IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 5 January 2024, accepted 17 January 2024, date of publication 26 January 2024, date of current version 6 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3358840

RESEARCH ARTICLE

Obfuscated Privacy Malware Classifiers Based on Memory Dumping Analysis

DAVID CEVALLOS-SALAS[1], FELIPE GRIJALVA[2], (Senior Member, IEEE), JOSÉ ESTRADA-JIMÉNEZ[1], DIEGO BENÍTEZ[2], (Senior Member, IEEE), AND ROBERTO ANDRADE[3], (Member, IEEE)

[1]Departamento de Electrónica, Telecomunicaciones y Redes de Información, Facultad de Ingeniería Eléctrica y Electrónica, Escuela Politécnica Nacional, Quito 170525, Ecuador
[2]Colegio de Ciencias e Ingenierías "El Politécnico," Universidad San Francisco de Quito (USFQ), Quito 170901, Ecuador
[3]Departamento de Informática y Ciencias de la Computación, Facultad de Ingeniería en Sistemas, Escuela Politécnica Nacional, Quito 170525, Ecuador

Corresponding author: David Cevallos-Salas (david.cevallos03@epn.edu.ec)

This work was supported in part by Escuela Politécnica Nacional, and in part by Universidad San Francisco de Quito (USFQ) through the Poli-Grants Program under Grant 24263.

ABSTRACT Malware targeting user privacy has seen a surge in recent times, attributed to evolving global regulations and the boost of electronic commerce and online services. Moreover, recognizing privacy malware that employs obfuscation as evasion mechanism presents a major challenge due to its dynamics, resilience, and polymorphism at runtime, necessitating the application of forensic techniques such as memory dumping analysis in order to reveal suitable patterns and behaviors that enable its subsequent detection and classification. In this paper, we present three obfuscated privacy malware classifiers trained on the CIC-MalMem-2022 dataset. These solutions include a binary classifier to distinguish benign from malicious samples using logistic regression (LR), a multiclass classifier that further categorizes benign, spyware, ransomware, and trojan horse obfuscated privacy malware; and a more detailed multiclass classifier capable of discriminating benign samples from fifteen specific obfuscated privacy malware families. Multiclass classifiers were built using several traditional machine learning algorithms and a novel Deep Neural Network (DNN). We applied the Synthetic Minority Oversampling Technique (SMOTE) to address data imbalances. In particular, our results demonstrate that DNN outperforms traditional machine learning algorithms, yielding statistically significant improvements in key metrics. These achievements reach practical thresholds, suggesting the potential for enhanced malware protection systems. The dataset and all the coding files required for experiments reproducibility are publicly available at https://github.com/dcevallossalas/PrivacyMalwareClassifiers.

INDEX TERMS Privacy, malware, obfuscation, classifier, memory dumping, CIC-MalMem-2022, SMOTE, ransomware, spyware, trojan horse.

I. INTRODUCTION

Privacy in the cyberworld is an inherent right of people that must be protected through legal and technological efforts. In this context, several laws, such as the General Data Protection Regulation (GDPR) in Europe and the Health Insurance Portability and Accountability Act (HIPAA) in the United States, seek to protect personal data's acquisition, storage, and processing [1].

Nevertheless, these legal efforts have been severely reduced through specialized malware focused on violating the privacy of its victims, either through direct attacks against users themselves or against agents authorized to treat their personal data [2].

As can be deduced from [3], there are currently three main categories of malware with specialized families that seek to violate the privacy of users: spyware, ransomware, and trojan horses. Spyware uses techniques such as snooping and eavesdropping to collect data, and it is the type of malware that most threatens the privacy of users [4]. Ransomware,

The associate editor coordinating the review of this manuscript and approving it for publication was Lorenzo Mucchi.

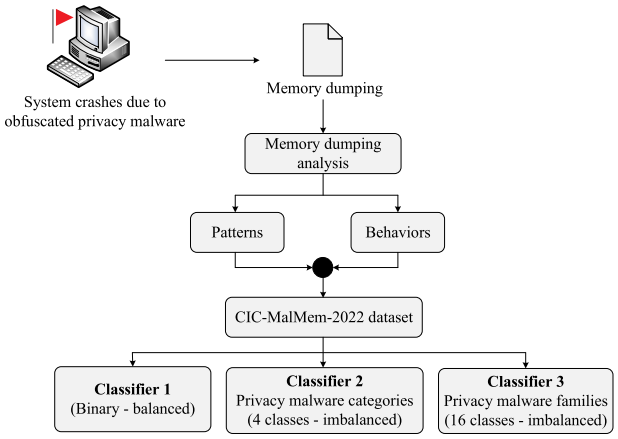

FIGURE 1. Memory dumping analysis for obfuscated privacy malware classification.

in the context of privacy, has a unique feature that allows it to open command and control communications to an attacker's work station in order to exfiltrate data prior to encrypting them [5]. Finally, trojan horse malware can install backdoors that allow it to snoop and steal data while deceiving the user by pretending to be a valid application [6].

Cyber attacks based on these three categories of privacy malware have been potentially increased due to the rise in electronic commerce and the provision of online services [7]. Although each category of privacy malware has its own working mechanisms and architecture, most current specialized spyware, ransomware and trojan horses families share the common feature that, once executed by a legitimate operating system process, they are likely impossible to detect by security controls until the malware has entirely or partially completed its target [8]. The main reason for this is that they employ obfuscation techniques as an evasion mechanism. In general, existing privacy malware detection techniques are mainly oriented to recognize non-obfuscated privacy malware, and there is no currently a clear and unified technique to detect and understand the patterns and behaviors of obfuscated privacy malware at runtime [3], [9].

As Fig. 1 exposes, after a system has been affected due to obfuscated privacy malware, dumping the memory and analyzing it can reveal specific patterns and behaviors experienced by the system processes to build an obfuscated privacy malware classifier. For instance, the number of handlers opened by the operative system on request of the process, the number of opened sockets for communication to remote sites; and the number of mutex and semaphores used are attributes that can help to detect obfuscated privacy malware categories and even its family sub-classification. The contributions presented by Lashkari et al. [10] and Mu et al. [11] detail a complete guide for this procedure.

However, traditional programming methods cannot deduce a general rule followed by the different categories and families of obfuscated privacy malware on their patterns and behaviors. Therefore, machine learning algorithms, and especially DNNs, are commonly used to analyze obfuscated privacy malware.

For that reason, this paper presents three obfuscated privacy malware classifiers capable of achieving comparable metrics without using advanced techniques proposed by other researchers.

The first classifier addresses the binary case and allows to distinguish whether a process presents general patterns and behaviors of a benign or malicious infection simply using a logistic regression model.

The second and third classifiers are multiclass. While the second classifier analyzes four obfuscated privacy malware categories: benign, spyware, ransomware, and trojan horse, the third classifier includes sixteen categories: benign and five specific obfuscated privacy malware families of spyware, ransomware, and trojan horses, respectively. Deep learning techniques have been used in this research to address the problem of multiclass classification.

A. MOTIVATION

Obfuscated malware encompasses malicious software that relies on concealing its characteristics, functionality, and actions as its primary technique to evade security mechanisms at runtime. Recognizing obfuscated malware poses a significant challenge, as it proves resistant to methods based on signature analysis employed by security controls such as antivirus, IDS (Intrusion Detection System), and IPS (Intrusion Protection System) engines [12]. This resistance results in lower detection metrics compared to those achieved against non-obfuscated malware. Moreover, the resilience and the ability of this type of malware to mutate and manifest itself in different ways once executed (polymorphism) add to the difficulty of establishing patterns and behaviors for recognition [13]. In addition, addressing multiclass classification with obfuscated malware is exceptionally complex, as it hinders the establishment of features that enable a clear boundary between different categories and families of malware [14]. In order to understand the nature of an attack and implement effective security controls and countermeasures, it is overriding to know not only the privacy malware category but also the family causing the attack [15].

While obfuscation as an evasion technique can be employed by several types of malware, most specialized privacy malware extensively utilizes obfuscation exerting a more significant impact on users and organizations compared to malware oriented to attack other security tenets such as the information integrity or its availability [16]. This is due to its dynamic patterns and behaviors strategically designed to facilitate the smoothly snoop, theft and extraction of personal data, ultimately leading to a data leakage [8].

In order to gather information enabling the subsequent detection and classification of obfuscated privacy malware, various techniques have been proposed. These include Domain Generation Algorithms (DGA) analysis [17],

patterns recognition for command and control sequences [18], analysis of Domain Name Systems (DNS) patterns for hosting malware [19], among others. However, memory dumping analysis stands out as the most promising [8] due to it enables the examination of postmortem scenarios, offering insights into the malware's primary features and behaviors after it has completed its entire attack sequence [3], [11].

The CIC-MalMem-2022 is a robust and modern dataset built on memory dumping analysis that exposes general patterns and behaviors for detecting obfuscated privacy malware at runtime [20]. In contrast to previous related work utilizing this dataset to address the challenges of obfuscated privacy malware through the application of ensemble classifiers or complex DNNs for binary classification, as well as resource-intensive Convolutional Neural Networks (CNNs) for multiclass classification, our approach takes a distinct and innovative route. We employ preprocessing and data augmentation techniques as input for more suitable models, deviating from the use of highly intricate architectures (difficult to interpret, implement, and resource-consuming) yet achieving comparable metrics. This novel strategy offers an alternative perspective on tackling the obfuscated privacy malware detection and classification challenges.

B. CONTRIBUTIONS

This research presents the following major contributions:

- A model of a binary obfuscated privacy malware classifier based on logistic regression. This model is able to achieve higher metrics than those reached by [8] using traditional machine learning algorithms, and comparable metrics to those achieved by the same author using complex DNNs. Therefore, our logistic regression model comprises a good solution for benign and malicious classification built and trained without the overhead caused by using advanced DNNs. Also, in comparison to [3], this logistic regression model achieves a high metric without using ensemble classifiers comprising a standalone solution.
- A novel and simplified DNN architecture for multiclass obfuscated privacy malware classifiers that can achieve metrics comparable to those reached by advanced DNNs architectures. In this sense, our multiclass classifiers are able to achieve comparable metrics to those exposed by [15] and [21], but without using advanced CNNs nor ensemble classifiers.
- A comparison of the metrics achieved by multiclass ensemble classifiers, constructed through a boosting sequence between the logistic regression binary classifier and specific multiclass classifiers (excluding the benign class), with those attained by our proposed standalone multiclass classifiers.
- A statistical comparison between the metrics values reached by traditional machine learning algorithms with our proposed DNN for each multiclass obfuscated privacy malware classifier.

Overall, the results obtained demonstrate that our DNN achieves statistically significant higher metrics than those reached by traditional machine learning algorithms for both obfuscated privacy malware category and family classifiers. Furthermore, our multiclass standalone classifiers achieve better results than ensemble classifiers.

C. PAPER ORGANIZATION

The remainder of the paper is organized as follows. In Section II is presented the related work that has been carried out by several researchers. In Section III we explain the methodology followed in this research. In particular, the CIC-MalMem-2022 dataset and the materials and tools that have been used are presented in detail. In Section IV the results and analysis of the research are described, and a discussion is carried out in Section V. Finally, the research conclusions and future work are presented in Section VI.

II. RELATED WORK

Since obfuscated privacy malware detection and classification is a relevant area, prior work and contributions have been made by several researchers. Likewise, various datasets based on different data-gathering methods have been used for this purpose, including but not limited to memory dumping analysis.

