Ransomware Early Detection Techniques

Asma A. Alhashmi
Department of Computer Science, College of Science, Northern Border University, Saudi Arabia
asma.alhashmi@nbu.edu.sa

Abdulbasit A. Darem
Department of Computer Science, College of Science, Northern Border University, Saudi Arabia
basit.darem@nbu.edu.sa (corresponding author)

Ahmed B. Alshammari
Department of Computer Science, College of Science, Northern Border University, Saudi Arabia
ahmed.alshammari@nbu.edu.sa

Laith A. Darem
Department of Electrical Engineering, Northern Border University, Saudi Arabia
laith.darem@nbu.edu.sa

Huda K. Sheatah
Department of Computer Science, College of Science, Northern Border University, Saudi Arabia
huda.sheatah@nbu.edu.sa

Rachid Effghi
Department of Big Data Analytics and Management, Bahcesehir University, Turkiye
rachideffghi@bahcesehir.edu.tr

Received: 16 January 2024 | Revised: 28 March 2024 | Accepted: 30 March 2024

 DOI: https://doi.org/10.48084/etasr.6915

ABSTRACT

Ransomware has become a significant threat to individuals and organizations worldwide, causing substantial financial losses and disruptions. Early detection of ransomware is crucial to mitigate its impact. The significance of early detection lies in the capture of ransomware in the act of encrypting sample files, thus thwarting its progression. A timely response to ransomware is crucial to prevent the encryption of additional files, a scenario not adequately addressed by current antivirus programs. This study evaluates the performance of six machine-learning algorithms for ransomware detection, comparing the accuracy, precision, recall, and F1-score of Logistic Regression, Decision Tree, Naive Bayes, Random Forest, AdaBoost, and XGBoost. Additionally, their computational performance is evaluated, including build time, training time, classification speed, computational time, and Kappa statistic. This analysis provides an insight into the practical feasibility of the algorithms for real-world deployment. The findings suggest that Random Forst, Decision Tree, and XGBoost are promising algorithms for ransomware detection due to their high accuracy of 99.37%, 99.42%, and 99.48%, respectively. These algorithms are also relatively efficient in terms of classification speed, which makes them suitable for real-time detection scenarios, as they can effectively identify ransomware samples even in the presence of noise and data variations.

Keywords-ransomware; early detection; machine learning; computational performance; cybersecurity

I. INTRODUCTION

Ransomware has rapidly become one of the most financially destructive cybersecurity threats of the decade. These attacks encrypt files and systems to extort massive amounts from victims by demanding ransoms for data recovery [1]. Recent trends exhibit an exponentially growing threat as ransomware-related damages increased by an astonishing 93% from 2018 to 2019 [2]. The major incidents in 2021 brought key healthcare networks, food supply infrastructure, and even law enforcement departments to their knees. Some estimates now shockingly indicate that a business falls prey every 14 seconds as new sophisticated malware families proliferate [3]. The average recovery cost from an

attack has reached a crippling $1.85 million according to the 2019 statistics [4]. However, defensive solutions struggle to contain this growth trajectory as threat actors rapidly innovate exploits against incremental security advances.

However, extensive research on fighting ransomware propagation contrasts with the relative study of early detection mechanisms [5]. Legacy signature-dependent models are ineffective against novel attacks. But critically, studies reveal that more than 68% of infections exhibit substantial lateral movement within breached networks before diagnosis [6]. Such activity spikes severely exacerbate financial bleeding and data loss. A Deloitte analysis identified that ransomware incidents averted in just 30 minutes after infiltration could have saved almost 95% of costs [7]. Therefore, the prompt detection of behavioral anomalies indicating ransomware is essential to rapid response and damage mitigation. Recent innovations demonstrate the promising potential to identify ransomware using system API monitoring, registry tracking, file activity logs, and network traffic metadata [8]. However, current tools remain limited to simulated environments, while practical, real-world diagnostic capabilities are unverified [9]. The challenge of collecting relevant data on the scale required for training, securing deployment infrastructure, obtaining ground truths, and maintaining high precision persists [10-12]. This study:

- Evaluates six Machine Learning (ML) algorithms, namely Logistic Regression (LR), Decision Tree (DT), Naive Bayes (NB), Random Forest (RF), Adaptive Boosting (AdaBoost), and Extreme Gradient Boosting (XGBoost) for ransomware detection, revealing strengths and weaknesses in accuracy, precision, recall, ROC AUC, MCC, and F1-score. Comparative analysis aims to discern the most effective algorithms for early detection in diverse ransomware scenarios.
- Extends analysis beyond traditional metrics to include computational performance, such as model building time, training time, classification speed, and computational time, improving real-world implementation considerations.
- Identifies Random Forest and XGBoost as superior choices for ransomware detection, offering exceptional accuracy and practical insights.

