Iraqi Journal of Information and Communication Technology(IJICT)
Vol. 5, Issue 3, December 2022

www.ijict.edu.iq

ISSN:2222-758X
e-ISSN: 2789-7362

AN EFFECTIVE AND EFFICIENT FEATURES VECTORS FOR RANSOMWARE DETECTION VIA MACHINE LEARNING TECHNIQUE

Nawaf A. Khalil [1], **Ban M. Khammas** [2]
[1,2] College of Information Engineering, Al-Nahrain University, Baghdad, Iraq
nawafaljubory@gmail.com [1] , bankhammas@coie-nahrain.edu.iq [2]
Received:1/1/2022, Accepted:15/3/2022
DOI:10.31987/ijict.5.3.205

***Abstract*- Malicious software known as ransomware encrypts or blocks access to files on a computer to extract a ransom from the user. Ransomware works in a variety of ways, from encrypting all the infected computer's data to merely locking the desktop. Restoring functionality or decrypting files is only possible once you pay the ransomware software with no guarantee to restore data. Unfortunately, ransomware that can identify its environment might elude dynamic analysis's valuable job. To build an effective and powerful detecting approach, this paper introduces a new technique based on static analysis to detect and classify ransomware using five machine learning techniques. First, n-gram features were extracted from samples and CF-NCF values were computed. Then the gain ratio approach was utilized to choose the most essential characteristic. Finally, the vectors of n-grams have been sent to machine-learning techniques to classify ransomware. To evaluate the proposed method, ten evaluation metrics have been used. Using real datasets, the proposed approach shows its ability to reliably identify between goodware and ransomware files successfully with an accuracy of classification equal to 98.33%.**

keywords: **Ransomwares detection, Machine learning, Cybersecurity, Malware analysis, Network security, Feature extraction, Feature selection.**

I. INTRODUCTION

Ransomware is a type of malicious program (malware) which is, after being run on a device such as a computer, mobile etc. blocks the client from ever using the machine or its files while demanding a quantity of money (ransom) for the computer's restoration [1]. This software encrypts the file system using the RSA and AES algorithms. The recommended approach of per-strategy wares is to use the Windows crypto API for RSA encryption and arbitrary key creation [2]. Ransomware typically targets Windows computers, but it also targets other platforms like Android, IOS, and Linux servers. It can operate as an independent piece of malicious software or as part of a bigger group alongside other malicious malware [3]. Ransomware may be transmitted via a variety of techniques, including email links, email attachments, and online sites [4]. There are two forms of ransomware: Locker Ransomware is aimed to limit accessibility to the victim device, preventing them from utilizing the system or any other services. While another type is Crypto Locker Ransomware encrypting sensitive files, data and making it unavailable to users. The victim of Lockers Ransomware loses access to the whole computer or a specific piece of software program on the pc. It chooses documents, photos, and other files with a favorable type. As threats are issued to promote speedy response and payment by victims, this produces a great deal of inconvenience and panic among victims [5]. The increasingly linked Internet world and cloud as many forms of communication make ransomware distribution easier. The consequences of a ransomware threat can be costlier than the ransom. Because of the loss of business, clients, data, and productivity, affected businesses may be harmed for years as a result of the ransomware assault [6]. Many machine learning algorithms and techniques for detection of ransomware have been proposed and developed by

ISSN:2222-758X
e-ISSN: 2789-7362

various researchers [7], but the challenge of efficiently identifying ransomware has yet to be solved. Also, the majority of research on ransomware relies on dynamic analysis via sandboxing. This technology executes malware or untrustworthy software in a safe artificial environment, ensuring that no real harm is done to the system. This strategy has certain drawbacks, including the possibility that ransomware would identify the sandbox environment and refuse to run. Another disadvantage is that because we are operating in a sandbox environment, we may be unaware of the command line options utilized by some ransomware [8]. According to the latest research, the proposed technique tries to improving detection and classification accuracy by using the Class Frequency-Non-Class Frequency (CF-NCF) inductor to describe features vectors in static analysis. To develop a strong method, extract features in direct way from files (Raw Bytes) and the gain ratio approach was used to reduce unneeded features during feature selection. Five supervised machine learning algorithms were utilized to distinguishing benign and ransomware executable files (Random Forest, Support Vector Machines, Logistics Regression, K-Nearest Neighbor and Näive Bayes). Based on our results of the tests, the suggested approach can identify ransomware from non-ransomware data. As a summary, the main contributions of the present work are as follows:

- The suggested solution overcomes the limitations of dynamic analysis by using static analysis that creates the detection model using machine learning methods, allowing for the identification of new ransomware samples through model evolution.
- Numerous metrics have been used to show the effectiveness of our suggested methods.
- The CF-NCF has been used as a variant of TF-IDF. CF-NCF is concerned with the appearance of features in each class, whereas TF-IDF is concerned with the appearance of terms in each document. Experimental data indicate that CF-NCF improves the detection accuracy after using CF-NCF.