Regardless of the dataset used, the methods for data collection and classification, and the obfuscated privacy malware categories and families that each classifier aims to recognize, it is feasible to analyze each contribution and determine its limitations.

Table 1 presents a comparison of existing contributions related to obfuscated privacy malware detection and classification, outlining the key features and limitations of each contribution.

In general, machine learning algorithms and DNNs are the preferred solutions for building privacy malware classifiers on existing datasets.

For example, a classifier that uses Adversarial Machine Learning (AML) in a Long Short Term Memory (LSTM) model with the Majestic top 1 million dataset and the Domain Generation Algorithm Archive (DGArchive) malware families repository is presented by Yilmaz et al. [22]. Although this contribution allows to reach a high accuracy metric, it is just limited to traditional patterns on domains seen in various families of malware and dictionary-based attacks.

A similar approach with structural variations has been proposed in order to improve the classifier's time of performance by applying a Bidirectional Long Short-Term Memory (BiLSTM) method to three different models [18]. The study is focused on analyzing behaviors of dynamic requests from the Windows API Calls Sequences dataset in order to detect and classify polymorphic malware.

The contribution presented in [23] is able to detect Advanced Persistent Threats (APT) that steals user credentials in just 2.7 minutes, using a Strange Behavior Inspection (SBI) machine learning model mainly oriented

TABLE 1. Related work comparison.

Reference	Data Gathering Method	Multiclass Problem	Classification Method	Dataset	Limitation(s)
Carrier et al. [3]	Memory dumping		Traditional machine learning algorithms	CIC-MalMem-2022	Use of ensemble classifiers (base learners and meta learners) in order to reach a higher accuracy metric
Dener et al. [8]	Memory dumping		Traditional machine learning algorithms and DNNs	CIC-MalMem-2022	Use of complex DNN architecture for tackling binary classification problem
Shafin et al. [15]	Memory dumping	✓	CNNs	CIC-MalMem-2022	Use of complex CNNs
Yang et al. [17]	DNS patterns	✓	HDNN	360netlab DGA	Limited to malicious DNS recognition
Girinoto et al. [18]	APIs patterns		BiLSTM	Windows API Calls Sequences	Limited to dynamic requests behaviors analysis
Vinayakumar et al. [19]	DGA patterns	✓	CNN-LSTM	AmritaDGA	Limited to trojan horses that run DGAs for hosting obfuscated privacy malware
Mezina and Burget [21]	Memory dumping	✓	CNNs	CIC-MalMem-2022	Use of complex CNNs
Yilmaz et al. [22]	DNS patterns	✓	AML with LSTM	Majestic top 1 million DGArchive	Limited to malware specialized on dictionary-based attacks
Mohamed and Belaton [23]	APT patterns	✓	Proposed SBI	SBI dataset	Limited to APT attacks
Mishra et al. [24]	Hardware patterns	✓	Random Forest (RF)	University of Mexico (UNM) and BareCloud datasets	Focus on obfuscated privacy malware families specialized on attacking virtual domains
Jones and Wan [25]	Entropy analysis	✓	Traditional machine learning algorithms	RWE	Malgazer classifier does not include to spyware. Assumes that the samples presented to Malgazer are already proven to be malware
Yang and Guo [26]	PCAP analysis	✓	Self-DNN	PCAP dataset	Limited to ransomware cyphering malware against privacy and availability
Zhang et al. [27]	Memory dumping		Operative system and firmware validation	No apply	Tackles availability but the idea is extensible to privacy
Mani et al. [28]	Powershell analysis	✓	DNN-LSTM	Windows Performance Counters dataset	Limited to cryptomining malware able to steal personal data. Tackles just special types of ransomware
Kumar et al. [29]	Memory dumping		Validation of memory consistency	No apply	Binary detector based on finding out or not a memory inconsistency when memory is analyzed
Apello et al. [30]	Operative system data		Data analysis through parallel computing and multi-core architectures	No apply	Performance not comparable to that of solutions based on machine learning. Limited binary classification through complex analysis of memory using parallel computing techniques
Vinayakumar et al. [31]	Pishing patterns	✓	CNN-LSTM	Spamdataset2	Limited to just certain types of obfuscated privacy malware spread through URLs, email and spamming
This research	Memory dumping	✓	Traditional machine learning algorithms and DNNs	CIC-MalMem-2022	We propose a novel solution to address the aforementioned limitations in literature

to detect long-term spyware techniques. The SBI database comprises live APT attacks observations directed to three victims.

A Heterogeneous Deep Neuronal Network (HDNN) proposed by Yang et al. [17] recognizes malicious DNS found in volatile memory that lead to malware capable of executing botnets and trojan horses to perpetrate Distributed Denial of Services (DDoS) attacks and stealthy exploits, respectively. This contribution is similar to the one presented by Yilmaz et al. [22], uses the 360netlab DGA dataset and is limited to only DNS name recognition generated by malware, but achieves high accuracy rates for these two types of threats.

Similar work has been done by Mishra et al. [24], although with an approach based on traditional machine learning algorithms applied to a cloud-based services environment, building a system capable of detecting and classifying stealthy malware attacks using a dataset built from memory introspection techniques.

The research carried out by Jones and Wan [25] achieves excellent results using the Running Window Entropy (RWE) analysis technique. However, the Malgazer dataset does not include to spyware (the main malware category against privacy) from the six malware classification groups analyzed.

In the same way, the contribution of Yang and Guo [26] presents a self-DNN solution capable of extracting features

and patterns, but mainly from traffic seen through packet capture (PCAP) files, that could be used as a baseline for a future memory dumping analysis approach.

Although not with a focus on data privacy but on the availability of information, the work presented in [27] demonstrates the good results that can be obtained applying deep learning techniques to analyze data from memory dumping. Therefore, the idea could be extended to other scenarios.

Reference [28] provides an example of this fact, which is limited to detecting and classifying cryptomining malware and special types of ransomware that could be applied in a privacy context by considering the features of the command and control sequences that can achieve certain families of ransomware.

It is also important to mention that there are relevant contributions that are not just based on machine learning techniques. For example, in [29] a solution based on chip multiprocessors is analyzed to validate memory consistency and detect malware if an inconsistency is discovered.

In the same way, the work carried out by Appello et al. [30] presents a solution that, although it demands the use of parallel computing and multi-core architectures, is capable of analyzing a memory dump that feeds a Value Change Dump (VCD) analysis tool to find out malware patterns at runtime. However, the performance achieved does not seem to be comparable to that of contributions using machine learning and deep learning techniques.

The CIC-MalMem-2022 dataset has previously been examined for binary classification challenges using both traditional machine learning algorithms and DNNs, as demonstrated by Carrier et al. [3] and Dener et al. [8]. Given the comprehensive analyses conducted by these authors, showcasing the dataset's efficacy in achieving acceptable metrics with traditional machine learning algorithms, we opted not to replicate this prior work in our research. Nevertheless, the study presented by Carrier et al. [3] highlights the effectiveness of logistic regression when it works together with those algorithms in ensemble classifiers, achieving higher accuracy metrics, although still less than DNNs. This insight led us to focus our research on proposing a novel logistic regression model capable of attaining comparable metrics to DNNs without the use of ensemble classifiers.

The multiclass problem in order to detect and distinguish among obfuscated privacy malware categories and families is not carried out in [3] nor [8]. However, the authors emphasize the fact that the CIC-MalMem-2022 dataset's observations comprise an analysis of in-memory obfuscated malware that is extremely complicated to analyze and classify into various categories with a standalone classifier, which is why the authors once again suggest using ensemble classifiers.

Therefore, it can be deduced from these contributions that any solution built to address the multiclass classification problem using ensemble classifiers (for instance a strategy based on taking the output of binary classification for subsequent malware classification if a malicious sample is detected using a boosting strategy) will achieve higher metrics values than a standalone multiclass classifier. In this sense, our research has focused on tackling the multiclass classification problem, including the benign class in standalone multiclass classifiers, which represents a challenge considering the imbalanced data between the benign class and each malware category or family.

The multiclass classification problem using the CIC-MalMem-2022 is analyzed mainly by Shafin et al. [15] and Mezina and Burget [21], but all the solutions proposed in these contributions are based on complex CNNs architectures and in [15] the study is oriented to smart city applications instead of privacy.

Although CNNs promise to reach high metrics values in comparison to other deep learning architectures, pooling layers involve data loss and the neural network requires a large amount of samples for training [32], which is a constraint in obfuscated privacy malware classification.

III. METHODOLOGY

In this section is detailed the CIC-MalMem-2022 dataset (Subsection III-A) and the methods used to carry out the experiments (Subsection III-B). In the same way, the performance metrics applied for building and evaluating the different obfuscated privacy malware classifiers are explained in Subsection III-C, as well as the materials and tools used that are mentioned in Subsection III-D.

Fig. 2 illustrates an overview of the methodology followed in this research, comprising three main stages. Initially, data acquisition, cleaning and preprocessing set the foundation for constructing obfuscated privacy malware classifiers in the second stage. This second step included implementing a logistic regression-based binary classifier (classifier 1) and developing multiclass classifiers (classifiers 2 and 3) using both traditional machine learning algorithms and deep learning techniques. Lastly, the gathered results underwent comprehensive statistical analysis, forming the basis for an in-depth discussion, final research conclusion, and the reporting of achievements. This systematic approach ensured a coherent flow from data handling to model creation, culminating in a robust analysis and meaningful research outcomes.

A. DATASET

The dataset used in this work was the Malware Memory Analysis CIC-MalMem-2022 created by the Canadian Institute for Cybersecurity of the University of New Brunswick (CIC-UNB). Released in April 2022, this dataset is available for download at [20] and a complete dataset description is presented in [3] and [8].

The dataset contains a total of 58,596 balanced observations with 29,298 benign and 29,298 malicious samples. Each observation corresponds to a memory dump, that is, a set of values with certain features that describe the patterns and behaviors of the obfuscated privacy malware once executed.

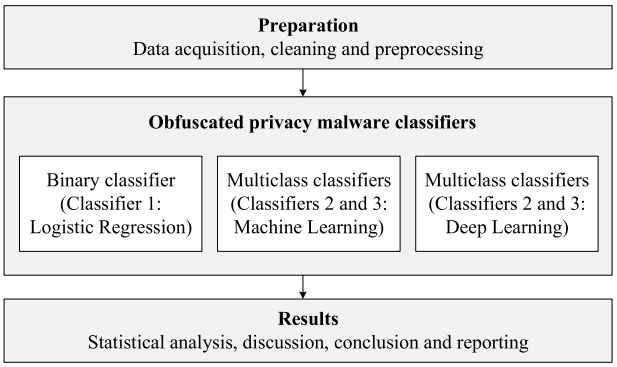

FIGURE 2. Methodology overview.

TABLE 2. CIC-MalMem-2022 obfuscated privacy malware classification.

Malware Category	Malware Family	Observations
Spyware	180Solutions	2 000
	Coolwebsearch	2,000
	Gator	2,200
	Transponder	2,410
	TIBS	1,410
Ransomware	Conti	1,988
	Maze	1,958
	Pysa	1,717
	Ako	2,000
	Shade	2,128
Trojan Horse	Zeus	1,950
	Emotet	1,967
	Refroso	2,000
	Scar	2,000
	Reconyc	1,570
Total		**29,298**

Having balanced benign and malicious observations makes the CIC-MalMem-2022 suitable for binary classification.

