A Lightweight Obfuscated Malware Multi-class Classifier for IoT Using Machine Learning

William Cassel, Nahid Ebrahimi Majd
Department of Computer Science and Information Systems
California State University San Marcos, United States
Casse017@csusm.edu, nmajd@csusm.edu

***Abstract*—The rapidly growing number of obfuscated malware attacks in the past few years has emerged as a significant threat for organizations and individuals, demanding prompt action to develop systems that accurately detect these attacks to block them or mitigate their impacts. These types of malwares use obfuscation techniques to hide their malicious functionalities from intrusion detection systems, which makes their detection more complicated than regular malwares. Most of the obfuscated malware detection systems primarily focus on binary classification. The existing multi-class classification methods mainly have used CNN-based deep learning to improve the model's accuracy. However, this approach is not suitable for resource constrained network nodes, such as IoT devices, which are widely used on the Internet to monitor and control different environments. To tackle this issue, in this paper, we propose a lightweight model that accurately and efficiently classifies benign traffic vs. different classes of obfuscated malwares. Our proposed model uses a hybrid method of SMOTE oversampling to synthetically create training records for the minority classes in combination with undersampling the majority class via Tomek Links algorithm to increase the model's performance in malware classification. W applied this hybrid data augmentation technique to our training dataset extracted from CIC-MalMem2022 dataset to build a Random Forest model. Our experimental results demonstrated that the proposed model outperforms the state-of-the-art with 87.1% accuracy in classifying obfuscated malwares.**

Keywords— Obfuscated Malware Classification; Network security; Machine Learning; Data Augmentation;

I. INTRODUCTION

During 2022, the worldwide number of malware attacks reached 5.5 billion [1]. Malware is a type of malicious software that aims to disrupt a system's activity, steal sensitive information, or cause harm in some way to a network or individual device. Obfuscated malware is a type of malware that hides its disruptive functionality from malware detectors. One of the main malware obfuscation detection techniques is memory analysis, i.e., by extracting aggregated data on memory handles, code injections, memory modules, etc. In our proposed framework, we use CIC-MalMem2022 dataset, created by extracting such features from memory dumps using an extension to VolMemLyzer tool [2].

This dataset presents data for benign traffic and three types of malwares: (1) Ransomware, (2) Spyware, and (3) Trojan. Ransomware gains the control of the victim machine's system file, encrypts the files, and holds the encryption key hostage. Oftentimes Ransomware attacks are extremely difficult to reverse and may even overwrite or delete all files even if the ransom is paid. Spyware aims to secretly infiltrate a network in order to collect sensitive data in the background without raising attention. The sensitive data collected by Spyware can range from passwords to social security numbers and can result in significant financial harm. A Trojan is a type of malware that disguises itself as a legitimate program to remain undetected. Once run, this seemingly innocent program can perform a wide variety of harmful actions from data theft to creating backdoor access for other attacks to take place on the infiltrated system.

Malware obfuscation is a technique that malware authors use to make their malware difficult to detect or analyze. There is a variety of techniques that malware authors use for this purpose. Most of these techniques mainly aim to hide a critical string in the malware code, which reveals the patterns of the malwares' behavior. Often, they use packing technique to compress the executable code and make it difficult for intrusion detection systems to reveal its pattern. Another technique is inserting a dead-code to change the malware's pattern.

The existing research on obfuscated malware multi-class classification have used CNN-based models, which are more suitable for an intrusion detection system located at the network edge rather than a resource constrained device, such as an IoT. In our models, rather than computationally expensive deep learning models, we augmented the training data at the pre-processing phase, and created a light-weight machine learning model, which outperforms the existing CNN-based models.

The main contributions of our research are as following.

1. In order to cope with the data imbalanced issue, we applied a hybrid data augmentation method of SMOTE and Tomek Links algorithm to the training dataset.
2. We used this augmented training dataset to build a lightweight multi-class classification machine learning model that classifies Benign vs. 3 types of obfuscated malwares: Ransomware, Spyware, and Trojan.
3. Our experimental results demonstrated that the proposed model outperforms the state of the art in terms of accuracy, precision, recall, and F1-Score.