The results can significantly augment organizational resilience against ransomware while alleviating financial and business operation impacts that cripple victims after attacks. This study makes seminal contributions to advance the empirical understanding of practical and deployable ransomware early warning solutions to meet a pressing need.

II. LITERATURE REVIEW

Several studies have identified gaps in the existing defense mechanisms, emphasizing the need for proactive and robust ransomware detection methods [13-15]. These gaps underscore the urgency of developing more effective early detection methods and strategies, especially in the face of evolving ransomware tactics and the increasing the sophistication of attacks. Ransomware attacks have become a significant threat to individuals, businesses, and even national security, encrypting victims' data and demanding ransom for decryption keys. Ransomware employs advanced encryption techniques, making data recovery without the decryption key extremely challenging [14]. The evolution of ransomware attacks requires robust early detection and prevention strategies to mitigate their impact. Early detection of ransomware is crucial in preventing data loss and financial repercussions. It involves identifying ransomware attempts before files are encrypted or systems are locked, allowing users or network administrators to take preemptive actions.

ML algorithms have become pivotal in the fight against ransomware, offering a dynamic and adaptive approach to detect and mitigate threats before they inflict damage [16]. These algorithms analyze huge datasets to learn and identify patterns indicative of ransomware activity, thus enabling early detection and response. In [17], the use of various ML algorithms for ransomware classification and detection was explored, highlighting their effectiveness in distinguishing between ransomware and benign software. A comprehensive survey in [18] discussed ML algorithms trained specifically to detect ransomware encryption activity, underlining the promising results in identifying ransomware behaviors. In [19], the focus was on automated ransomware behavior analysis, detailing how pattern extraction from ransomware's operational behavior can facilitate early detection, suggesting that understanding ransomware's execution patterns is key to developing effective countermeasures.

Another dimension of ransomware detection involves analyzing network traffic for signs of ransomware communication. In [20], ML techniques were used to detect ransomware through Windows ransomware network traffic, indicating the potential of network behavior analysis to identify ransomware activities. In [21], a hardware-assisted runtime detection technique called RanStop was introduced, which utilized hardware-level micro-architectural information to detect crypto-ransomware, showcasing the advantages of hardware-assisted approaches in bypassing software-level obfuscation tactics adopted by ransomware authors. In [22], it was displayed the extent to which process memory analysis can be an effective strategy in ransomware detection. This approach relied on identifying abnormal memory patterns associated with ransomware encryption processes. In [23], a feature selection technique was proposed based on redundancy coefficient gradual up-weighting and mutual information to improve the early detection of crypto-ransomware. This method emphasized the importance of selecting relevant features that contribute to the accuracy of the detection models.

Although individual ML algorithms have manifested effectiveness in detecting ransomware, a comparative analysis reveals the nuanced advantages of different approaches. For instance, Decision Tree-based models, such as Random Forest, offer the benefit of interpretability, which is crucial for understanding the decision-making process behind classifications. Despite their promise, ML-based detection systems are not without challenges. The dynamic nature of ransomware means that detection models must be continuously updated to recognize new and evolving threats. Additionally, attackers are increasingly employing techniques such as adversarial ML to evade detection, creating a cat-and-mouse

game between defenders and attackers. While various detection techniques offer promising results, their effectiveness can vary based on the ransomware strain, the system architecture, and the operational environment. Despite advances in detection techniques, ransomware continues to evolve, presenting new challenges that require continuous research and development.

III. RESEARCH METHODOLOGY

The method followed to evaluate the performance and computational metrics of the different ML algorithms used for ransomware detection involves several steps, including data collection, preprocessing, feature engineering, model selection, and experimental implementation.