Each malicious observation belongs to one of the three categories of obfuscated privacy malware (spyware, ransomware, or trojan horse) distributed in the following way: 10,020 observations belong to spyware, 9,791 observations are ransomware samples, and 9,487 observations belong to trojan horse malware.

The dataset also discriminates among obfuscated privacy malware families for each malware category, as detailed in Table 2 which is exposed in [8, Table 3]. The malware families taken into account by the dataset are those that exclusively threaten the privacy of users.

In total, the dataset comprises 55 features plus two labels as explained in Table 3 which has been taken from [8, Table 2] and is presented here for convenience.

The label Category of the dataset describes whether the observation is benign or, in case of malware, the family to which the observation belongs, thus allowing to distinguish at the same time if the observation belongs to a spyware, ransomware, or trojan horse category. The last label (Class) is redundant.

B. METHODS

This research followed a methodology in stages, as described in Fig. 3. Each stage achieved results that were used to structure the final research conclusion.

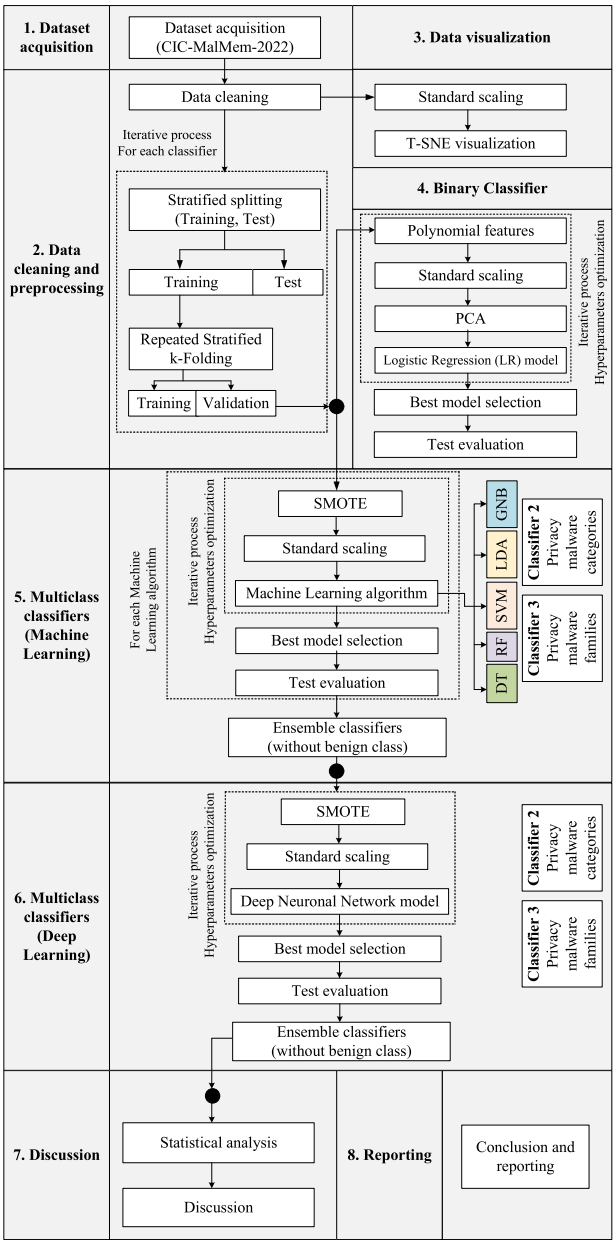

FIGURE 3. Detailed methodology.

In the first stage, the search and acquisition of the CIC-MalMem-2022 dataset that significantly covers the patterns and behaviors of the three types of obfuscated privacy malware were carried out.

In the second stage, data cleaning and splitting were performed on the acquired dataset.

Data cleaning allowed identifying and distinguish between the dataset observations corresponding to benign and malicious memory dumps (classifier 1), as well as between the malicious observations which correspond to the three categories of obfuscated privacy malware (classifier 2) and within each category to which specific family it belongs (classifier 3), achieving a dataset with three more fields (labels) for malware classification.

TABLE 3. CIC-MalMem-2022 features and labels.

Number	Field	Description
1	Category	Label for obfuscated privacy malware categories and families according to Table 2
2	pslist.nproc	Total number of processes
3	pslist.nppid	Average number of threads for the processes Total
4	pslist.avg_threads	Average number of 64 bit processes
5	pslist.nprocs64bit	Total number of 64 bit processes
6	pslist.avg_handlers	Average number of handlers
7	dllist.ndlls	Total number of loaded libraries for every process
8	dllist.avg_dlls_per_proc	Average number of loaded libraries per process
9	handles.nhandles	Total number of opened handles
10	handles.avg_handles_per_proc	Average number of handles per process
11	handles.nport	Total number of port handles
12	handles.nfile	Total number of file handles
13	handles.nevent	Total number of event handles
14	handles.ndesktop	Total number of desktop handles
15	handles.nkey	Total number of key handles
16	handles.nthread	Total number of thread handles
17	handles.ndirectory	Total number of directory handles
18	handles.nsemaphore	Total number of semaphore handles
19	handles.ntimer	Total number of timer handles
20	handles.nsection	Total number of section handles
21	handles.nmutant	Total number of mutant handles
22	ldrmodules.not_in_load	Total number of modules missing from the load list
23	ldrmodules.not_in_init	Total number of modules missing from the init list
24	ldrmodules.not_in_mem	Total number of modules missing from the memory list
25	ldrmodules.not_in_load_avg	Average amount of modules missing from the load list
26	ldrmodules.not_in_init_avg	Average amount of modules missing from the init list
27	ldrmodules.not_in_mem_avg	Average amount of modules missing from the memory
28	malfind.ninjections	Total number of hidden code injections
29	malfind.commitCharge	Total number of Commit Charges
30	malfind.protection	Total number of protection
31	malfind.uniqueInjection	Total number of unique injections
32	psxview.not_in_pslist	Total number of processes not found in the pslist
33	psxview.not_in_eprocess_pool	Total number of processes not found in the psscan
34	psxview.not_in_ethread_pool	Total number of processes not found in the thrdproc
35	psxview.not_in_pspcid_list	Total number of processes not found in the pspcid
36	psxview.not_in_csrss_handles	Total number of processes not found in the csrss
37	psxview. not_in_session	Total number of processes not found in the session
38	psxview. not_in_deskthrd	Total number of processes not found in the desktrd
39	psxview.not_in_pslist_false_avg	Average false ratio of the process list
40	psxview.not_in_eprocess_pool_false_avg	Average false ratio of the process scan
41	psxview.not_in_ethread_pool_false_avg	Average false ratio of the third process
42	psxview.not_in_pspcid_list_false_avg	Average false ratio of the process id
43	psxview.not_in_csrss_handles_false_avg	Average false ratio of the csrss
44	psxview.not_in_session_false_avg	Average false ratio of the session
45	psxview.not_in_deskthrd_false_avg	Average false ratio of the deskthrd
46	modules.nmodules	Total number of modules
47	svcscan.nservices	Total number of services
48	svcscan.kernel_drivers	Total number of kernel drivers
49	svcscan.fs_drivers	Total number of file system drivers
50	svcscan.process_services	Total number of Windows 32 owned processes
51	svcscan.shared_process_services	Total number of Windows 32 shared processes
52	svcscan.interactive_process_services	Total number of interactive service processes
53	svcscan.nactive	Total number of actively running service processes
54	callbacks.ncallbacks	Total number of callbacks
55	callbacks.nanonymous	Total number of unknown processes
56	callbacks.ngeneric	Total number of generic processes
57	Class	Label to distinguish between benign and malware samples

For each classifier, the dataset was then split taking 80% of the total data for training and cross-validation and the rest 20% for test. This data splitting was done in a stratified way among the categories handled by each classifier. On the other hand, a repeated k-fold stratified cross-validation strategy was carried out on the 80% of training data using 2 repetitions and 5 folds (10 iterations in total).

The t-Distributed Stochastic Neighbor Embedding (t-SNE) technique was used in stage 3 in order to visualize data, allowing to obtain a clear idea of the complexity of the

Algorithm 1 Binary classifier pseudocode.

```
Data: CIC-MalMem-2022 dataset
Result: Test resulting metrics for best logistic
        regression model
Execute data cleaning
Execute data splitting
models ← get all possible logistic regression's models
bestModel ← null
bestMetric ← 0
for each model do
    Train model with the model's hyperparameters
    Obtain mean training metrics using k-fold
      cross-validation
    if Mean Recall metric > bestMetric then
        bestModel ← model
        bestMetric ← Mean Recall metric
    end
end
Retrain best model with all training data
Evaluate model with test data
Tabulate achieved results
```

problem for each classifier. t-SNE was used with a perplexity value of 100 for all classifiers.

In stage four, a deep grid search was performed to optimize the hyperparameters of the logistic regression binary classifier. The hyperparameters searched were the polynomial degree (up to degree two), the regularization type (L1 and L2), and the regularization coefficient. This resulted in the validation of 20 different models.

For each model, the training data for the respective fold was standardized and dimensionally reduced to two axes using Principal Component Analysis (PCA). The logistic regression models were trained using the saga solver with a maximum of 200 iterations, which was enough to achieve high metrics.

Then, the best logistic regression model achieved from the hyperameter optimization process was evaluated with the test data.

Algorithm 1 details the pseudocode outlining the steps taken to construct the binary classifier. The final outcome of this process comprises the metrics obtained using the test dataset.

In stage five, SMOTE was used to address imbalanced data in multiclass obfuscated privacy malware classifiers; and five different machine learning algorithms were used for training. These classifiers considered the benign class and the three malware categories for classifier 2, as well as the fifteen malware families for classifier 3.

The machine learning algorithms used for both multiclass classifiers were:

- **Decision Tree (DT):** A decision tree with a maximum depth size equal to 3 was taken as a reference for classifier 2 and a value equal to 5 for classifier 3. Gini and entropy criterion were taken as hyperparameters for optimizing both classifiers.

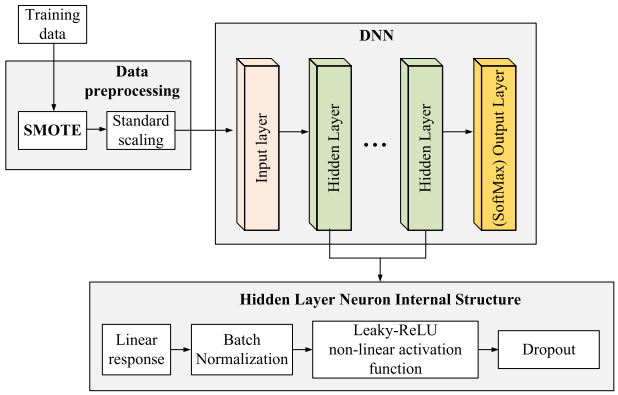

FIGURE 4. DNN architecture.

- **Random Forest (RF):** A random forest with the number of trees and a comparison between the Gini and the entropy criterion were taken as hyperparameters for optimization. The number of trees ranged from 100 to 200 in steps of 10 while the maximum depth was taken with a value equal to 3 for classifier 2 and a value equal to 5 for classifier 3.
- **Support Vector Machine (SVM) Classifier:** For this classifier a value of C equal to 10 was taken for classifier 2 and a value of C equal to 1 was considered for classifier 3. The type of kernel between linear and rbf was taken as the hyperparameter optimization criterion.
- **Linear Discriminant Analysis (LDA):** No hyperparameters were considered for this machine learning algorithm.
- **Gaussian Naive Bayes (GNB):** Similar to the previous case, no hyperparameters were needed for this algorithm.