The remainder of the paper is presented under the following headings: Part II reviews relevant ransomware detection investigations; part III details the proposed approach; part IV highlights the experimental data; and V concludes the study.

II. Related Work

Numerous detections, analysis, and investigation processes have been created to combat malware; nonetheless, malicious programs employ a variety of propagation and evasion techniques to circumvent protective measures. Machine-learning-based malware classification has been shown to be extremely successful in detecting malware [9]. Ransomware analysis includes measures aimed at analyzing ransomware's characteristics and behavior [10]. There are several analysis methodologies, which may be classified into three broad categories: dynamic, static, and hybrid. Static analysis identifies harmful programs by evaluating the raw data included inside samples without executing them. While the dynamic approach requires all samples to run in order to analyze their behavior and data flows, the static technique does not. Rather than that, the hybrid approach is a synthesis of static and dynamic techniques [11]. First, the related works related to dynamic analysis will be described hereunder then go on to other works related to the static and hybrid analysis. Yuki Takeuchi et al. [12] suggested a method using Support Vector Machines to detect ransomware (SVMs). An SVM learns ransomware Application Program Interface (API) calls as features, allowing it to recognize and categorize previously unknown malware. The technique was tested in a controlled environment utilizing dynamic analysis and the Cuckoo Sandbox. The results of the experiments show

ISSN:2222-758X
e-ISSN: 2789-7362

that the suggested approach enhances the accuracy of ransomware detection. Bae et al. [7] proposed a detection technique to differentiate between benign and ransomware files, as well as between malware and ransomware. They suggested a method for building feature vectors by using dynamic analysis and CF-NCF. There were six different machine learning algorithms utilized (Random Forest, Logistic Regression, Native Bayes, Gradient Descent, K-Nearest Neighbors, Support Vector Machine). According to the results of their tests, their suggested work can identify ransomware amongst benign and malicious samples with a detection rate of 98.65%. Kamalakanta Sethi et al. [13] created an efficient and effective malware identification and classification by using dynamic analysis and machine learning. They employed the Cuckoo sandbox, which runs malware in a controlled environment and gives an analysis report based on system behaviors. The detection results by using the Scikit-Learn package were 99.11 percent. Faizan Ullah et al. [14] provided a methodology for categorizing ransomware and benign files based on unique properties extracted from the malware dataset. The suggested model can identify and scan the network, registry activity, and file system in real-time while in operation, and it can be evaluated using online machine learning algorithms to forecast ransomware. The testing precision has been increased to 99.56 percent. Zubaile Abdullah et al. [15] recommended employing a dynamic analysis approach to identify Android ransomware. The authors created uniPDroid, a tool for extracting features from Android apps. The random forest algorithm achieved the best detection accuracy up to 98.31 percent and the lowest false positive rate of 0.016. Jinsoo Hwang et al. [16] created a random forest model and Markov, as well as a two-mixed stages ransomware detection algorithm. To begin, they create Markov models for both malicious and benign applications to concentrate just on the sequential nature of Windows APIs. They next looked at various statistical machine learning algorithms that used all of the characteristics, in addition to the Windows API call. Finally, they presented a mixed detection approach with two stages. Using our two-stage mixed detection technique, they achieved an accuracy of 97.3 percent. While for static malware analysis, Hamed Haddad et al. [17] proposed a machine learning model based on the Radial Base Function (RBF) in the SVM approach to identify OS X malware. They created a new grading system for identifying OS X goodware from malware based on the frequency of library calls. The main classification task in the proposed technique is built with SVM. According to the data, the model achieved a 97.1 percent detection accuracy and a 3.9 percent false alarm rate. Hanqi Zhang et al. [18] established a technique for converting ransomware opcode sequences into N-gram sequences. The Term Frequency-Inverse Document Frequency (TF-IDF) is then calculated for each N-gram, resulting in a better family identification. Then, 5 machine-learning algorithms were utilized to categorize ransomware, that was achieving accuracy up to 91.43 percent based on experimental results. Akram M. Radwan [19] demonstrated machine learning to determine if a portable executable file is malicious or goodware. The integrated feature set was extracted using the static analysis technique. Seven supervised learning algorithms are utilized to classify malware, and testing findings show that the integrated feature set beats the raw feature set on all measures. Subash Poyudal et al. [8] created a reverse engineering framework for ransomware detection that integrates feature generating engines and machine learning (ML). This approach is being used to analyze in multi-levels on malicious source codes to better assess and clarify their aim. They used the object-code dump tool (Linux) and the portable executable processor to convert binaries to assembly level instructions and dynamic link libraries (DLLs). The experimental findings showed that the detection performance for ransomware samples ranged between 76