A. Algorithms Used

Logistic Regression (LR) is a statistical method for analyzing a dataset with one or more independent variables that calculate the probability of a given outcome. It is often employed in ML models for classification tasks, such as fraud detection. LR utilizes a logistic function to model the relationship between the independent variables and the target variable [24]. The Decision Tree (DT) algorithm involves recursively partitioning the dataset based on the features to create a tree-like structure. The primary formula implemented in DT is the impurity measure, commonly the Gini index or information gain (entropy). DT aims to create a tree that effectively classifies instances based on feature values, making it a powerful tool for both classification and regression tasks [25]. Naive Bayes (NB) is a probabilistic ML algorithm based on the Bayes theorem. It is particularly effective for classification tasks and is known for its simplicity and efficiency. The naive assumption is that the features are conditionally independent of the class label. Given an input vector x, the NB algorithm predicts the class label y by calculating the probability of each class given the input features. The Random Forest (RF) algorithm is based on decision trees and random sampling used for classification and regression tasks. It builds multiple decision trees during training and selects the best prediction from each tree to improve overall accuracy [26]. Adaptive Boosting (AdaBoost) is an ensemble learning method that combines multiple weak classifiers to create a strong one. The algorithm assigns weights to instances and classifiers and adjusts them on the basis of classification errors. For a given input x, the AdaBoost algorithm predicts the class label y by aggregating the predictions from multiple weak learners. The final classification is determined through a weighted sum of weak classifier predictions. AdaBoost iteratively improves the performance of weak classifiers by assigning higher weights to misclassified instances. The final prediction is a weighted combination of weak classifier predictions. Extreme Gradient Boosting (XGBoost) is a powerful gradient-boosting algorithm that is particularly effective for regression and classification problems. It combines the concepts of gradient boosting and regularization to achieve high performance. The algorithm iteratively builds decision trees and optimizes the objective function by updating the leaf scores. XGBoost also introduces concepts such as feature importance and early stopping to enhance its performance.

B. Dataset

This study used the UGRansome dataset [27], which is a key instrument for the identification of ransomware threats [28-29]. This dataset is distinctive in its inclusion of ransomware types not previously documented in other datasets [29]. It encompasses a spectrum of infamous ransomware families, including Locky and CryptoLocker, along with the notorious WannaCry, and extends to cover sophisticated persistent cyber threats. The dataset comprises 207,533 samples. Each sample was characterized by 14 distinct features, providing a rich representation of the file's properties, as illustrated in Table I. This dataset was carefully selected due to its substantial sample size, enabling effective training, and testing of ML models. Furthermore, the well-defined features facilitated the extraction of meaningful insights from the data. This dataset offered several advantages in contrast to other datasets. First, it surpasses other ransomware datasets in terms of sample size, ensuring robust model training and evaluation. Second, unlike generic malware datasets, this dataset specifically focuses on ransomware, aligning with the specific objectives of this study. Third, compared to datasets intended for signature-based detection or dynamic analysis, this dataset was more suitable for the ML-based approach adopted in this study.

TABLE I. DATASET FEATURES DESCRIPTION

Feature	Description
Time	Quantitative column with integers indicating the timestamp of network attacks.
Protocol	Qualitative/categorical column representing the network protocol used (e.g., TCP, UDP)
Flag	Qualitative/categorical column indicating network connection status (e.g., SYN, ACK).
Family	Qualitative/categorical column describing network intrusion category.
Clusters	Quantitative column with integers denoting event clusters or groups.
SeedAddress	Qualitative/categorical column representing formatted ransomware attack links.
ExpAddres	Qualitative/categorical column indicating original ransomware attack links.
BTC	Numeric column with values related to Bitcoin transactions in attacks.
USD	Numeric column indicating financial damages in USD caused by attacks.
Netflow Bytes	Quantitative column with integers showing bytes transferred in network flow.
IPaddress	Qualitative column with IP addresses associated with network events.
Threats	Qualitative column representing the nature of threats or intrusions.
Port	Quantitative column indicating network port number in events.
Prediction	This is the target variable. It is a qualitative/categorical column indicating predictive model outcomes: Anomaly (A), Signature (S), and Synthetic Signature (SS).