The hyperparameters optimization process for each machine learning algorithm and multiclass classifier was similar to the process for the binary classifier.

The best models achieved for each machine learning algorithm for classifiers 2 and 3 were taken to evaluate them with the test data and gather the results.

In order to demonstrate the validity of our results, we also built ensemble classifiers in this stage. Each ensemble classifier comprised the previously achieved binary logistic regression classifier, in sequence with a specific multiclass classifier trained using the same method but excluding the benign class. This architecture is helpful for boosting the metric achieved by the ensemble classifier due to the action of the strong binary classifier [3]. The experiment encompassed all the previously mentioned traditional machine learning algorithms, enabling us to conduct a comprehensive comparison between the results obtained by ensemble classifiers and those achieved by the standalone multiclass classifiers.

In stage six, a general DNN for both multiclass classifiers was built. Fig. 4 illustrates the proposed DNN architecture.

The number of hidden layers and the value of the regularization coefficient were taken as criteria for hyperparameters optimization. In total, 9 different models were validated and

the best model achieved by each multiclass classifier was evaluated with the test data.

At each hidden layer, the Leaky-ReLU non-linear activation function was used with a factor of 0.1. This allowed to avoid the problem of dead neurons during the training phase. Also, after the linear response of each neuron and before its Leaky-ReLU non-linear activation function, a Batch Normalization step was added, which allowed us to accelerate the learning process. In the same way, after the Leaky-ReLU non-linear activation function a Dropout step with a factor equal to 0.01 was added to help avoid overfitting and memorization of the neural network.

L2 regularization was also used to help avoid overfitting. In the output layer, the softmax activation function was used, which allowed to obtain a vector of probabilities for each category analyzed. Regarding the solver, the Adam optimizer was used for these models because it showed, in general, better performance than other optimizers such as SGD, Adagrad, Adadelta, AdamW, among others [15]. The sparse categorical cross-entropy was taken as a loss criterion for analyzing each step.

Each hidden layer was built with 2,048 neurons and a total of 10 epochs with a mini-batch size of 32 were used for training. A learning rate equal to 1×10^{-3} was used because this rate was shown to decrease loss during the experiments.

Algorithm 2 outlines the overarching pseudocode employed in constructing multiclasses classifiers, leveraging both traditional machine learning algorithms and DNNs. This process served both for the classification of categories (classifier 2) and for the classification of families (classifier 3). The ultimate outcome comprises the metrics obtained with the test data.

As in the previous stage of our methodology, the corresponding ensemble classifiers were constructed as the sequence of the binary classifier followed by the multiclass classifier trained with the same method but without considering the benign class following a boosting strategy.

Table 4 provides an insightful overview of the key design criteria underpinning our research for both binary and multiclass classifiers. Each criterion has been meticulously chosen to align with and optimize the proposed architectures of our models. The table elucidates the specific reasons why these criteria are instrumental in enhancing the effectiveness and performance of our classifiers.

In stage seven, the unidirectional Wilcoxon rank sum test was used at a significant level of 0.01 to perform a statistical analysis between the metrics achieved by the traditional machine learning algorithms and the proposed DNN for each multiclass classifier.

For this, the classifier that achieved the best cross-validation metric among the different evaluated folds was taken as a pivot to compare it with the remaining classifiers. The calculated p-value allowed us to discern whether the observed difference was statistically significant. We also conducted a performance comparison on the test data for each multiclass classifier, utilizing both traditional machine learning algorithms and our proposed DNN.

Algorithm 2 Multiclass classifiers pseudocode.

```
Data: CIC-MalMem-2022 dataset
Result: Test resulting metrics for each best machine
        learning (ML) and DNN model
Execute data cleaning
Execute data splitting
for each ML and DNN algorithm do
    models ← get all possible algorithm's models
    bestModel ← null
    bestMetric ← 0
    for each model do
        Train model with the model's hyperparameters
        Obtain mean training metrics using k-fold
         cross validation
        if Mean Accuracy metric > bestMetric then
            bestModel ← model
            bestMetric ← Mean Accuracy metric
        end
    end
    Retrain best model with all training data
    Evaluate model with test data
    Tabulate achieved results
end
```

The final stage of our methodology involved defining the research conclusion and reporting the results, making possible the reproducibility of our experiments.

C. PERFORMANCE METRICS

The mean recall was used as the metric for hyperparameter optimization for the binary case in order to achieve a classifier capable of reducing false negative rates. The malicious category was taken as positive class to calculate the metrics.

Precision, recall, F1-score, accuracy, and the Area Under the Curve (AUC) were used to evaluate the best binary classifier with the test data.

A complete guide to calculate these metrics from an information security perspective can be found in [36].

On the other hand, the mean accuracy was taken as a reference for hyperparameter optimization for both multiclass classifiers.

The best multiclass classifiers achieved were evaluated with the test data using the precision, recall, F1-score, accuracy and AUC metrics. The overview provided by [37] explains in depth how these metrics can be calculated for multiclass scenarios.

These same metrics for cross-validation and evaluation at test were applied to the ensemble classifiers. However, the ultimate test metric achieved by the ensemble classifier was derived as a compound value by assigning equal weight to both the binary classifier metric and the multiclass classifier

TABLE 4. Design criteria explanation.

Classifier	Design criteria	Explanation
Binary	Logistic regression	Suitable baseline machine learning algorithm. Unlike [8] we do not use ensemble classifiers to achieve a high accuracy metric.
	PCA	This technique allows to reduce the required model's training time considering the main components of data, with optimal results for binary classification problems [33].
Multiclass	DNN	Unlike the architectures exposed in [15] and [21], DNNs comprise less complex models for training without the disadvantages exposed by [32] in general malware classification problem.
	SMOTE	Data augmentation technique that allows to tackle the imbalance problem through the generation of synthetic data samples [3].
	Batch Normalization	Used properly it allows for improvement of initial standard scaling of data [18], [34].
	Leaky-ReaLU	Helps to reduce dead neurons problem [18], [35].
	Dropout	Helps to avoid the learning of statistical noise at each epoch [18].
	Include benign class	As [25] demonstrates, excluding the benign class usually allows to reach higher metrics through ensemble classifiers comprised by a strong binary classifier in sequence with classifiers focused exclusively on malware. Thus, the challenge of this research is to achieve a high metric using standalone classifiers.

metric, as Equation 1 suggests.

$$ECM = 0.5 \times LRM + 0.5 \times MCM, \tag{1}$$

where *ECM* (Ensemble Classifier Metric) denotes the final ensemble classifier metric achieved, *LRM* (Logistic Regression Metric) denotes the metric reached by the binary logistic regression classifier, and *MCM* (Multiclass Metric) refers to the metric achieved by the multiclass classifier without considering the benign class. Even though interpreting the final result of an ensemble classifier is a difficult task due to the complexity of defining the contribution and importance of each component, this compound metric can be considered fair enough for the purpose of comparing results [38].

D. MATERIALS AND TOOLS

This research was carried out using the Google Colaboratory platform (Google Colab) and the Python programming language. The Google Colab environment provided up to 12.7 GB of RAM memory and 78.2 GB of disk capacity with NVIDIA T4 GPU processing.

The Sci-kit Learn: Machine learning in Python framework (best known as sklearn) [39] was used to clean and preprocess data, as well as to build the different machine learning models and evaluate them.

Google TensorFlow and its API Keras [40] were used to build DNNs, which were embedded in sklearn pipelines using the technique described in [41].

Furthermore, the DNNs of this research were implemented using the Keras Wrapper solution for Python described in [42], which allowed the models achieved with TensorFlow to be adapted and integrated into data structures supported by sklearn.

The statistical analysis of this paper was performed using the software R.[1]

IV. RESULTS AND ANALYSIS

In this section we present and analyze the results obtained with the proposed methodology. In Subsection IV-A the visualization pictures of t-SNE data are displayed, whereas in Subsections IV-B, IV-C and IV-D are exposed the results achieved for the binary case, the obfuscated privacy malware category classifier and the obfuscated privacy malware family classifier, respectively.

A. T-SNE DATA VISUALIZATION

The visualization of t-SNE data with dimension reduction to two components and a perplexity value of 100 demonstrates that the CIC-MalMem-2022 dataset is capable of clearly distinguishing between benign and malicious observations, as Fig. 5a suggests.

However, when trying to discriminate between different categories of malware, it is not feasible to find such a clear border as in the binary case, as illustrated in Fig. 5b.

This demonstrates the difficulty that currently exists in distinguishing between different categories of obfuscated privacy malware. As suggested by [43], many ransomware families have patterns similar to trojan horses and several behaviors that resemble spyware attacks.

Similarly, Fig. 5c suggests that the classification of different malware families is even a more difficult task due to the fact that each family shares common properties with others, more likely if they are of the same malware category.

To effectively tackle obfuscated privacy malware detection and classification at a granular level, employing a diverse set of features aligned with the problem's complexity is crucial. The number of features required increases with the growing number of classes, contributing to reaching enhanced metrics. Hence, applying dimensionality reduction techniques is not advisable for the multiclass problem. Conversely, for binary classification, optimizing feature space by reducing dimensionality is an insightful strategy. This is attributed to the limited number of classes, where common and overlapping patterns and behaviors predominantly exist within malware categories and families.

[1]The CIC-MalMem-2022 dataset and all the Python and R files for experiments reproducibility are available at https://github.com/dcevallossalas/PrivacyMalwareClassifiers

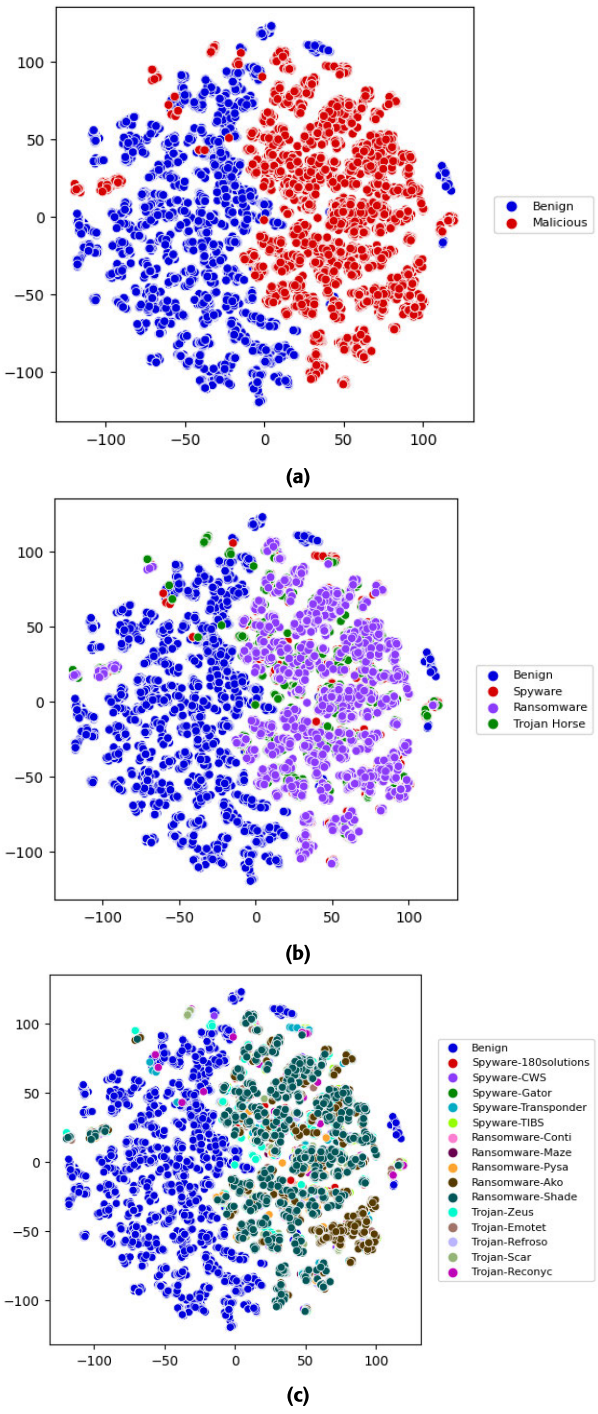

FIGURE 5. t-SNE data visualization (a) Binary case (b) Obfuscated privacy malware categories (c) Obfuscated privacy malware families.