ISSN:2222-758X
e-ISSN: 2789-7362

and 97 percent depending on the machine learning technique used. To perform a hybrid analysis, Ashik, Mathew, et al. [20] combined static and dynamic analysis in a hybrid system to detect malware performed in multi datasets. Experiments were conducted utilizing machine learning and deep learning methodologies. To increase accuracy, they implemented their solution utilizing multiple deep learning approaches and presented a fine-tuned deep neural network that achieved an F1 score of 99.1 percent. Iman Almonani et al. [11] developed a hybrid approach for detecting Android ransomware that relies on evolutionary machine learning. The binary particle swarm optimization approach is used to tune the classification algorithm's hyperparameters and to perform feature selection. Classification is performed using the SVM technique in conjunction with the synthetic minority oversampling technique (SMOTE). Samuel Egunjobi et al. [5] proposed a hybrid technique to improve the accuracy of ransomware detection. They utilize a test set to train machine learning techniques and a confusion matrix to measure accuracy. This research yielded a 96 percent accuracy with the test set result, 99.5 percent accuracy with SVM, 99.5 percent accuracy with random forest, and 96 percent accuracy with the test set result. Dynamic analysis techniques have drawbacks in that the ransomware must be run in a secure and supervised environment, or else the system will become contaminated. In detecting ransomware samples, static analysis is quick, safe, and accurate. More significantly, in dynamic analysis, the analyzing environment varies from the actual one, the malicious software may act differently, resulting in distinct execution logs. As a result, static analysis was used in this work.

III. METHODOLOGY

The present study illustrates a new effective strategy for rapidly developing and overcoming the limitations of dynamic analysis for ransomware executable file identification and classification, by utilizing numerous machine learning models based on static analysis. Since every ransomware family has its own set of traits. The proposed approach makes advantage of byte-level static analysis, which extracts features in direct way from the raw bytes of each acquired executable file. Extraction at the byte level is considered to be a quicker and more straightforward method [21]. The process design for the suggested methodology is depicted in Fig. 1. Which is separated into six major sections and begins with the gathering of executable files (ransomware and benign). Then, divide the dataset in half for training and testing purposes. Following that is preprocessing, which consists of three implicit phases (feature extraction, N-gram vector, and CF-NCF). After that, feature selection, model construction, and classification are performed. Each stage is described in full below. In dataset collection, the newest version of ransomware executable applications has been collected to compile the dataset. The dataset was collected from two different website which are VirusTotal [22] and ShieldFS [23] websites. While the benign samples, collected from Windows platform. Each file must be Fig. 1. The suggested methodology framework. under or equal 1MB in size; files over 1MB will be ignored. To prevent unbalancing the files obtained, split them into two equal groups for training and testing datasets. Both the training and testing datasets include a same amount of malicious and benign files. Then comes the preprocessing stage, which is a three-phase technique. The feature extraction module extracts feature from any executable file, even those that contain malware. It extracts features from files using the n-gram feature extraction technique. The N-gram is a sequence of n bytes with features corresponding to the different combinations of these n bytes; more precisely, each feature specifies the frequency with which a certain combination of n bytes occurs in the binary [24], [25]. When n equals 4, n-gram has the highest accuracy, and as n grows, the accuracy drops. As a result, when eight, nine,

ISSN:2222-758X
e-ISSN: 2789-7362

or ten grams are used, the technique gets more complicated, and the accuracy diminishes [16]- [19]. For high detection accuracy, the feature extraction technique is carried out in a virtual machine (VM) and utilizes 32-bit sliding windows (4-gram) features. Then compute the CF-NCF values. CF-NCF is a model-based categorization indicator. The CF-NCF have been introduced to emphasize the peculiarities of each class. Rather than relying on word occurrences within a text, CF-NCF focuses on the presence of features inside a certain class. Due to the fact that CF-NCF can compute weights for objects inside a class, it has the potential to increase classification accuracy. The following equation was used to calculate CF-NCF [7].

$$\begin{aligned} CF(s,C) &= f(s,C) \\ NCF(s,N) &= log(\frac{1}{0.001 + f(s,N)}) \\ CF - NCF &= CF * NCF \end{aligned} \tag{1}$$