The rest of this paper is organized as the following. Section II describes the related work. Section III explains the methodology, including the dataset and preprocessing. Section IX explains the proposed machine learning models and hyperparameter tunning. Section X presents the performance measures. Section XI presents the results and discussion. Section XII draws the conclusion.

979-8-3503-7099-7/24/$31.00 ©2024 IEEE

Table 1: A summary of accuracies of existing binary and family classifications research on CIC-MalMem2022

	Reference	Year	Model	Acc
Binary Classification	[3] Carrier et al.	2022	EL	99
	[4] Dener et al.	2022	LR	99.97
	[5] Ghazi et al.	2022	RF	99.99
	[6] Smith et al.	2023	RF, DT, ET, AB	99.99
	[7] Talukder et al.	2023	RF, ANN	100
	[8] Louk et al.	2022	RF	100
	[9] Mezina et al.	2022	DCNN	99.92
	[10] Shafin et al.	2023	CompactCBL RobustCBL	99.92 99.96
Family Classification	[9] Mezina et al.	2022	Decision Tree DCNN	79.16 83.53
	[10] Shafin et al.	2023	CompactCBL RobustCBL	84.22 84.56

II. Related Work

Machine learning has been used to classify malware attacks. Carrier et al. [2] created CIC-MalMem2022, a popular dataset on obfuscated malware memory data, which is publicly available at the Canadian Institute for Cybersecurity website. The authors extended the VolMemLyzer tool to extract hidden and obfuscated memory features. This dataset is constructed of benign and three types of malwares, namely Spyware, Ransomware, and Trojan horse. The authors also developed a stacked ensemble learning framework for a binary classification of benign vs. malware.

A variety of research work has studied obfuscated malware classification using CIC-MalMem2022 dataset. They can be categorized to binary and multi-class (family) classification models. [3] - [10] worked on **binary classification** to classify benign vs. malware and achieved accuracies in range [99% - 100%]. Table 1 presents a summary of existing binary and multi-class classifications research on CIC-MalMem2022 dataset.

Authors of [9] and [10] studied **multi-class classification** as well to classify the traffic to either benign or one of the three malware classes: Ransomware, Spyware, or Trojan. Mezina et al. [9] achieved the best accuracy of 83.53% for their Dilated CNN model and 79.1% for their DT model. Shafin et al. [10] achieved the best accuracy of 84.22% for their CompactCBL (Compact CNN-BiLSTM) model and 84.56% for their RobustCBL (Robust CNN-BiLSTM) model. Our research is multi-class classification, and we compare our results with RobustCBL [10], which is the state-of-the-art on multi-class classification on this dataset. All these models are CNN-based. [10] used CNN and bidirectional LSTM to propose their models. Although these models achieved good accuracies, the complex structure of CNN and LSTM rises the training and prediction times. To address this issue, we propose a lightweight model with higher accuracy and simpler structure.

Table 2: Number of records in each family

Class	Count	Percentage
Benign	29,298	50%
Ransomware	9,791	17.1%
Spyware	10,020	16.7%
Trojan	9,487	16.2%
Total	**58,596**	**100%**

III. Methodology

A. Dataset

We used the CIC-MalMem2022 dataset [3] for our research, which consists of memory analysis features of real network scenarios of obfuscated malware attacks. This dataset consists of four different classes: Benign, Ransomware, Spyware, and Trojan. There are 29,298 Benign records, 9,791 Ransomware records, 10,020 Spyware records, and 9,487 Trojan records for a total of 58,596 entries in the roughly 19 MB dataset. Each record has 56 features, 54 of which are numerical. The remaining two features describe the class and subclass of the record, with each class containing five subclasses. The dataset classes are broken down in Table 2. In our problem, we build a model that classifies the traffic to 4 classes.