C. Data Preprocessing

An examination suggested that the dataset contained no missing values or duplicate entries. This eliminated the need for extensive preprocessing, allowing the study to focus on extracting the most relevant information from the data. Two features, namely "Name" and "md5," were identified as traits

containing limited information for the classification task. Therefore, these features were removed from the dataset. The remaining features were all numerical, eliminating the need for any encoding or transformation.

D. Dataset Split

The train_test_split function from the Scikit-learn library was used to create a training set and a test set. This function randomly partitioned the dataset into two subsets, maintaining the original class distribution. The resulting training set consisted of 70% of the total samples (43,740 samples), whereas the test set consisted of the remaining 30% (18,745 samples). A separate validation set was not created due to the implementation of 5-fold cross-validation. This technique allowed for the performance evaluation of the models to be more comprehensive and mitigated the impact of any potential biases in the data split.

E. Performance Metrics

Several quantitative metrics were deployed to evaluate the predictive performance of the models. Accuracy measures the overall fraction of correct predictions made by the model:

$$\text{Accuracy} = \frac{\text{Total Number of Predictions}}{\text{Number of Correct Predictions}} \quad (1)$$

Precision is given by the fraction of positive predictions that were positive.

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \quad (2)$$

Recall is the fraction of actual positive cases that were correctly predicted.

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

F1-score is the harmonic mean of precision and recall, providing a balance between the two metrics. It is the best at 1 and the worst at 0.

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision+Recall}} \quad (4)$$

The Receiver Operating Characteristic Area Under the Curve (ROC AUC) is a performance metric commonly applied to evaluate the ability of a classification model to distinguish between positive and negative classes across different thresholds. ROC AUC is calculated based on the True Positive Rate (sensitivity) and False Positive Rate. The formula for ROC AUC is:

$$\text{ROC AUC} \int_0^1 \text{TPR}\, d(\text{FPR}) \quad (5)$$

In discrete terms, the ROC AUC can be approximated as the area under the ROC curve, which is constructed by plotting the True Positive Rate (TPR - sensitivity) against the False Positive Rate (FPR) at various classification thresholds.

$$\text{ROC AUC} \approx$$

$$21 \sum_{i=1}^{N-1} (\text{TPR}_i + \text{TPR}_i + 1) \times (\text{FPR}_i + 1 - \text{FPR}_i) \quad (6)$$

where N is the number of thresholds, TPR_i is the True Positive Rate at the i-th threshold, and FPR_i is the False Positive Rate at the i-th threshold.

In practice, ROC AUC values range from 0 to 1. A higher value indicates better discrimination between positive and negative classes. A ROC AUC of 0.5 suggests a model performing no better than random chance, whereas a ROC AUC of 1.0 indicates perfect classification.

The Matthews Correlation Coefficient (MCC) is a measure used in binary classification to assess the quality of a classification model. It considers True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN) to provide a balanced evaluation even in the presence of class imbalance. MCC is given by:

$$\text{MCC} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP+FP})(\text{TP+FN})(\text{TN+FP})(\text{TN+FN})}} \quad (7)$$

MCC ranges from -1 to +1, where +1 indicates a perfect prediction, 0 suggests no better than a random prediction, and -1 indicates total disagreement between prediction and observation. Higher MCC values generally imply better classification performance.

IV. RESULTS AND DISCUSSION

The models were evaluated using five-fold cross-validation and the average performance across all folds is reported.

A. Performance Metrics

The results, as noticed in Table II and Figure 1, provide a critical insight into the competence of various ML algorithms to thwart ransomware attacks.

TABLE II. . PERFORMANCE EVALUATION

Algorithm	Accuracy	Precision	Recall	ROC AUC	MCC	F1
LR	72.46%	72.55%	72.46%	86.31%	57.55%	72.46%
DT	99.42%	99.42%	99.42%	99.59%	99.10%	99.42%
NB	71.92%	73.56%	71.92%	86.83%	58.11%	71.46%
RF	99.37%	99.37%	99.37%	99.99%	99.02%	99.37%
AdaBoost	87.93%	87.96%	87.93%	85.15%	81.32%	87.94%
XGBoost	99.48%	99.48%	99.48%	100.00%	99.19%	99.48%

