing two hybridized decision tree algorithms. The performances of the proposed LMT and NBTree methods were evaluated via a 10-fold cross-validation technique. The comparative analysis of the performances of the proposed methods reveals they are not only better than the multi-classification methods but also better than single decision tree methods. In the future, consideration of ensemble methods for detecting generic attacks will be made.

References

[1] A. Verma and V. Ranga, "Machine Learning Based Intrusion Detection Systems for IoT Applications," *Wirel. Pers. Commun.*, vol. 111, no. 4, pp. 2287–2310, 2020.

[2] J. Li, Z. Zhao, R. Li, and H. Zhang, "AI-based two-stage intrusion detection for software defined IoT networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2093–2102, 2019.

[3] S. Anindita, S. R. Chatterjee, and M. Chakraborty, "Role of Cryptography in Network Security," in *The "Essence" of Network Security: An End-to-End Panorama*, 2021, pp. 103–143.

[4] A. V. Elijah, A. Abdullah, N. Z. JhanJhi, M. Supramaniam, B. A. O, and O. Balogun Abdullateef, "Ensemble and deep-learning methods for two-class and multi-attack anomaly intrusion detection: An empirical study," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 9, pp. 520–528, 2019.

[5] Y. A. Alsariera, V. E. Adeyemo, A. O. Balogun, and A. K. Alazzawi, "AI Meta-Learners and Extra-Trees Algorithm for the Detection of Phishing Websites," *IEEE Access*, vol. 8, no. August, pp. 142532–142542, 2020.

[6] A. Alsadhan *et al.*, "Locally weighted classifiers for detection of neighbor discovery protocol distributed denial-of-service and replayed attacks," *Trans. Emerg. Telecommun. Technol.*, no. June, pp. 1–15, 2019.

[7] F. Feng, X. Liu, B. Yong, R. Zhou, and Q. Zhou, "Anomaly detection in ad-hoc networks based on deep learning model: A plug and play device," *Ad Hoc Networks*, vol. 84, pp. 82–89, 2019.

[8] Y. Xin *et al.*, "Machine Learning and Deep Learning Methods for Cybersecurity," *IEEE Access*, vol. 6, pp. 35365–35381, 2018.

[9] M. Nawir, A. Amir, N. Yaakob, and O. N. G. B. I. Lynn, "Multi-Classification of Unsw-Nb15 Dataset for Network Anomaly Detection System," *J. Theor. Appl. Inf. Technol.*, vol. 96, no. 15, pp. 5094–5104, 2018.

[10] T. Salman, D. Bhamare, A. Erbad, R. Jain, and M. Samaka, "Machine Learning for Anomaly Detection and Categorization in Multi-Cloud Environments," *Proc. - 4th IEEE Int. Conf. Cyber Secur. Cloud Comput. CSCloud 2017 3rd IEEE Int. Conf. Scalable Smart Cloud, SSC 2017*, pp. 97–103, 2017.

[11] V. Kumar, D. Sinha, A. K. Das, S. C. Pandey, and R. T. Goswami, "An integrated rule based intrusion detection system: analysis on UNSW-NB15 data set and the real time online dataset," *Cluster Comput.*, vol. 23, no. 2, pp. 1397–1418, 2020.

[12] O. O. Olasehinde, "A Stacked Ensemble Intrusion Detection Approach for the Protection of Information System," *Int. J. Infomation Secur. Res.*, vol. 10, no. 1, pp. 910–923, 2020.

[13] Y. A. Alsariera, "Detecting Generic Network Intrusion Attacks using Tree-based Machine Learning Methods," *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 2, pp. 597–603, 2021.

[14] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *2015 Military Communications and Information Systems Conference, MilCIS 2015 - Proceedings IEEE*, 2015, pp. 1–6.

[15] T. Hamed, J. B. Ernst, and S. C. Kremer, "A Survey and Taxonomy of Classifiers of Intrusion Detection Systems," 2018, pp. 21–39.

[16] S. Lee and C. H. Jun, "Fast incremental learning of logistic model tree using least angle regression," *Expert Syst. Appl.*, vol. 97, pp. 137–145, 2018.

[17] T. D. Pham, D. T. Bui, K. Yoshino, and N. N. Le, "Optimized rule-based logistic model tree algorithm for mapping mangrove species using ALOS PALSAR

imagery and GIS in the tropical region," *Environ. Earth Sci.*, vol. 77, no. 5, p. 159, 2018.

[18] R. Kohavi, "Scaling Up the Accuracy of Naive-Bayes Classi ers : a Decision-Tree Hybrid Accuracy Scale-Up : the Learning," *Kdd*, vol. 96, pp. 202–207, 1996.

[19] N. Landwehr, M. Hall, and E. Frank, "Logistic model trees," *Mach. Learn.*, vol. 59, no. 1–2, pp. 161–205, 2005.

[20] Y. A. Alsariera, A. V. Elijah, and A. O. Balogun, "Phishing Website Detection: Forest by Penalizing Attributes Algorithm and Its Enhanced Variations," *Arab. J. Sci. Eng.*, vol. 45, no. 12, pp. 10459–10470, 2020.

[21] Y. A. Alsariera, "Detecting Generic Network Intrusion Attacks using Tree-based Machine Learning Methods," *Inter. J. of Adv. Comp. & Science and Applications.*, vol. 12, no. 2, pp. 597–603, 2021.

Yazan A. Alsariera (PhD) is Assistant Professor of Software Engineering in the department of computer sciences at Northern Border University. He obtained Bachelor of Computer Science from Mut'ah University, Jordan, in 2010. Master of Science in Computer Science (Minor in Software Engineering) from University of Putra Malaysia (UPM), Malaysia, in 2013. A Ph.D. degree in Science (Software Engineering) from University of Malaysia Pahang (UMP), Malaysia, in 2018. His main research interest includes artificial intelligence, computational intelligence, combinatorial optimization, cybersecurity, secure software development, and high-performance computing.