B. SMOTE

Synthetic Minority Over-sampling Technique (SMOTE) is a data augmentation technique used to synthetically oversample the minority classes. It first selects a random record from a minority class, then selects neighboring records of that class and creates a new synthetic record at a random point between one of the random neighbors and the first selected record. We first split out dataset to 70% training set (41,017 records) and 30% test set (17,579 records). Then, we applied SMOTE on the training set to get a synthetic distribution of records for 4 classes (benign and 3 types of malwares).

SMOTE has been used in other research studies to oversample imbalanced datasets. Basgall et al. [11] and Ramentol et al. [12] studied the effectiveness of SMOTE on sparse and imbalanced datasets. Ma et al. [13] studied SMOTE on Bioinformatics datasets. Seo et al. [14] studied SMOTE on Intrusion Detection datasets. Kudugunta et al. [15] and Gonzalez et al. [16] studied SMOTE on Botnet attacks datasets. These studies demonstrated the effectiveness of SMOTE on machine learning models on imbalanced datasets.

Table 3: Number of records in each class **before** applying SMOTE&TL

Class	Train records	Train portion	Test records	Test portion
Benign	20,509	35%	8,790	15%
Ransomware	6,854	11.7%	2,937	5%
Spyware	7,014	12%	3,006	5%
Trojan	6,641	11.3%	2,846	5%
Total	**41,018**	**70%**	**17,579**	**30%**

Table 4: Number of records in each class **after** applying SMOTE&TL

Class	Train records	Train portion	Test records	Test portion
Benign	20,509	21%	8,790	9%
Ransomware	20,276	20%	2,937	3%
Spyware	20,316	20.5%	3,006	3%
Trojan	20,352	20.5%	2,846	3%
Total	**81,450**	**82%**	**17,579**	**18%**

C. *Tomek Links (TL)*

The Tomek Links algorithm aims to undersample the majority class by finding samples of the majority class that most resemble samples from the minority classes, and then removes that record from the dataset. After SMOTE, we applied Tomek Links algorithm on the training set.

After applying the hybrid method of SMOTE and Tomek Links algorithm (SMOTE&TL), the training set consisted of 81,450 records with roughly 20,300 records for each family, which is almost 25% of the dataset records. Tables 3 and 4 illustrate the data distribution of classes before and after applying SMOTE&TL on the training set. Note that we apply SMOTE&TL only on the training set, not the test set.

D. *Data Standardization*

To standardize features, we applied the StandardScaler method from scikit learn library, which scales to unit variance. The standard score of a sample x is calculated as z = (x-u)/s, where u is the mean of the training samples and s is the standard deviation of the training samples.

IV. Hyperparameter Tunning

We applied the SMOTE&TL data augmentation technique to the training dataset and then used it to build two machine learning models: Random Forest (RF) and Extra Tree (ET). Then, we tunned hyperparameters for each model to get the best performances. We used grid search for hypertunning. We did not observe any sign of overfitting in our models during hypertunning. Table 5 illustrates the tunned hyperparameters for each model.

V. Performance Measures

To analyze the effectiveness of our proposed machine learning models, we used the standard metrics.

1. TP (True Positive): An instance of class B was correctly predicted to be in class B.

Table 5: Tunned hyperparameters for each ML model

ML	Hyperparameters
RF	n_estimators = 100, criterion = "log_loss"
ET	n_estimators = 100, criterion = "gini", random_state = 0

2. FP (False Positive): An instance of a non-B class was incorrectly predicted to be in class B.

3. TN (rue Negative): An instance of a non-B class was correctly predicted to be in a non-B class.

4. FN (False Negative): An instance of class B was incorrectly predicted to be in a non-B class.

Accuracy: Accuracy measures the proportion of the correctly predicted instances of a class to the total number of predictions in that class. It is measured by equation 1.

$$Accuracy = (TP + TN)/(TP + FP + FN + TN) \quad (1)$$

Precision: Precision measures the proportion of the correctly predicted instances of a class to the total number of instances that were predicted to be in that class, either correctly or incorrectly. It is measured by equation 2.