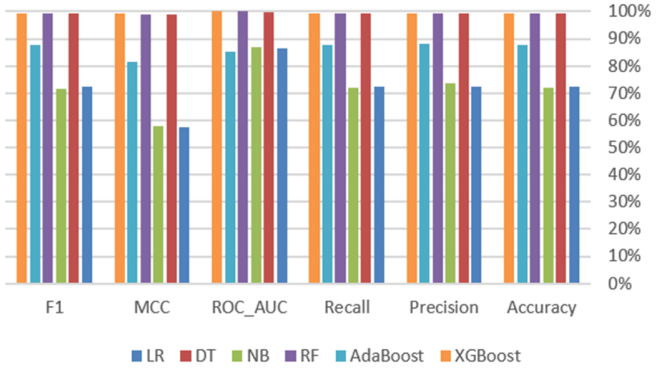

Fig. 1. Performance evaluation.

The LR exhibited moderate accuracy, precision, recall, and F1 scores (around 72%), but had a relatively high ROC AUC score, suggesting that it can distinguish between classes to a satisfactory extent. Its MCC is also moderate, indicating reasonable quality predictions. DT and XGBoost performed exceptionally well across all metrics, showing very high

accuracy, precision, recall, F1, and ROC AUC scores close to 1. This suggests that they are excellent at classifying ransomware accurately. NB had lower performance metrics compared to DT and XGBoost, but similar to LR, which might imply that it struggles with this dataset or type of problem. RF also manifested excellent performance, comparable to that of DT and XGBoost, which is not surprising given that it is an ensemble method that typically performs well in classification tasks. AdaBoost fell between high-performing ensembles and more moderate LR and NB, with decent scores but not as high as those of RF, DT, or XGBoost.