B. OBFUSCATED PRIVACY MALWARE BINARY CLASSIFIER

The hyperparameter values that allowed us to obtain the best logistic regression model among the 20 possible binary classifiers analyzed were a regularization parameter equal to 1×10^{-3} with L2 regularization and a polynomial degree equal to 1.

Therefore, the results suggest that it is not necessary to generate polynomial features to obtain a high metric for binary classification with the CIC-MalMem-2022.

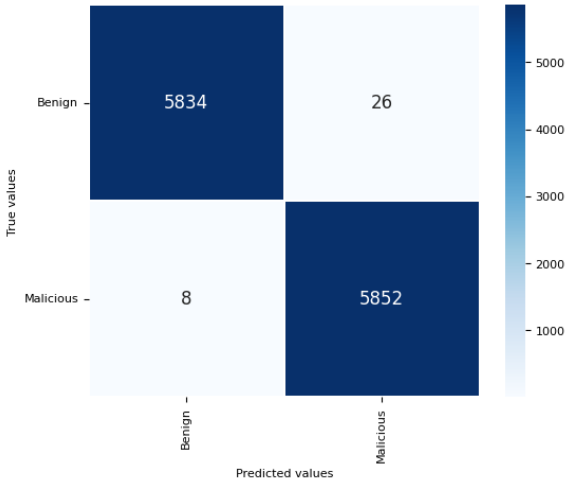

FIGURE 6. Binary classifier's test confusion matrix.

These hyperparameters allowed us to reach a mean recall cross-validation score of 0.997 ± 0.002.

The metrics achieved with the test data were a precision of 0.996, recall of 0.999, F1-score of 0.997, accuracy of 0.997 and AUC of 0.997.

Fig. 6 presents the resulting confusion matrix for the binary classifier's evaluation with the test data. As the figure suggests, 26 observations were classified as false positive whereas just 8 observations were classified as false negative cases, which demonstrates that the binary classifier built is able to reduce the false negative cases in order to achieve a high recall metric.

The achievement of high metrics is attributed to the PCA technique and the balance maintained in the dataset between benign and malicious observations which facilitates the classification problem considering both classes on equal terms.

The results achieved are comparable to those reached by [3] using ensemble classifiers and to those achieved by [8] using DNNs. However, our model comprises a standalone classifier based solely on logistic regression without requiring the computational processing of DNNs.

C. OBFUSCATED PRIVACY MALWARE CATEGORY CLASSIFIERS

The best models for the standalone category classifiers were achieved with the following hyperparameters:

- For the DT algorithm, a maximum depth of 3 with the Gini criterion.
- For the RF algorithm, 120 trees with a maximum depth of 3 and the Gini criterion were used.
- For the SVM classifier, a value of C equal to 10 and the rbf kernel achieved the best metrics.
- For DNN, the best hyperparameter values were a number of hidden layers equal to 3 and a value of regularization coefficient equal to 1×10^{-6}.

Table 5 details the values of the mean accuracy metrics achieved by each best model of the machine learning

TABLE 5. Best models comparison for standalone category classifiers.

Classifier	Mean accuracy	p-value (α = 0.01)
DT	0.713±0.003	p<0.0001
RF	0.747±0.009	p<0.001
SVM	0.693±0.008	p<0.0001
LDA	0.730±0.003	p<0.0001
GNB	0.685±0.009	p<0.0001
DNN	**0.769±0.012**	**pivot**

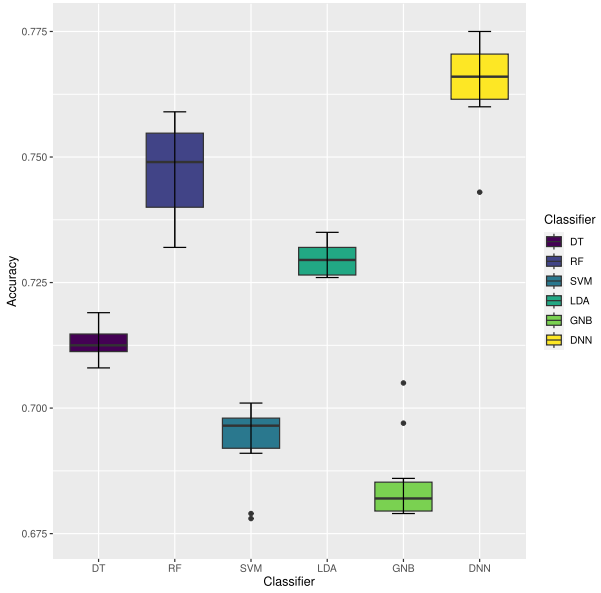

FIGURE 7. Box plots for best models of standalone category classifiers.

algorithm and deep learning. The best classifier obtained was the proposed deep learning model, reaching a mean accuracy value of 0.769 ± 0.012.

The obtained results demonstrate that our proposed DNN architecture leverages data preprocessing and SMOTE techniques, strategically incorporating deep learning concepts like batch normalization, leaky-ReLU, and dropout layers. This surpasses traditional machine learning algorithms that lack these principles by default in their architecture.

In Fig. 7 are detailed the box plots for the best models of standalone category classifiers, which corroborate that the best mean accuracy was reached by the DNN followed by the RF classifier.

Taking the DNN mean accuracy value as pivot and a significance level of 0.01, the unidirectional Wilcoxon rank sum test through the obtained p-values demonstrates that the differences observed between the pivot with the rest of algorithms are statistically significant.

Table 6 presents the metrics achieved with the test data for both standalone multiclass category classifiers and the ensemble classifiers built on top of the logistic regression binary classifier. The best results in test are for both classification strategies achieved by DNNs, demonstrating the capability of our proposed architecture even without the use of the benign class, in which the imbalance between

TABLE 6. Test resulting metrics for obfuscated privacy malware category classifiers.

Standalone Classifiers					
Classifier	**Precision**	**Recall**	**F1-score**	**Accuracy**	**AUC**
DT	0.760	0.711	0.677	0.712	0.874
RF	0.767	0.748	0.741	0.748	0.906
SVM	0.678	0.671	0.624	0.671	0.830
LDA	0.733	0.724	0.723	0.724	0.895
GNB	0.737	0.686	0.642	0.686	0.876
DNN	**0.767**	**0.761**	**0.754**	**0.761**	**0.923**
Ensemble Classifiers					
Classifier	**Precision ECM**	**Recall ECM**	**F1-score ECM**	**Accuracy ECM**	**AUC ECM**
LR+DT	0.772	0.741	0.717	0.740	0.817
LR+RF	0.770	0.759	0.753	0.758	0.850
LR+SVM	0.714	0.699	0.671	0.698	0.707
LR+LDA	0.741	0.733	0.731	0.732	0.826
LR+GNB	0.735	0.705	0.662	0.704	0.793
LR+DNN	**0.773**	**0.769**	**0.767**	**0.768**	**0.875**

the malware categories is not so pronounced and therefore the SMOTE technique is not taken full advantage of. Furthermore, it should be taken into account that half of the metric of each ensemble classifier is defined by the classification of the strong binary classifier, an advantage that the standalone classifiers do not have. Even so, we believe it is reasonable and fair to make a comparison in order to demonstrate the validity of the results achieved in this research.

Fig. 8 shows a comparison of the metrics achieved among the standalone category classifiers proposed in this research. For all the considered metrics, the higher values are achieved by our proposed DNN surpassing even the RF algorithm popularly known for its great classification capacity in multiclass problems [44].

Fig. 9 presents a comparison of the standalone DNN classifier and the ensemble LR+DNN classifier, demonstrating the feasibility of obtaining good results for the multiclass classification problem using our proposed standalone classifier. The metrics achieved with test data by the standalone DNN classifier are comparable and, for practical purposes, the same as those achieved by the ensemble LR+DNN classifier. Even the standalone classifier achieves a higher AUC value because by integrating the benign class it takes better advantage of the SMOTE technique.

In order to demonstrate the effect and importance of the SMOTE technique, Figs. 10a and 10b expose the Receiver Operating Characteristic (ROC) curves achieved for the standalone category classifier without and with SMOTE in the test dataset, respectively.

Overall, the AUC metric for each obfuscated privacy malware category increases when SMOTE is applied mainly for the ransomware and trojan horse categories. The AUC reached for the benign category classification is excellent for practical purposes.

Although this classifier just considers four categories, the effect of SMOTE can be clearly seen due to the ROC curves

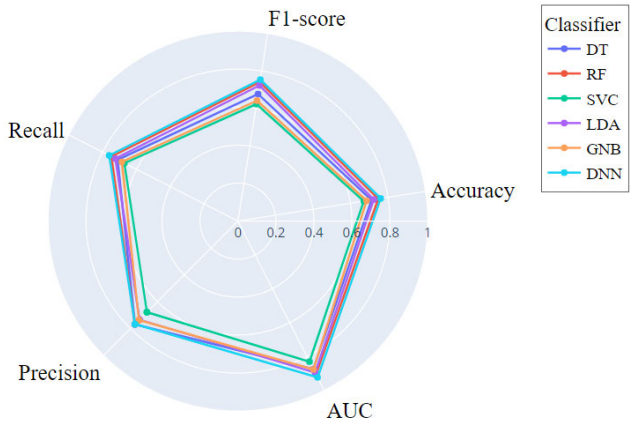

FIGURE 8. Test metrics comparison for standalone category classifiers.

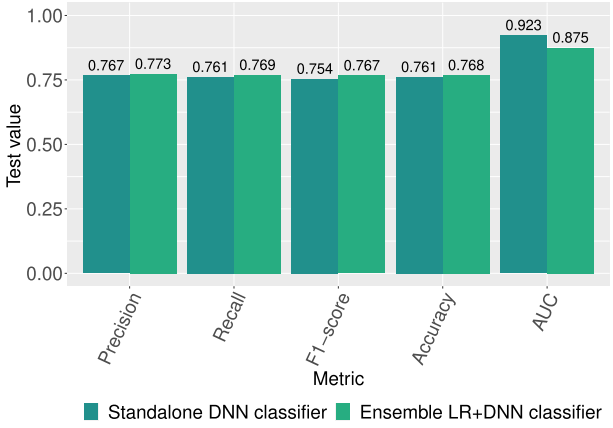

FIGURE 9. DNN test metrics comparison for standalone category classifier and ensemble classifier.