Wheres is a n-gram, $f(s,C)$ is the number of times the n-gram s occurs in one class's n-gram sequence C, N is the other classes' n-gram sequences, and 0.001 keeps the denominator from approaching zero. The $CF-NCF$ values for the ransomware class's n-gram sequence are C for the ransomware class and N for the benign class's n-gram sequences. To increase detection accuracy, $CF-NCF$ was used in this investigation for two classes: benign and ransomware classes. After that, the feature selection step is to identify a subset of the original set of characteristics that is sufficiently representational of the data and contains highly relevant attributions for prediction [26]. The dataset contains thousands of n-gram features. Because a large number of these qualities have little influence on the process of categorization. As a result, the technique to feature selection is vital for picking the fewest possible relevant features, as a smaller feature space is likely to be more essential to the classifier than the original dataset [19]. The Gain Ratio has been picked as one of the feature selection strategies (GR). Model generation is the next phase. To create the final input vector, a set of n-grams and CF-NCF values is used. Finally, to identify unknown binary samples from the testing dataset as ransomware or benign files, the classification procedure developed using the training dataset's generated vectors with weights is used to classify the files using five machine learning approaches. These techniques began by developing machine learning models with the training dataset as their input. The models were then validated against the test dataset to determine their accuracy of categorization. The ransomware classification technique can handle big data set; this is referred to as a supervised learning algorithm, which develops a formal model capable of classifying and sorting data based on correlating features. The supervised learning technique supports a two-phase classification evaluation. The first part is to train a classifier using labeled training data, and the second phase is testing, during which the classifier determines if an instant is normal or suspicious [27]. Both ransomware (this demonstrates that the infected samples carried ransomware with payloads belonging to a specific family) and benign software are scanned at "virustotal.com" . VirusTotal is a freeware program that is used to determine whether a file contains ransomware or not.

ISSN:2222-758X
e-ISSN: 2789-7362

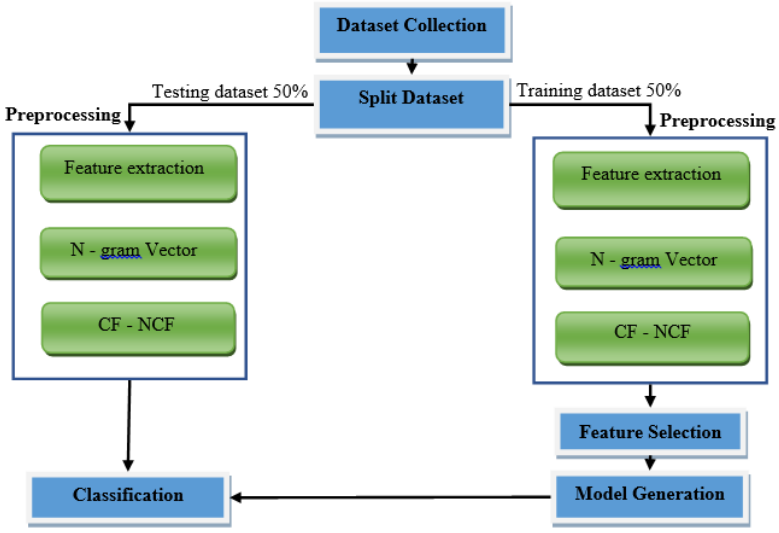

Figure 1: The suggested methodology framework

IV. Experimental Results

The suggested approach classifies benign files from ransomware ones. The following experiments were conducted to demonstrate the CF-NCF inductor's performance using commonly used machine learning performance evaluation criteria, including classification Accuracy (ACC), True Negative Rate (TNR), True Positive Rate (TPR), False Negative Rate (FNR), False Positive Rate (FPR), Precision, Recall, F-measure, and Mathews Correlation Coefficient (MCC). This study utilized a PC with a Core i7, 2.4 GHz CPU, and 16GB RAM; as two operating systems to safeguard the host against ransomware compromise: Windows 10 and Ubuntu 18.04, 64-bit, 1200 samples, 600 ransomware files, and 600 benign files have been downloaded from the dataset collection. The file selected is less than or equal to 1MB in size; otherwise, the file is canceled. To build the datasets, the 4-gram feature have been extracted directly from 600 ransomware files and 600 benign executables. Then, WEKA program is used to select the most important feature (1000 features) for high detection accuracy by using Gain Ratio techniques. Finally, we used the MATLAB environment to develop detection models for the created vector using a collection of extracted n-gram sequences and CF-NCF values, as well as classification models due to the simplicity and scalability of the environment. The following five supervised machine learning algorithms were used for classification purposes in this work [28]-[30]:

A. Support Vector Machines (SVM):

For pattern recognition and data analysis, SVM is a widely used supervised learning model. SVM creates a non-probabilistic binary linear classification model that is used to determine the category to which fresh data belongs.

B. K-Nearest Neighbors (KNN):

This algorithm prioritizes keeping comparable objects close together. This model is applied to a data set's class labels and feature vectors. KNN stores all instances and assists in classifying new instances using a similarity metric.