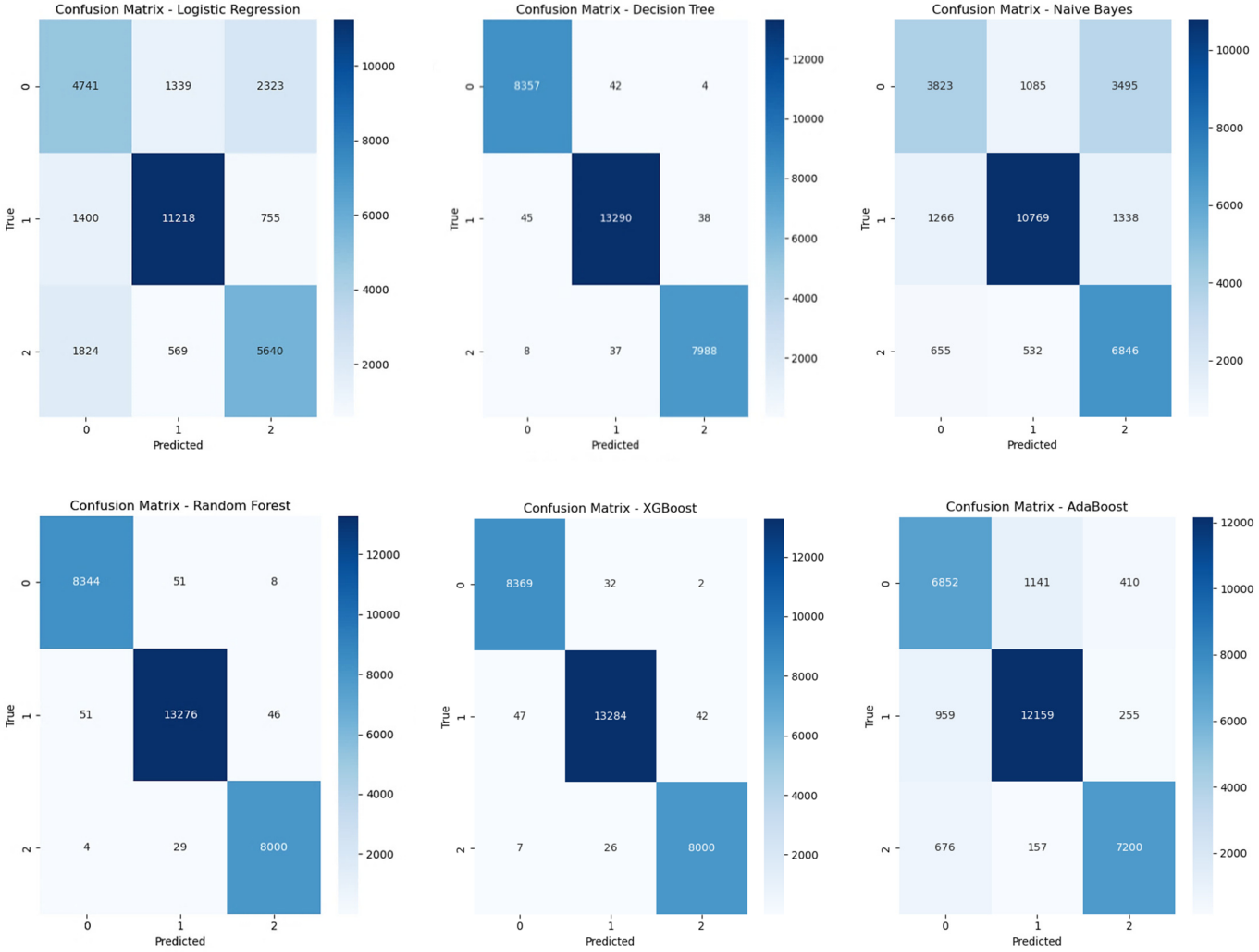

Fig. 2. Confusion matrices.

TABLE III. COMPUTATIONAL PERFORMANCE

Algorithm	Build Time	Training Time	Classification Speed	Computational Time	Kappa
LR	1.879834	1.888065	0.000000	1.887005	0.724580
DT	0.289654	0.286311	0.016478	0.300140	0.994096
NB	0.056549	0.044288	0.013017	0.057103	0.719179
RF	7.573886	7.330230	0.328042	7.717711	0.993693
AdaBoost	4.435372	4.389565	0.203077	4.600153	0.879298
XGBoost	9.407490	9.498313	0.046863	9.592122	0.994767

B. Computational Efficiency

This study extends to analyzing the computational performance of these algorithms, encompassing aspects, such as build time, training time, classification speed, overall computational time, and the Kappa statistic. Table III offers additional perspectives on the practical deployment of these algorithms. LR had the longest computational time, which included the training and classification speed, and a moderate

Kappa score, which measures the agreement of the predicted classification with the actual class. DT had a very low build and computational time, indicating that it was fast and had a very high Kappa score, suggesting strong agreement. NB demonstrated the shortest build and computational times, suggesting that it is the fastest algorithm among the evaluated ones but with a moderate Kappa score. RF took significantly longer in terms of build and computational times, which is expected given that it constructed multiple decision trees. AdaBoost and XGBoost had higher computational times, possibly due to the iterative nature of the boosting methods, but they both had very high Kappa scores.

DT and XGBoost were the best-performing algorithms for this ransomware detection task, balancing computational efficiency with high classification performance. However, the choice of algorithm can also be influenced by the specific requirements of the task, namely the need for speed versus the need for accuracy.

V. CONCLUSIONS

This study significantly advances the cybersecurity domain, particularly in ransomware threat detection. The findings accentuate the exceptional performance of the DT algorithm, with RF and XGBoost outshining simpler models such as NB and LR. These advanced algorithms demonstrate a commendable trade-off between accuracy, computational efficiency, and resilience, positioning them as superior for deployment in real-world scenarios. They proficiently navigate the intricate and mutable data patterns that are characteristic of ransomware threats, a challenge that proves too complex for less sophisticated algorithms. The comprehensive analysis utilizing a realistic dataset of ransomware signatures reveals that the RF and XGBoost algorithms stand out, reflecting high scores in accuracy, precision, recall, and F1. Their robustness is further emphasized by the high TP rates and minimal FNs, highlighting their precision in flagging ransomware instances. On the contrary, NB was found to produce a higher rate of FPs. LR and AdaBoost, while showing competent performance, were not without limitations. This study underscores the critical role of early and accurate ransomware detection in reducing the potential damage inflicted by such cyber threats.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the approval and support of this research study by grant no. SCIA-2023-12-2354 from the Deanship of Scientific Research at Northern Border University, Arar, KSA.