$$Precision = TP/(TP + FP) \quad (2)$$

Recall: Recall measures the proportion of the correctly predicted instances of a class to the total number of instances in that class that were provided. It is measured by equation 3.

$$Recall = TP/(TP + FN) \quad (3)$$

F1 score: F1 score measures the 'Harmonic mean' of precision and recall values. It is measured by equation 4.

$$F1\ score = (2\times(precision \times recall))/(precision + recall) \quad (4)$$

VI. Results and Analysis

The state-of-the-art on obfuscated malware binary classification, which aims detecting obfuscated malware vs. benign traffic has been promising. Some proposed models [7], [8] achieved 100% accuracy. However, to effectively combat the malicious functionality of malware traffic, it is essential to detect the malware class, which gives accurate information about the nature of the malware and the way it infects the network devices and spreads in the network. This information could help mitigate the malware's destructive impacts on the network devices. Our problem is multi-class classification to detect the Benign traffic vs. one of the three types of obfuscated malwares: (1) Ransomware, (2) Spyware, and (2) Trojan.

We explored several machine learning models, including XGBoost and Decision Tree, and hypertunned them. Here, we present the best results, which we achieved from our Random Forest and Extra Tree models. We also explored different feature selection techniques with different numbers of features, which did not make significant improvement to our best models. We created two machine learning models to test the effectiveness of SMOTE&TL technique data augmentation on obfuscated malware multi-class classification. We used Random Forest (RF) and Extra Tree (ET) machine learning algorithms for our models. Table 6 presents the results. The proposed two models achieved 87.1% (RF with SMOTE&TL) and 86.1% (ET with SMOTE&TL) multi-class classification accuracies and outperformed the state-of-the-art.

Table 6: Comparison of the proposed models and the-state-of-the-art.

No	Model	Accuracy	Precision	Recall	F1-Score
1	Random Forest with SMOTE&TL	0.871	0.871	0.871	0.871
2	Extra Tree with SMOTE&TL	0.861	0.861	0.862	0.862
3	RobustCBL [10]	0.8456	0.85	0.85	0.84
4	CompactCBL [10]	0.8422	0.84	0.84	0.84
5	DCNN [9]	0.8353	0.76	0.75	0.75
6	Decision Tree [9]	0.7916	0.69	0.69	0.69

The most accurate existing models for obfuscated malwares multi-class classification are RobustCBL [10] with 84.56% accuracy, CompactCBL [10] with 84.22% accuracy and DCNN [9] with 83.53% accuracy, which are much less accurate than our proposed models. The models proposed by [9] and [10] are all CNN-based deep learning models, which are heavyweight for an IoT device. However, our models are lightweight and remarkably more accurate, which make them suitable for resource constrained IoT devices.

We implemented and tested our models using Google Colab with Tesla 4 GPU. Our RF with SMOTE&TL model was built in 33.53 seconds. It took 0.36 seconds for this model to classify the test set's instances. Our ET with SMOTE&TL model was built in 12.69 seconds, which is shorter than the first model. However, it took 0.49 seconds for this model to classify the test set's instances, which is more than the first model. Thus, we select our RF with SMOTE&TL model as our best model, a lightweight model providing the highest accuracy and low prediction time, suitable for a resource constrained IoT device. We also ran extensive experiments on other machine learning algorithms, e.g., XGBoost and Decision Tree. However, the proposed models outperformed all other models.

Table 6 illustrates that our RF with SMOTE&TL model remarkably outperforms other models in terms of precision, recall, and F1-Score as well. We studied these metrics with more details to investigate the performance of our model in classifying each type of obfuscated malware. Table 8 presents the confusion matrix for our best model. Classes 0, 1, 2, 3, and 4 are Benign, obfuscated Ransomware, obfuscated Spyware, and obfuscated Trojan respectively.