ISSN:2222-758X
e-ISSN: 2789-7362

C. Random Forest (RF):

Random Forest is an ensemble learning technique for classification and regression analysis that generates outputs by mixing several decision trees constructed throughout training.

D. Logistic Regression (LR):

Logistic regression indicates where the boundary between classes exists and indicates how class probabilities depend on distance from the boundary.

E. Näive Bayes (NB):

The Näive Bayes classification method is constructed using the Bayesian theorem. Numerous classes can be assigned to a single element. Random Forest is an ensemble learning technique for classification and regression analysis that generates outputs by mixing several decision trees constructed throughout training. When supervised machine learning is used, it is common to partition randomly the dataset into two groups using percentages determined by the researcher's desire. In this stage of workflow, the random division has been used in a 50:50 ratio, to avoid unbalancing problem. Half (50%) of the dataset will be utilized to train the machine learning algorithm that will build the predictive model. The remaining 50% of the dataset is then used to produce predictions using this predictive model. With 300 ransomware files and 300 benign files, both the training and testing datasets have the same number of ransomware and benign files. When the CF-NCF is applied to both the testing and training datasets, the experimental findings will show two distinct groups of results for standard machine learning performance evaluation metrics before and after the CF-NCF. The proposed technique has been evaluated with multiple machine learning performance evaluation metrics such as TPR, FPR, TNR, FNR, ACC, Precision, Recall, F-measure, and MCC. These metrics were calculated using the following equations [31]- [33]:

$$ACC = \frac{TP + TN}{TP + FP + TN + FN} \tag{2}$$

$$TPR = \frac{TP}{TP + FN} \tag{3}$$

$$FPR = \frac{FP}{FP + TN} \tag{4}$$

$$TNR = \frac{TN}{TN + FP} \tag{5}$$

$$FNR = \frac{FN}{FN + TP} \tag{6}$$

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

$$F - measure = 2 * \frac{Precision \cdot Recall}{Precision + Recall} \tag{9}$$

$$MCC = 2 * \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{10}$$

ISSN:2222-758X
e-ISSN: 2789-7362

Where: True Positive refers to the number of ransomware instances that are accurately identified as ransomware (TP). The total amount of benign files is categorized accurately as benign (True Negative, or TN). False Positive (FP): a small number of benign files were mistakenly detected as ransomware. False Negative (FN): the number of ransomwares that are incorrectly categorized as benign. Fig. 2 illustrates the comparisons on experimental results of used performance evaluation metrics including ACC, Precision, Recall, F-measure, and MCC in two cases before using CF-NCF inductor in the dataset and after using it. As shown in Fig. 2. A the experimental results of classification accuracy on the testing dataset. The suggested method can distinguish between benign and ransomware programs. With the application of CF-NCF to both the testing and training datasets, the classification accuracy for three models SVM, RF, and NB increased from 95.83 percent, 97.12 percent, and 88.83 percent before using CF-NCF to 97.16 percent, 98.33 percent, and 89.83 percent respectively after using CF-NCF. While KNN model was 88% before the CF-NCF and little affected after CF-NCF becoming 87.83%. The LR didn't work probably in the two cases, it was 60.33% before CF-NCF and then become 45.16% after CF-NCF. Other measures have improved in value as well, following the adoption of CF-NCF on the dataset for three of the five models employed in this study. Fig. 2. B illustrates the results of precision that are enhanced by applying CF-NCF in four models including SVM, KNN RF, and NB. Fig. 2. C show experimental results of Recall metrics that slightly improved after the CF-NCF was utilized for SVM, RF, and NB. Fig. 2. D compares the F-measure testing results in two scenarios before and after adopting CF-NCF equations on the dataset. While Fig. 2. E represents MCC results. Fig. 3, demonstrates the TPR and TNR experimental results before utilizing the CF-NCF method, whilst Fig. 4 represents the significant improvement in results of the same metrics after using CF-NCF inductors. The present work's findings are compared to those of Hanqi Zhang et al. [18], and Subash Poyudal et al. [8], who utilized static analysis, as shown in Table I. Subash el al. used eight machine learning methods and Cosine similarity to achieve a classification accuracy up to 97%. While Hanqi et al. used five machine learning models and TF-IDF method to achieve accuracy up to 91.43%. Static analysis is also used in the current procedure, that used five models and CF-NCF method to achieve a classification accuracy up to 98.33

TABLE I
Show the comparisons between proposed method, Subash Poyudal [8] method, and Hanqi Zhang [18] method