REFERENCES

[1] D. Dang, F. D. Troia, and M. Stamp, "Malware Classification using Long Short-term Memory Models," in *Proceedings of the 7th International Conference on Information Systems Security and Privacy*, Apr. 2024, pp. 743–752, https://doi.org/10.5220/0010378007430752.

[2] A. Moses and S. Morris, "Analysis of Mobile Malware: A Systematic Review of Evolution and Infection Strategies," *Journal of Information Security and Cybercrimes Research*, vol. 4, no. 2, pp. 103–131, Dec. 2021, https://doi.org/10.26735/KRVI8434.

[3] "Playing with Lives: Cyberattacks on Healthcare are Attacks on People," CyberPeace Institute, 2021.

[4] D. Hummer and J. M. Byrne, *Handbook on Crime and Technology*. Cheltenham, UK: Edward Elgar Publishing, 2023.

[5] M. Rigaki and S. Garcia, "Bringing a GAN to a Knife-Fight: Adapting Malware Communication to Avoid Detection," in *2018 IEEE Security and Privacy Workshops (SPW)*, San Francisco, CA, USA, May 2018, pp. 70–75, https://doi.org/10.1109/SPW.2018.00019.

[6] "Ponemon Institute reveals 68% of organizations were victims of successful endpoint attacks in 2019," *Security Info Watch*, Jan. 30, 2020. https://www.securityinfowatch.com/cybersecurity/press-release/21123576/ponemon-institute-ponemon-institute-reveals-68-of-organizations-were-victims-of-successful-endpoint-attacks-in-2019.

[7] "Ransomware Response: Time is More Than Just Money," *Security Intelligence*. https://securityintelligence.com/posts/ransomware-response-time-more-than-money/securityintelligence.com/posts/ransomware-response-time-more-than-money.

[8] K. Shaukat, S. Luo, V. Varadharajan, I. A. Hameed, and M. Xu, "A Survey on Machine Learning Techniques for Cyber Security in the Last Decade," *IEEE Access*, vol. 8, pp. 222310–222354, 2020, https://doi.org/10.1109/ACCESS.2020.3041951.

[9] J. Hwang, J. Kim, S. Lee, and K. Kim, "Two-Stage Ransomware Detection Using Dynamic Analysis and Machine Learning Techniques," *Wireless Personal Communications*, vol. 112, no. 4, pp. 2597–2609, Jun. 2020, https://doi.org/10.1007/s11277-020-07166-9.

[10] A. A. Alhashmi, A. M. Alashjaee, A. A. Darem, A. F. Alanazi, and R. Effghi, "An Ensemble-based Fraud Detection Model for Financial Transaction Cyber Threat Classification and Countermeasures," *Engineering, Technology & Applied Science Research*, vol. 13, no. 6, pp. 12433–12439, Dec. 2023, https://doi.org/10.48084/etasr.6401.

[11] A. Al-Marghilani, "Comprehensive Analysis of IoT Malware Evasion Techniques," *Engineering, Technology & Applied Science Research*, vol. 11, no. 4, pp. 7495–7500, Aug. 2021, https://doi.org/10.48084/etasr.4296.

[12] K. Aldriwish, "A Deep Learning Approach for Malware and Software Piracy Threat Detection," *Engineering, Technology & Applied Science Research*, vol. 11, no. 6, pp. 7757–7762, Dec. 2021, https://doi.org/10.48084/etasr.4412.

[13] S. Sechel, "A Comparative Assessment of Obfuscated Ransomware Detection Methods," *Informatica Economica*, vol. 23, no. 2, pp. 45–62, 2019, https://doi.org/10.12948/issn14531305/23.2.2019.05.

[14] H. Oz, A. Aris, A. Levi, and A. S. Uluagac, "A Survey on Ransomware: Evolution, Taxonomy, and Defense Solutions," *ACM Computing Surveys*, vol. 54, no. 11s, Jun. 2022, Art. no. 238, https://doi.org/10.1145/3514229.

[15] K. Lee, K. Yim, and J. T. Seo, "Ransomware prevention technique using key backup," *Concurrency and Computation: Practice and Experience*, vol. 30, no. 3, 2018, Art. no. e4337, https://doi.org/10.1002/cpe.4337.