Table 7 illustrates the precision, recall, and F1-Score per class in our best model and compares them with the state-of-the-art, which are the models proposed by [10]. The results reveal that all classifiers accurately classify almost all Benign traffic. This could be associated with the absence of obfuscation in Benign samples, which make them easier to classify. It could also be associated with the original number of Benign samples in the training set, which is much higher than the original obfuscated malware samples.

All models demonstrate the highest F1-Score for obfuscated Spyware (78% for the proposed model and 73% and 72% for [10]), and lower F1-Scores for obfuscated Trojan (72% for the proposed model and 70% and 71% for [10]) and obfuscated Ransomware (73% for the proposed model and 64% and 63% for [10]).

This could be associated with strong association between the obfuscation features of the dataset [3] and the Spyware class. This indicates despite obfuscation the model still successfully classifies a large portion of obfuscated Spywares. On the other hand, Ransomwares and Trojans are more successful in obfuscation, and the memory analysis features cause misclassifications for many samples in these two classes.

Overall, the proposed model remarkably outperforms the models of [10] in classifying malwares in terms of F1-Score. A closer look shows that the reason why the models of [10] present less performance is low precision in classifying some classes of obfuscated malware and low recall in other classes. For example, RobustCBL [10] demonstrates low precision of 69% in classifying obfuscated Spyware, indicating high False Positives in this class, meaning this model incorrectly classifies several non-Spywares to be Spywares.

Also, this model demonstrates low recall of 62% in classifying obfuscated Ransomware, indicating high False Negatives in this class, meaning this model incorrectly classifies several Ransomwares to be non-Ransomwares. On the other hand, our proposed model presents the same high precisions and recalls in range [72% - 78%] for each class, indicating low False predictions in our proposed model, meaning the number of False Positives and False Negatives in each class is almost the same and relatively low.

This can be observed in Table 8 as well. This makes sense because the SMOTE&TL data augmentation that we initially applied to the training dataset added a large number of simulated samples to the training set, giving more opportunity to the model to get trained with high number of samples in each class and correctly classify the class of each malware at prediction time.

Table 7: Family-wise classification performances of our proposed model comparing to [10]

	Proposed RF with SMOTE&TL			RobustCBL [10]			CompactCBL [10]		
Class	**Precision**	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**
Benign	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Ransomware	0.73	0.73	0.73	0.67	0.62	0.64	0.67	0.60	0.63
Spyware	0.78	0.78	0.78	0.69	0.77	0.73	0.70	0.72	0.72
Trojan	0.72	0.72	0.72	0.71	0.67	0.70	0.68	0.74	0.71

Table 8: Confusion Matrix for RF with SMOTE&TL

		Predicted label			
Actual label		**Class 0**	**Class 1**	**Class 2**	**Class 3**
	Class 0	8788	0	2	0
	Class 1	0	2146	349	442
	Class 2	0	308	2329	369
	Class 3	0	515	272	2059

VII. Conclusion

In this research, we proposed a machine learning model for an intrusion detection system that classifies obfuscated malwares. The proposed model is lightweight, suitable for resource constrained IoT devices on the Internet. We applied a hybrid method of SMOTE oversampling and Tomek Links undersampling to our training dataset extracted from CIC-MalMem2022 dataset and built the model with that. Our experimental results demonstrated that the proposed model remarkably improved the performance comparing to the state-of-the-art in terms of accuracy, precision, recall, and F1-Score. We also analyzed the performance of our proposed model in classifying each obfuscated malware class in terms of class-wise precision, recall, and F1-Score.

References

[1] Available online, https://www.statista.com/statistics/873097/malware-attacks-per-year-worldwide/ (accessed on July 30, 2023).

[2] A.H. Lashkari, B. Li, T.L. Carrier, and G. Kaur, "Volmemlyzer: Volatile memory analyzer for malware classification using feature engineering," 2021 IEEE Reconciling Data Analytics, Automation, Privacy, and Security: A Big Data Challenge (RDAAPS), 2021. doi: 10.1109/RDAAPS48126.2021.9452028.