System	*Method of Analysis*	*Samples (Ransomware/ Benign*	*Feature Type*	*Method*	*Algorithm*	*Accuracy %*
Proposed method	Static Analysis	600/600	N-gram	CF-NCF	RF, KNN, SVM, LR, and NB	98.33%
Subash et al. [8]	Static Analysis	178/178	Opcodes	Cosine similarity	"LR, RF, Bayesian Network, Sequential Minimal Optimization (SMO) with Linear Kernel, SMO with Poly kernel, J48, AdaboostM1 with J48, and AdaboostM1 with RF".	97 %
Hanqi et al [18]	Static Analysis	1787/ 100	N-gram	TF-IDF	Decision Tree, RF, KNN, NB And Gradient Boosting Decision.	91.43%

ISSN:2222-758X
e-ISSN: 2789-7362

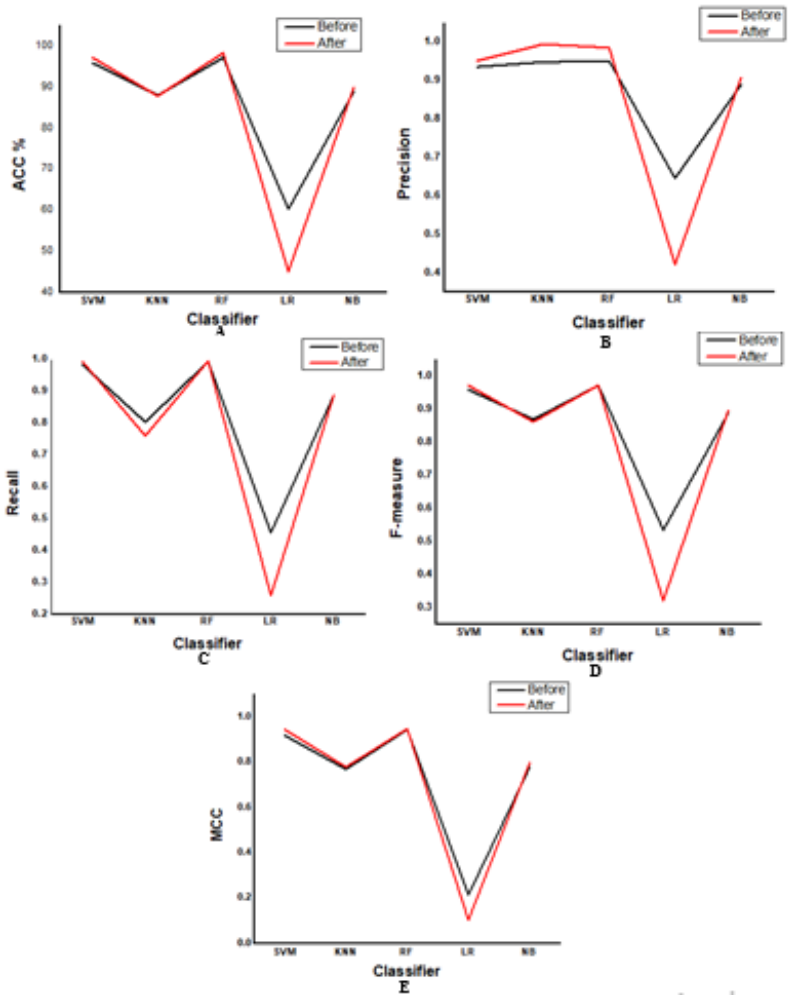

Figure 2: Show before and after CF-NCF, compare the outcomes of ACC, Precision, Recall, F-measure, and MCC

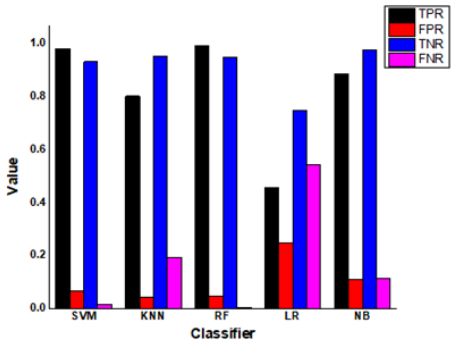

Figure 3: Before utilizing CF-NCF, display the TPR, TNR, FPR, and FNR values

ISSN:2222-758X
e-ISSN: 2789-7362

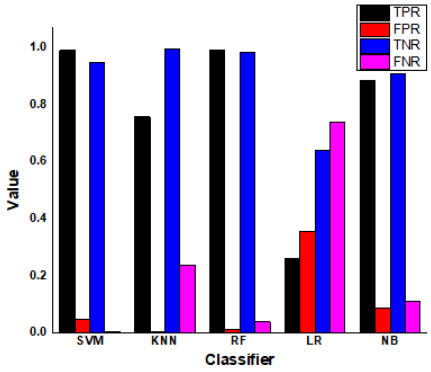

Figure 4: After using CF-NCF, display the TPR, FPR, TNR, and FNR values

V. CONCLUSION

With the rise in ransomware-based cyberattacks, ransomware detection techniques are required. To protect against ransomware the current study presented a technique that utilized n-gram feature for classifying ransomware with the use of CF-NCF method and five machine learning models based on static analysis which doesn't need to run ransomware files. The suggested ransomware detection technique used the Gain ratio technique to select the most relevant 1000 features for enhancing the classification accuracy. According to the experimental results, the RF beat all other used models in all evaluation metrics which can achieve high detection accuracy 98.33%. For future study, a novel technique based on static analysis will be developed to identify ransomware from other types of malwares.