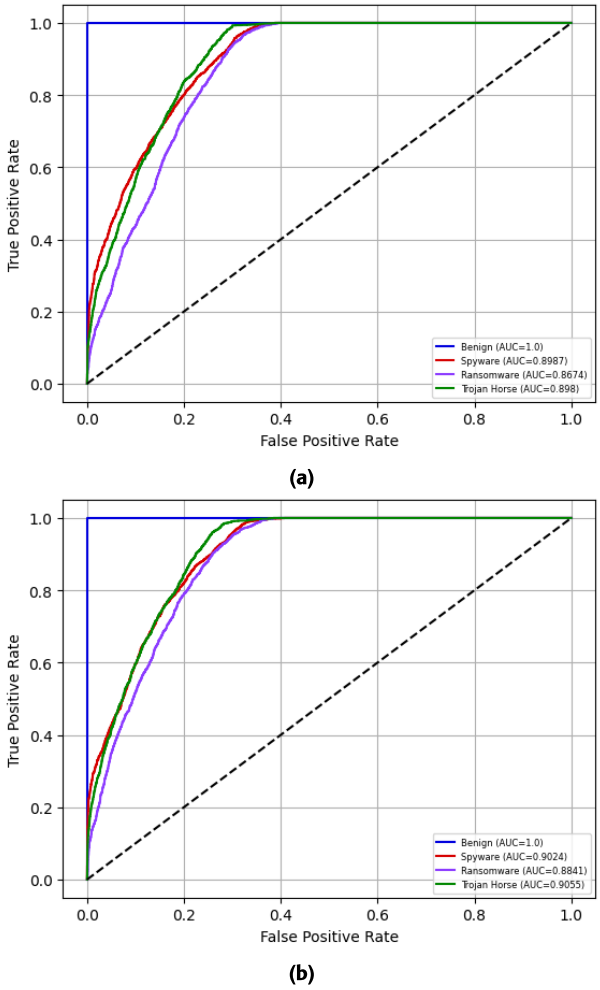

FIGURE 10. Standalone DNN category classifier's ROC curves (a) Without SMOTE (b) With SMOTE.

cover a larger area compared to the scenario in which SMOTE is not used.

D. OBFUSCATED PRIVACY MALWARE FAMILY CLASSIFIERS

In this section are presented the results achieved for the obfuscated privacy malware family classifiers. As in the previous section, this is a multiclass scenario and the best standalone models where achieved with the following hyperparameters:

- For DT algorithm a maximum depth of 5 with the entropy criterion.
- For RF algorithm 160 trees with a maximum depth of 5 and the entropy criterion were used.
- For the SVM classifier, a value of C equal to 1 and the linear kernel achieved the best metrics.
- As in the previous section, for the DNN the best hyperparameters values obtained were a number of hidden layers equal to 3 and a value of the regularization coefficient equal to 1×10^{-6}.

The mean accuracy metrics achieved by each machine learning algorithm and the proposed DNN are presented in Table 7. Our proposed DNN achieved the best mean accuracy with a value of 0.683 ± 0.030.

Given that the classification problem addressed by this multiclass classifier is much more granular and complex, the metrics achieved by the standalone DNN classifier are even more highly noticeable than those reached by the considered traditional machine learning algorithms due to it takes optimal advantage of the SMOTE technique.

Fig. 11 shows the box plots for the best models achieved by each one of the standalone family classifiers analyzed. The DNN reaches the higher mean accuracy followed by the RF classifier.

Taking the DNN mean accuracy value as pivot, the unidirectional Wilcoxon rank sum test at a significance level of 0.01 demonstrates that the differences observed between our proposed DNN and the rest of the algorithms analyzed are statistically significant.

In Table 8 the metrics achieved by each one of the best models with the test data for both standalone family classifiers and ensemble classifiers built using the logistic regression binary classifier are presented. For both types of classifiers, the best metric values were again obtained using the DNN solution.

TABLE 7. Best models comparison for standalone family classifiers.

Classifier	Mean accuracy	p-value (α = 0.01)
DT	0.591±0.002	p<0.0001
RF	0.619±0.003	p=0.0014
SVM	0.549±0.004	p<0.0001
LDA	0.590±0.003	p<0.0001
GNB	0.566±0.003	p<0.0001
DNN	**0.683±0.030**	**pivot**

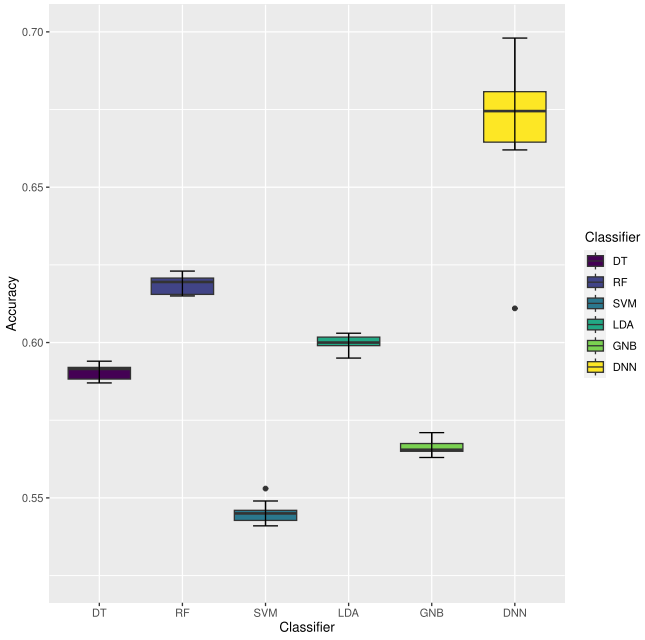

FIGURE 11. Box plots for best models of standalone family classifiers.

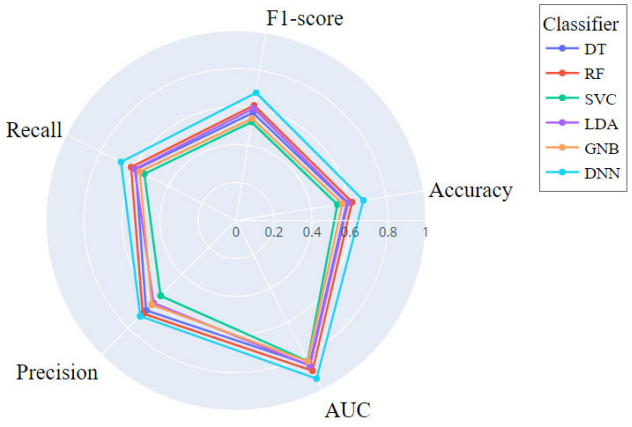

FIGURE 12. Test metrics comparison for standalone family classifiers.

A comparison between the metrics achieved by the standalone family classifiers with the test data is presented in Fig. 12. As can be seen, our proposed DNN architecture presents a better result compared to the rest of algorithms, and even more so compared to the previous multiclass category classifier, it shows that it is capable of achieving better metrics values than traditional machine learning algorithms as the complexity of the problem is greater.

TABLE 8. Test resulting metrics for obfuscated privacy malware family classifiers.

Standalone Classifiers					
Classifier	**Precision**	**Recall**	**F1-score**	**Accuracy**	**AUC**
DT	0.669	0.592	0.574	0.592	0.853
RF	0.693	0.619	0.614	0.619	0.888
SVM	0.560	0.541	0.525	0.541	0.830
LDA	0.614	0.598	0.598	0.598	0.859
GNB	0.627	0.568	0.544	0.568	0.833
DNN	**0.712**	**0.679**	**0.682**	**0.679**	**0.934**
Ensemble Classifiers					
Classifier	**Precision ECM**	**Recall ECM**	**F1-score ECM**	**Accuracy ECM**	**AUC ECM**
LR+DT	0.673	0.596	0.573	0.595	0.595
LR+RF	0.692	0.623	0.613	0.622	0.877
LR+SVM	0.600	0.546	0.536	0.545	0.649
LR+LDA	0.611	0.601	0.598	0.600	0.842
LR+GNB	0.620	0.573	0.548	0.572	0.820
LR+DNN	**0.682**	**0.661**	**0.656**	**0.660**	**0.910**

Fig. 13 presents a comparison between the metrics achieved for classifying obfuscated privacy malware families with the test data using both the standalone DNN classifier and the ensemble LR+DNN classifier. The metrics obtained by the standalone DNN classifier are comparable and even slightly superior to those reached by the ensemble classifier. This is because the benign class takes more advantage of the SMOTE technique, given that the obfuscated privacy malware families solely without the benign class are not strongly imbalanced. These results demonstrate the validity of our research by corroborating the ability of the proposed DNN architecture to address the multiclass classification problem, not only for obfuscated privacy malware categories but also for families.

Finally, Figs. 14a and 14b expose the ROC curves achieved by our DNN obfuscated privacy malware family classifier without and with SMOTE in the test dataset, respectively.

In contrast to the multiclass category classification problem of the previous subsection, the distinctions between the ROC curves are more pronounced, resulting in more arched and grouped curves within each family. This ultimately led to the attainment of a high AUC metric.

Unlike the contributions presented in [15] and [21], the results exposed for the category and family classification problem have been achieved without using advanced CNNs nor ensemble classifiers.

V. DISCUSSION

Obfuscated privacy malware detection and classification is a complex problem that depends to a large extent on the quality of the data to be solved.

The obfuscated nature of the memory dumps observations of the CIC-MalMem-2022 dataset allows to achieve high metrics for the binary classification problem, but limited values for the multiclass classification scenario.

With regard to the binary classification the contribution of this research has been to achieve through a logistic regression model metrics values comparable to those achieved with

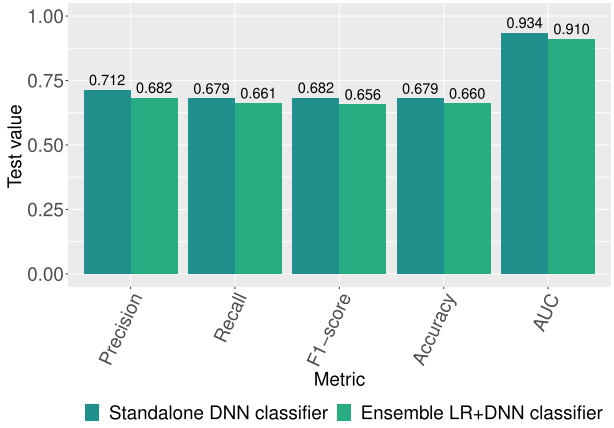

FIGURE 13. DNN test metrics comparison for standalone family classifier and ensemble classifier.