[3] T. Carrier, P. Victor, A. Tekeoglu, and A.H. Lashkari, "Detecting Obfuscated Malware using Memory Feature Engineering," ICISSP, 2022. doi: 10.5220/0010908200003120.

[4] M. Dener, G. Ok, and A. Orman, "Malware detection using memory analysis data in big data environment," Applied Sciences, 2022. doi: 10.3390/app12178604.

[7] M.A. Talukder, K.F. Hasan, M.M. Islam, M.A. Uddin, A. Akhter, M.A. Yousuf, F. Alharbi, and M.A. Moni, "A dependable hybrid machine learning model for network intrusion detection," Journal of Information Security and Applications, 2023. doi: 10.1016/j.jisa.2022.103405.

[8] M.H.L. Louk, and B.A. Tama, "Tree-based classifier ensembles for PE malware analysis: A performance revisit," Algorithms, 2022. doi: 10.3390/a15090332.

[6] D. Smith, S. Khorsandroo, and K. Roy, "Supervised and unsupervised learning techniques utilizing malware datasets," IEEE 2nd International Conference on AI in Cybersecurity (ICAIC), 2023. doi: 10.1109/ICAIC57335.2023.10044169.

[5] M.R. Ghazi and N.S. Raghava, "Machine Learning Based Obfuscated Malware Detection in the Cloud Environment with Nature-Inspired Feature Selection," IEEE 5th International Conference on Multimedia, Signal Processing and Communication Technologies (IMPACT), 2022. doi: 10.1109/IMPACT55510.2022.10029271.

[9] A. Mezina and R. Burget, "Obfuscated malware detection using dilated convolutional network," 14th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2022. doi: 10.1109/ICUMT57764.2022.9943443.

[10] S.S. Shafin, G. Karmakar, and I. Mareels, "Obfuscated Memory Malware Detection in Resource-Constrained IoT Devices for Smart City Applications," Sensors, vol. 23, no. 11, p. 5348, Jun. 2023, doi: 10.3390/s23115348.

[11] M.J. Basgall, W. Hasperué, M. Naiouf, A. Fernández, and F. Herrera, "SMOTE-BD: An exact and scalable oversampling method for imbalanced classification in big data," Journal of Cloud Computing & Big Data (JCC&BD), 2018.

[12] E. Ramentol, Y. Caballero, R. Bello, and F. Herrera, "Smote-rsb: a hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using smote and rough sets theory," Knowledge and information systems, 2012. doi: 10.1007/s10115-011-0465-6.

[13] L. Ma, and S. Fan, "CURE-SMOTE algorithm and hybrid algorithm for feature selection and parameter optimization based on random forests," BMC bioinformatics, 2017. doi: 10.1186/s12859-017-1578-z.

[14] J.H. Seo, and Y.H. Kim, "Machine-learning approach to optimize smote ratio in class imbalance dataset for intrusion detection," Computational intelligence and neuroscience, 2018. doi: 10.1155/2018/9704672.

[15] S. Kudugunta, and E. Ferrara, "Deep neural networks for bot detection," Information Sciences, 2018. doi: 10.1016/j.ins.2018.08.019.

[16] D. Gonzalez-Cuautle, A. Hernandez-Suarez, G. Sanchez-Perez, L.K. Toscano-Medina, J. Portillo-Portillo, J. Olivares-Mercado, H.M. Perez-Meana, and A.L. Sandoval-Orozco, "Synthetic minority oversampling technique for optimizing classification tasks in botnet and intrusion-detection-system datasets," Applied Sciences, 2020. doi: 10.3390/app10030794.

[17] Y. Wang, H. Yao, and S. Zhao, "Auto-encoder based dimensionality reduction," Neurocomputing, 2016. doi: 10.1016/j.neucom.2015.08.104.

[18] Available online, https://github.com/chasedehan/BoostARoota/blob/master/README.md (accessed on July 30, 2023).