REFERENCES

[1] D. Nieuwenhuizen, "A Behavioural-Based Approach to Ransomware Detection" , Whitepaper. MWR Labs Whitepaper, 2017, [Online], Available: https://labs.mwrinfosecurity.com/assets/resourceFiles/mwri-behavioural-ransomware-detection-2017-04-5.pdf.
[2] A. Tandon and A. Nayyar, "A Comprehensive Survey on Ransomware Attack: A Growing Havoc Cyberthreat" , Vol. 839, Springer Singapore, 2019.
[3] S. Usharani, P. M. Bala, and M. M. J. Mary, "Dynamic Analysis on Crypto-Ransomware by Using Machine Learning: Gandcrab Ransomware" , J. Phys. Conf. Ser. , Vol. 1717, No. 1, 2021, doi: 10.1088/1742-6596/1717/1/012024.
[4] A. H. Mohammad, "Analysis of Ransomware on Windows platfor" , Vol. 20, No. 6, pp. 21-27, 2020.
[5] S. Egunjobi, S. Parkinson, and A. Crampton, "Classifying Ransomware Using Machine Learning Algorithms" , Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics) , Vol. 11872 LNCS, pp. 45-52, 2019, doi: 10.1007/978-3-030-33617-2-5.
[6] F. Salahdine and N. Kaabouch, "Social Engineering Attacks: A Survey" , Futur. Internet, Vol. 11, No. 4, 2019, doi: 10.3390/FI11040089.
[7] S. Il Bae, G. Bin Lee, and E. G. Im, "Ransomware Detection Using Machine Learning Algorithms" , Concurr. Comput. , No. May, pp. 1-11, 2019, doi: 10.1002/cpe.5422.
[8] S. Poudyal, K. P. Subedi, and D. Dasgupta, "A Framework for Analyzing Ransomware using Machine Learning" , Proc. 2018 IEEE Symp. Ser. Comput. Intell. SSCI 2018, pp. 1692-1699, 2019, doi: 10.1109/SSCI.2018.8628743.
[9] C. Keong Ng, S. Rajasegarar, L. Pan, F. Jiang, and L. Y. Zhang, "VoterChoice: A Ransomware Detection Honeypot with Multiple Voting Framework" , Concurr. Comput. , Vol. 32, No. 14, pp. 1-29, 2020, doi: 10.1002/cpe.5726.
[10] H. Oz, A. Aris, A. Levi, and A. S. Uluagac, "A Survey on Ransomware: Evolution, Taxonomy, and Defense Solutions" , ACM Comput. Surv. , Vol. 1, No. 1, 2021, [Online] , Available: http://arxiv.org/abs/2102.06249.
[11] I. Almomani, R. Qaddoura, and M. Habib, "Android Ransomware Detection Based on a Hybrid Evolutionary Approach in The Context of Highly Imbalanced Data" , No. April, 2021, doi: 10.1109/ACCESS.2021.3071450.
[12] G. Cusack, O. Michel, and E. Keller, "Machine Learning-Based Detection of Ransomware Using SDN" , SDN-NFVSec 2018 - Proc. 2018 ACM Int. Work. Secur. Softw. Defin. Networks Netw. Funct. Virtualization, Co-located with CODASPY 2018, Vol. 2018, Janua. , pp. 1-6, 2018, doi: 10.1145/3180465.3180467.
[13] K. Sethi, R. Kumar, L. Sethi, P. Bera, and P. K. Patra, "A Novel Machine Learning Based Malware Detection and Classification Framework" , 2019 Int. Conf. Cyber Secur. Prot. Digit. Serv. Cyber Secur. 2019, pp. 1-4, 2019, doi: 10.1109/CyberSecPODS.2019.8885196.
[14] F. Ullah et al. , "Modified Decision Tree Technique for Ransomware Detection at Runtime through API Calls" , Sci. Program, Vol. 2020, 2020, doi: 10.1155/2020/8845833.
[15] Z. Abdullah, F. W. Muhadi, M. M. Saudi, I. R. A. Hamid, and C. F. M. Foozy, "Android Ransomware Detection Based on Dynamic Obtained Features" , Adv. Intell. Syst. Comput. , Vol. 978 AISC, No. March 2016, pp. 121-129, 2020, doi: 10.1007/978-3-030-36056-6-12.
[16] J. Hwang, J. Kim, S. Lee, and K. Kim, "Two-Stage Ransomware Detection Using Dynamic Analysis and Machine Learning Techniques" , Wirel. Pers. Commun. , Vol. 112, No. 4, pp. 2597-2609, 2020, doi: 10.1007/s11277-020-07166-9.