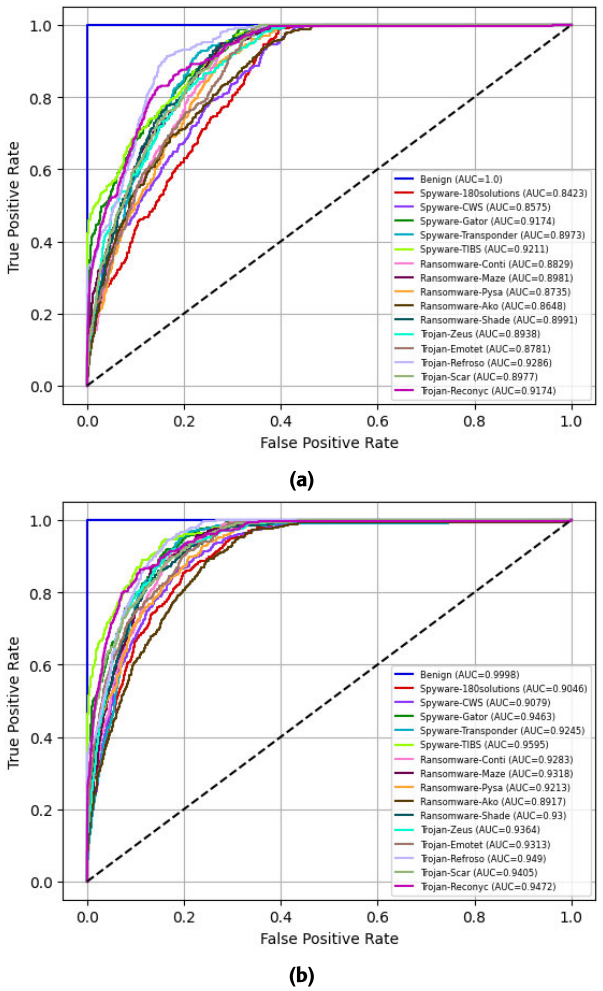

FIGURE 14. Standalone DNN family classifier's ROC curves (a) Without SMOTE (b) With SMOTE.

DNNs. For multiclass classification, the major contribution of this research has been to expose several preprocessing and data augmentation techniques, as well as a DNN architecture that can overcome the metrics reached by traditional machine learning algorithms when using this dataset and reach values comparable to those achieved using CNNs by other researchers. All our proposed models comprise standalone solutions and no ensemble classifiers that can be difficult to implement and interpret.

While prior studies have conducted experiments on diverse baseline datasets with different categories and families of obfuscated privacy malware, employing varied types of features, different numbers of observations, and varying methods of data division and evaluation across distinct applications, we would like to conduct and present a quite fair and reasonable comparison of the classifiers developed in this research with those presented in the major prior works with similar methodology and target, utilizing a common test metric as comparison criterion, and not just limited to solutions based on memory dumping analysis for data gathering. This metric encompasses accuracy for binary classification and F1-score for multiclass classification. The selection of F1-score for multiclass classification comparison is driven by the common imbalance scenario faced by most contributions.

However, it is important to highlight that a direct comparison with related works may not be strictly fair, considering the diversity of criteria, different contexts, and unique focuses that each research endeavor has in addressing the obfuscated privacy malware detection and classification problem. Table 9 summarizes the comparison of the results achieved in this research with other major contributions.

In the case of the binary classifier, the best test accuracy score of 0.970 achieved by Carrier et al. [3] using a RF classifier is lower than the 0.997 presented in this work, achieved through logistic regression and the PCA preprocessing technique. Our reached metric is comparable to the score of 0.990 achieved by the same authors when using ensemble classifiers for boosting several weak classifiers with a strong logistic regression model, which may be a practical limitation [25]. In the same way, logistic regression has also been applied by Dener et al. [8] reaching exactly the same accuracy score of 0.970 as [3] with the RF classifier.

Our logistic regression classifier comprises a lightweight model capable of achieving metrics values comparable to those achieved through a DNN (which reach practically a perfect accuracy score of 1 [8]), avoiding investing high computational resources in training complex DNN models or costly computational tasks.

For example, the overall accuracy value achieved in this research is greater than the one reached by Yang et al. [17] equal to 0.9773 without using parallel computing techniques, even when this last contribution uses another malware classification technique based on the DGA dataset with 500,000 observations.

We strongly encourage that our proposed logistic regression model be used with other datasets that address the problem of binary obfuscated privacy malware classification.

With regard to multiclass classification, the SMOTE technique used to address the imbalanced scenarios to

TABLE 9. Comparison of results with other obfuscated privacy malware classification contributions.

Classifier	Reference	Metric	Value	Classification Method	Dataset	Imbalance	Number of Observations	Number of Classes
Binary	Carrier et al. [3]	Accuracy	0.970	RF	CIC-MalMem-2022		58,596	2
			0.990	Ensemble LR				
	Dener et al. [8]	Accuracy	0.970	RF	CIC-MalMem-2022		58,596	2
			0.999	DNN				
	Yang et al. [17]	Accuracy	0.977	HDNN	360netlab DGA		500,000	2
	This research	Accuracy	0.997	LR	CIC-MalMem-2022		58,596	2
Categories	Mezina and Burget [21]	F1-score	0.750	CNN	CIC-MalMem-2022	✓	58,596	4
	Vinayakumar et al. [31]	F1-score	0.769	CNN-LSTM	Spamdataset2	✓	33,895	3
	This research	F1-score	0.754	SMOTE-DNN	CIC-MalMem-2022	✓	58,596	4
Families	Shafin et al. [15]	F1-score	0.720	CNN	CIC-MalMem-2022	✓	58,596	16
	Vinayakumar et al. [19]	F1-score	0.622	CNN-LSTM	AmritaDGA	✓	40,000	21
	This research	F1-score	0.682	SMOTE-DNN	CIC-MalMem-2022	✓	58,596	16

distinguish between obfuscated privacy malware categories and families is not enough on its own to achieve better metrics values, given that only by combining this oversampling technique with the proposed DNN it was feasible to obtain statistically significant higher metrics compared to those achieved by traditional machine learning algorithms.

The results suggest that metrics achieved by our proposed DNN architecture with the exposed preprocessing data techniques are higher than those achieved by traditional machine learning algorithms as there are more classification categories, that is, by addressing an even more complicated classification problem. Furthermore, the proposed DNN has been able to achieve comparable and even better metrics than multiclass obfuscated privacy malware classifiers based on other detection techniques.

For example, the testing F1-score achieved in this research of 0.754 for the obfuscated privacy malware categories is slightly less than the achieved by Vinayakumar et al. [31] (0.769) although their research focuses on classifying threats spread through uniform resource locators (URLs) and/or email and subject to only three categories.

In the case of the obfuscated privacy malware family classifier, SMOTE helped to reach better metrics, including AUC values. The metrics reached were even better than those achieved by Vinayakumar et al. [19], who conducted a pretty similar research (in the same way restricted to malware identified by malicious domain names) achieving metrics for different scenarios with testing F1-score ranging from 0.590 to 0.622 in comparison to the value of 0.682 achieved in this research.

Finally, the metrics reached by the multiclass classifiers of this work are comparable to those presented in [15] and [21] using the CIC-MalMem-2022 dataset. Our F1-score metric of 0.754 is the same as that achieved by Mezina and Burget [21] equal to 0.750 using CNNs to tackle the obfuscated privacy malware category classification problem.

On the other hand, the best F1-score for the classification of obfuscated privacy malware families of 0.720 reported by Shafin et al. [15] using the solution named "RobustCBL" is slightly greater than the 0.682 value achieved in this research, but our proposed solution does not use complex CNNs.

VI. CONCLUSION AND FUTURE WORK

The CIC-MalMem-2022 dataset based on memory dumping observations is suitable for recognizing between benign and malicious samples, allowing to reach high metrics in the building of classifiers for this purpose. Traditional machine learning algorithms and deep learning techniques can be used with this dataset to successfully tackle the binary classification problem. However, the results reached in this research suggest that traditional machine learning algorithms are limited in their ability to detect and classify multiclass obfuscated privacy malware based on memory dumping observations due to the challenge that its complex patterns and behaviors pose.

For that reason, deep learning is currently being extremely used, and novel data preprocessing techniques and architectures such as those exposed in this research will be proposed in the future to tackle multiclass obfuscated privacy malware classification and its applications. As long as new techniques are developed to detect and classify obfuscated privacy malware, new categories and families will also appear, further increasing the importance of research in this field.

For this reason, as future work, the datasets to be developed (based on memory dumping analysis or another technique) will need to manage a substantial improvement in the quality of the data to achieve effective controls aimed at safeguarding users' personal data, which constitutes a real challenge due to the complexity of the malware structure and the similar patterns and behaviors at runtime between categories and families.

ACKNOWLEDGMENT

The authors gratefully acknowledge the financial support provided by the Escuela Politécnica Nacional.

REFERENCES

[1] I. Essefi, H. B. Rahmouni, T. Solomonides, and M. F. Ladeb, "HIPAA controlled patient information exchange and traceability in clinical processes," in *Proc. IEEE 9th Int. Conf. Sci. Electron., Technol. Inf. Telecommun. (SETIT)*, May 2022, pp. 452–460.

[2] A. N. Jahromi, S. Hashemi, A. Dehghantanha, R. M. Parizi, and K. R. Choo, "An enhanced stacked LSTM method with no random initialization for malware threat hunting in safety and time-critical systems," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 4, no. 5, pp. 630–640, Oct. 2020.

[3] T. Carrier, P. Victor, A. Tekeoglu, and A. H. Lashkari, "Detecting obfuscated malware using memory feature engineering," in *Proc. 8th Int. Conf. Inf. Syst. Security Privacy (ICISSP)*, 2022, pp. 177–188.

[4] H. Huseynov, K. Kourai, T. Saadawi, and O. Igbe, "Virtual machine introspection for anomaly-based keylogger detection," in *Proc. IEEE 21st Int. Conf. High Perform. Switching Routing (HPSR)*, May 2020, pp. 1–6.

[5] S. Homayoun, A. Dehghantanha, M. Ahmadzadeh, S. Hashemi, and R. Khayami, "Know abnormal, find evil: Frequent pattern mining for ransomware threat hunting and intelligence," *IEEE Trans. Emerg. Topics Comput.*, vol. 8, no. 2, pp. 341–351, Apr. 2020.

[6] S. Shukla, G. Kolhe, P. D. S. Manoj, and S. Rafatirad, "Stealthy malware detection using RNN-based automated localized feature extraction and classifier," in *Proc. IEEE 31st Int. Conf. Tools Artif. Intell. (ICTAI)*, Nov. 2019, pp. 590–597.

[7] Y. Lee, S. Woo, Y. Song, J. Lee, and D. H. Lee, "Practical vulnerability-information-sharing architecture for automotive security-risk analysis," *IEEE Access*, vol. 8, pp. 120009–120018, 2020.

[8] M. Dener, G. Ok, and A. Orman, "Malware detection using memory analysis data in big data environment," *Appl. Sci.*, vol. 12, no. 17, p. 8604, Aug. 2022.

[9] C.-W. Chen, C.-H. Su, K.-W. Lee, and P.-H. Bair, "Malware family classification using active learning by learning," in *Proc. 22nd Int. Conf. Adv. Commun. Technol. (ICACT)*, Feb. 2020, pp. 590–595.

[10] A. H. Lashkari, B. Li, T. L. Carrier, and G. Kaur, "Volmemlyzer: Volatile memory analyzer for malware classification using feature engineering," in *Proc. Reconciling Data Anal., Automation, Privacy, Secur., Big Data Challenge*, Jul. 2021, pp. 1–8.

[11] D. Mu, Y. Du, J. Xu, J. Xu, X. Xing, B. Mao, and P. Liu, "POMP++: Facilitating postmortem program diagnosis with value-set analysis," *IEEE Trans. Softw. Eng.*, vol. 47, no. 9, pp. 1929–1942, Sep. 2021.

[12] Y. Xu, D. Li, Q. Li, and S. Xu, "Malware evasion attacks against IoT and other devices: An empirical study," *Tsinghua Sci. Technol.*, vol. 29, no. 1, pp. 127–142, Feb. 2024.