[16] U. Urooj, B. A. S. Al-rimy, A. Zainal, F. A. Ghaleb, and M. A. Rassam, "Ransomware Detection Using the Dynamic Analysis and Machine Learning: A Survey and Research Directions," *Applied Sciences*, vol. 12, no. 1, Jan. 2022, Art. no. 172, https://doi.org/10.3390/app12010172.

[17] M. Masum, M. J. Hossain Faruk, H. Shahriar, K. Qian, D. Lo, and M. I. Adnan, "Ransomware Classification and Detection With Machine Learning Algorithms," in *2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, USA, Jan. 2022, pp. 0316–0322, https://doi.org/10.1109/CCWC54503.2022.9720869.

[18] E. Larsen, D. Noever, and K. MacVittie, "A Survey of Machine Learning Algorithms for Detecting Ransomware Encryption Activity." arXiv, Oct. 14, 2021, https://doi.org/10.48550/arXiv.2110.07636.

[19] Q. Chen, S. R. Islam, H. Haswell, and R. A. Bridges, "Automated Ransomware Behavior Analysis: Pattern Extraction and Early Detection," in *Science of Cyber Security*, Nanjing, China, 2019, pp. 199–214, https://doi.org/10.1007/978-3-030-34637-9_15.

[20] O. M. K. Alhawi, J. Baldwin, and A. Dehghantanha, "Leveraging Machine Learning Techniques for Windows Ransomware Network Traffic Detection," in *Cyber Threat Intelligence*, A. Dehghantanha, M. Conti, and T. Dargahi, Eds. Cham, Switzerland: Springer International Publishing, 2018, pp. 93–106.

[21] N. Pundir, M. Tehranipoor, and F. Rahman, "RanStop: A Hardware-assisted Runtime Crypto-Ransomware Detection Technique." arXiv, Nov. 24, 2020, https://doi.org/10.48550/arXiv.2011.12248.

[22] A. Singh, R. Ikuesan, and H. Venter, "Ransomware Detection using Process Memory," in *Proceedings of the 17th International Conference on Cyber Warfare and Security*, Albany, NY, USA, Mar. 2022, vol. 17, pp. 413–422, https://doi.org/10.34190/iccws.17.1.53.

[23] B. A. S. Al-rimy *et al.*, "Redundancy Coefficient Gradual Up-weighting-based Mutual Information Feature Selection technique for Crypto-ransomware early detection," *Future Generation Computer Systems*, vol. 115, pp. 641–658, Feb. 2021, https://doi.org/10.1016/j.future.2020.10.002.

[24] Y. Sahin and E. Duman, "Detecting credit card fraud by ANN and logistic regression," in *2011 International Symposium on Innovations in Intelligent Systems and Applications*, Istanbul, Turkey, Jun. 2011, pp. 315–319, https://doi.org/10.1109/INISTA.2011.5946108.

[25] A. S. Alraddadi, "A Survey and a Credit Card Fraud Detection and Prevention Model using the Decision Tree Algorithm," *Engineering, Technology & Applied Science Research*, vol. 13, no. 4, pp. 11505–11510, Aug. 2023, https://doi.org/10.48084/etasr.6128.

[26] Y. Ding, W. Kang, J. Feng, B. Peng, and A. Yang, "Credit Card Fraud Detection Based on Improved Variational Autoencoder Generative Adversarial Network," *IEEE Access*, vol. 11, pp. 83680–83691, 2023, https://doi.org/10.1109/ACCESS.2023.3302339.

[27] M. Wa Nkongolo, "UGRansome Dataset." Kaggle, https://doi.org/10.34740/KAGGLE/DSV/7172543.

[28] M. Tokmak, "Deep Forest Approach for Zero-Day Attacks Detection," in *Innovations and Technologies in Engineering*, S. Tasdemir and I. Ali Ozkan, Eds. Istanbul, Turkey: Eğitim Yayinevi, 2022.

[29] D. Shankar, G. V. Sudha, J. N. S. S. Naidu, and P. S. Madhuri, "Deep Analysis of Risks and Recent Trends Towards Network Intrusion Detection System," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 1, pp. 262-276, 2023, https://doi.org/10.14569/IJACSA.2023.0140129.