[17] H. H. Pajouh, A. Dehghantanha, R. Khayami, and K. K. R. Choo, "Intelligent OS X Malware Threat Detection with Code Inspection" , J. Comput. Virol. Hacking Tech. , Vol. 14, No. 3, pp. 213-223, 2018, doi: 10.1007/s11416-017-0307-5.
[18] H. Zhang, X. Xiao, F. Mercaldo, S. Ni, F. Martinelli, and A. K. Sangaiah, "Classification of Ransomware Families with Machine Learning Based on N-Gram of Opcodes" , Futur. Gener. Comput. Syst. , Vol. 90, pp. 211-221, 2019, doi: 10.1016/j.future.2018.07.052.
[19] A. M. Radwan, "Machine Learning Techniques to Detect Maliciousness of Portable Executable Files" , Proc. - 2019 Int. Conf. Promis. Electron. Technol. ICPET 2019, pp. 86-90, 2019, doi: 10.1109/ICPET.2019.00023.
[20] M. Ashik et al. , "Detection of Malicious Software by Analyzing Distinct Artifacts Using Machine Learning and Deep Learning Algorithms" , pp. 1-28, 2021.
[21] B. M. Khammas, S. Hasan, N. Nateq, J. S. Bassi, I. Ismail, and M. N. Marsono, "First Line Defense Against Spreading New Malware in the Network" , 2018 10th Comput. Sci. Electron. Eng. Conf. CEEC 2018 - Proc. , pp. 113-118, 2019, doi: 10.1109/CEEC.2018.8674214.
[22] "VirusTotal Website" , https://www.virustotal.com.
[23] "ShieldFS Website" , http://shieldfs.necst.it.
[24] D. Ucci, L. Aniello, and R. Baldoni, "Survey of Machine Learning Techniques for Malware Analysis" , Comput. Secur. , Vol. 81, pp. 123-147, 2019, doi: 10.1016/j.cose.2018.11.001.
[25] C. Liangboonprakong and O. Sornil, "Classification of Malware Families Based on N-Grams Sequential Pattern Features" , Proc. 2013 IEEE 8th Conf. Ind. Electron. Appl. ICIEA 2013, pp. 777-782, 2013, doi: 10.1109/ICIEA.2013.6566472.
[26] Y. Zhou, G. Cheng, S. Jiang, and M. Dai, "Building An Efficient Intrusion Detection System Based on Feature Selection and Ensemble Classifier" .
[27] A. Azman, W. Yassin, O. Mohd, M. F. Abdollah, and R. S. Abdullah, "Ransomware Detection Using Classification Method Against Registry Data" , J. Theor. Appl. Inf. Technol. , Vol. 97, No. 22, pp. 3366-3376, 2019.
[28] T. Singh, F. Di Troia, V. A. Corrado, T. H. Austin, and M. Stamp, "Support Vector Machines and Malware Detection" , J. Comput. Virol. Hacking Tech. , Vol. 12, No. 4, pp. 203-212, 2016, doi: 10.1007/s11416-015-0252-0.
[29] A. G. P., G. R., K. S., and A. Gladston, "A Machine Learning Framework for Domain Generating Algorithm Based Malware Detection" , Secur. Priv. , Vol. 3, No. 6, pp. 1-16, 2020, doi: 10.1002/spy2.127.
[30] K. Shah, H. Patel, D. Sanghvi, and M. Shah, "A Comparative Analysis of Logistic Regression, Random Forest and KNN Models for the Text Classification" , Augment. Hum. Res. , Vol. 5, No. 1, 2020, doi: 10.1007/s41133-020-00032-0.
[31] A. M. Hamad alhussainy and A. D. Jasim, "ECG Signal Classification Based on Deep Learning by Using Convolutional Neural Network (CNN)" , Iraqi J. Inf. Commun. Technol. , Vol. 3, No. 3, pp. 12-23, 2020, doi: 10.31987/ijict.3.3.106.
[32] Q. Wu, X. Zhu, and B. Liu, "A Survey of Android Malware Static Detection Technology Based on Machine Learning" , Vol. 2021, 2021.
[33] H. M and S. M. N, "A Review on Evaluation Metrics for Data Classification Evaluations" , Int. J. Data Min. Knowl. Manag. Process, Vol. 5, No. 2, pp. 01-11, 2015, doi: 10.5121/ijdkp.2015.5201.