[13] A. Cletus, A. A. Opoku, and B. A. Weyori, "A novel hybrid features with ensemble and data augmentation for efficient and resilient malware variant detection," *Int. J. Eng. Trends Technol.*, vol. 71, no. 8, pp. 439–457, Aug. 2023.

[14] S. Aurangzeb and M. Aleem, "Evaluation and classification of obfuscated Android malware through deep learning using ensemble voting mechanism," *Sci. Rep.*, vol. 13, p. 3093, Feb. 2023.

[15] 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.

[16] A. Hidouri, N. Hajlaoui, H. Touati, M. Hadded, and P. Muhlethaler, "A survey on security attacks and intrusion detection mechanisms in named data networking," *Computers*, vol. 11, no. 12, p. 186, 2022. [Online]. Available: https://www.mdpi.com/2073-431X/11/12/186

[17] L. Yang, G. Liu, Y. Dai, J. Wang, and J. Zhai, "Detecting stealthy domain generation algorithms using heterogeneous deep neural network framework," *IEEE Access*, vol. 8, pp. 82876–82889, 2020.

[18] H. Setiawan, P. A. W. Putro, and Y. R. Pramadi, "Comparison of LSTM architecture for malware classification," in *Proc. Int. Conf. Informat., Multimedia, Cyber Inf. Syst. (ICIMCIS)*, Nov. 2020, pp. 93–97.

[19] R. Vinayakumar, K. P. Soman, S. Akarsh, and M. Elhoseny, *Improved DGA Domain Names Detection and Categorization Using Deep Learning Architectures With Classical Machine Learning Algorithms*. Cham, Switzerland: Springer, 2019, pp. 161–192.

[20] CIC. (2023). *Malware Memory Analysis CIC-MALMEM-2022*. Accesed: Jul. 28, 2023. [Online]. Available: https://www.unb.ca/cic/datasets/malmem-2022.html

[21] A. Mezina and R. Burget, "Obfuscated malware detection using dilated convolutional network," in *Proc. 14th Int. Congr. Ultra Mod. Telecommun. Control Syst. Workshops*, 2022, pp. 110–115.

[22] I. Yilmaz, A. Siraj, and D. Ulybyshev, "Improving DGA-based malicious domain classifiers for malware defense with adversarial machine learning," in *Proc. IEEE 4th Conf. Inf. Commun. Technol. (CICT)*, Dec. 2020, pp. 1–6.

[23] N. Mohamed and B. Belaton, "SBI model for the detection of advanced persistent threat based on strange behavior of using credential dumping technique," *IEEE Access*, vol. 9, pp. 42919–42932, 2021.

[24] P. Mishra, P. Aggarwal, A. Vidyarthi, P. Singh, B. Khan, H. H. Alhelou, and P. Siano, "VMShield: Memory introspection-based malware detection to secure cloud-based services against stealthy attacks," *IEEE Trans. Ind. Informat.*, vol. 17, no. 10, pp. 6754–6764, Oct. 2021.

[25] K. J. Jones and Y. Wang, "Malgazer: An automated malware classifier with running window entropy and machine learning," in *Proc. 6th Int. Conf. Mobile Secure Services*, Feb. 2020, pp. 1–6.

[26] J. Yang and Y. Guo, "AEFETA: Encrypted traffic classification framework based on self-learning of feature," in *Proc. 6th Int. Conf. Intell. Comput. Signal Process. (ICSP)*, Apr. 2021, pp. 876–880.

[27] J. Zhang, M. Kwon, S. Han, N. S. Kim, M. Kandemir, and M. Jung, "FastDrain: Removing page victimization overheads in NVMe storage stack," *IEEE Comput. Archit. Lett.*, vol. 19, no. 2, pp. 92–96, Jul. 2020.

[28] G. Mani, V. Pasumarti, B. Bhargava, F. T. Vora, J. MacDonald, J. King, and J. Kobes, "Decrypto pro: Deep learning based cryptomining malware detection using performance counters," in *Proc. Int. Conf. Auton. Comput. Self-Organizing Syst. (ACSOS)*, Sep. 2020, pp. 109–118.

[29] B. Kumar, S. Thakur, K. Basu, M. Fujita, and V. Singh, "A low overhead methodology for validating memory consistency models in chip multiprocessors," in *Proc. 33rd Int. Conf. VLSI Design 19th Int. Conf. Embedded Syst. (VLSID)*, Jan. 2020, pp. 101–106.

[30] D. Appello, P. Bernardi, A. Calabrese, S. Littardi, G. Pollaccia, S. Quer, V. Tancorre, and R. Ugioli, "Accelerated analysis of simulation dumps through parallelization on multicore architectures," in *Proc. 24th Int. Symp. Design Diag. Electron. Circuits Syst. (DDECS)*, Apr. 2021, pp. 69–74.

[31] R. Vinayakumar, K. P. Soman, P. Poornachandran, S. Akarsh, and M. Elhoseny, *Deep Learning Framework for Cyber Threat Situational Awareness Based on Email and URL Data Analysis*. Cham, Switzerland: Springer, 2019, pp. 87–124.

[32] K. Özkan, S. Isik, and Y. Kartal, "Evaluation of convolutional neural network features for malware detection," in *Proc. 6th Int. Symp. Digit. Forensic Secur. (ISDFS)*, Mar. 2018, pp. 1–5.

[33] S. Dissanayake, S. Gunathunga, D. Jayanetti, K. Perera, C. Liyanapathirana, and L. Rupasinghe, "An analysis on different distance measures in KNN with PCA for Android malware detection," in *Proc. 22nd Int. Conf. Adv. ICT Emerg. Regions (ICTer)*, Nov. 2022, pp. 178–182.

[34] J. Zhu, J. Jang-Jaccard, and P. A. Watters, "Multi-loss Siamese neural network with batch normalization layer for malware detection," *IEEE Access*, vol. 8, pp. 171542–171550, 2020.

[35] I. Idrissi, M. Azizi, and O. Moussaoui, "A stratified IoT deep learning based intrusion detection system," in *Proc. 2nd Int. Conf. Innov. Res. Appl. Sci., Eng. Technol. (IRASET)*, Mar. 2022, pp. 1–8.

[36] I. H. Sarker, Y. B. Abushark, F. Alsolami, and A. I. Khan, "IntruDTree: A machine learning based cyber security intrusion detection model," *Symmetry*, vol. 12, no. 5, p. 754, May 2020.

[37] M. Grandini, E. Bagli, and G. Visani, "Metrics for multi-class classification: An overview," 2020, *arXiv:2008.05756*.

[38] U. Gohar, S. Biswas, and H. Rajan, "Towards understanding fairness and its composition in ensemble machine learning," in *Proc. IEEE/ACM 45th Int. Conf. Softw. Eng. (ICSE)*, May 2023, pp. 1533–1545, doi: 10.1109/ICSE48619.2023.00133.

[39] Scikit Learn. (2023). *Scikit-Learn Machine Learning in Python*. Accesed: Jul. 28, 2023. [Online]. Available: https://scikit-learn.org/stable/

[40] Google. (2023). *Google TensorFlow*. Accesed: Sep. 29, 2023. [Online]. Available: https://www.tensorflow.org/?hl=es-419

[41] J. Brownlee. (2023). *Modeling Pipeline Optimization With Scikit-Learn*. Accesed: Sep. 29, 2023. [Online]. Available: https://machinelearningmastery.com/modeling-pipeline-optimization-with-scikit-learn/

[42] J. Brownlee. (2023). [Online]. Available: https://machinelearningmastery.com/use-keras-deep-learning-models-scikit-learn-python/

[43] A. O. Almashhadani, M. Kaiiali, S. Sezer, and P. O'Kane, "A multi-classifier network-based crypto ransomware detection system: A case study of locky ransomware," *IEEE Access*, vol. 7, pp. 47053–47067, 2019.

[44] S. Alabadee and K. Thanon, "Evaluation and implementation of malware classification using random forest machine learning algorithm," in *Proc. 7th Int. Conf. Contemp. Inf. Technol. Math. (ICCITM)*, Aug. 2021, pp. 112–117.

DAVID CEVALLOS-SALAS was born in Quito, Ecuador. He received the bachelor's degree (summa cum laude) in electronics and information networks from Escuela Politécnica Nacional, in 2014, and the Master of Science degree (summa cum laude) in information systems and a mention in information security management from Universidad UTE. He has also earned several international certifications in the field of computer science and networking. As a computer programmer he has experience working for private and public institutions in Ecuador. His main research areas are security, distributed computing, programming, software-defined networking, routing, digital television, big data, and cloud computing.

FELIPE GRIJALVA (Senior Member, IEEE) received the B.S. degree in electrical engineering and telecommunications from the Army Polytechnic School, Quito, Ecuador, in 2010, and the M.Sc. and Ph.D. degrees in electrical engineering (major in computing engineering) from the University of Campinas, Campinas, Brazil, in 2014 and 2018, respectively. He is currently a full-time Professor with Universidad San Francisco de Quito (USFQ), Quito. His research interests include spatial audio, machine learning, computer vision applications, and assistive technologies aimed at visually impaired people.

JOSÉ ESTRADA-JIMÉNEZ received the bachelor's degree from Escuela Politécnica Nacional (EPN), Quito, Ecuador, in 2007, and the M.S. and Ph.D. degrees from Universitat Politécnica de Catalunya (UPC), Barcelona, Spain, in 2013 and 2020, respectively. His current research interests include encompass data privacy and information security.

DIEGO BENÍTEZ (Senior Member, IEEE) received the B.Eng. degree in electrical engineering from Escuela Politécnica Nacional, Quito, Ecuador, in 1994, and the M.Sc. and Ph.D. degrees in electrical engineering from the Institute of Science and Technology, The University of Manchester, Manchester, U.K., in 1997 and 2001, respectively. From 2005 to 2007, he was a Postdoctoral Research Associate with the Sensing, Imaging, and Signal Processing Research Group, School of Electrical and Electronic Engineering, The University of Manchester. From 2007 to 2012, he was a Senior Research Engineer with the Bosch Research and Technology Center, Pittsburgh, PA, USA. He was also an Academic Visitor with the Institute for Complex Engineered Systems and the INFER Laboratory, Carnegie Mellon University. From 2012 to 2014, he was a Visiting Research Scholar with Universidad de las Fuerzas Armadas ESPE under the Prometeo Program of SENESCYT, Ecuador. He is currently a full-time Professor with Universidad San Francisco de Quito (USFQ), Quito, where he has cofounded and led the Applied Signal Processing and Machine Learning Research Group. He holds 26 granted and pending U.S. and overseas patents. His research interests include signal and image processing, pattern recognition and machine learning, intelligent instrumentation and measurement systems for medical, energy management, security, and smart building applications. He has authored more than 160 refereed journals and conference papers on these topics. He has served as the IEEE Ecuador Section Chair, from 2018 to 2019.

ROBERTO ANDRADE (Member, IEEE) received the degree in electronics and telecommunications engineering from Escuela Politécnica Nacional (EPN), in 2007, the master's degree in network and telecommunications management from the Army Polytechnic School (ESPE), in 2013, and the Ph.D. degree in security systems from the School of Systems Engineering, EPN, in 2023. He worked in the security areas of the Ministry of Education of Ecuador (MINEDUC) SENPLADES. He has been a certified Technical Instructor of CCNA, CCNP, and CCNA Security with EPN, since 2010.