International Journal of Engineering Trends and Technology
ISSN: 2231–5381 / https://doi.org/10.14445/22315381/IJETT-V71I8P238

Volume 71 Issue 8, 439-457, August 2023

Original Article

A Novel Hybrid Features with Ensemble and Data Augmentation for Efficient and Resilient Malware Variant Detection

Azaabi Cletus[1], Alex Akwasi Opoku[2], Benjamin Asubam Weyori[3]

[1]*Department of Computer Science and Informatics, University of Energy and Natural Resources, Sunyani, Ghana. St. John Boscos College of Education, Narongo, Ghana.*
[2]*Department of Mathematics and Statistics, University of Energy and Natural Resources, Sunyani, Ghana.*
[3]*Department of Computer Science and Informatics, University of Energy and Natural Resources, Sunyani, Ghana.*

[1]*Corresponding Author : cleinhim@yahoo.com*

Received: 05 April 2023 Revised: 16 May 2023 Accepted: 14 July 2023 Published: 15 August 2023

Abstract *- The use of Machine Learning (ML) solutions in place of signature-based detection systems is widely explored and settled. However, Poor features for efficient classification, malware obfuscation, class imbalance problem resulting in the accuracy paradox, and the use of conventional ML algorithms remain some of the challenges. The paper proposed a novel hybrid feature set with an ensemble algorithm and data augmentation technique for efficiently detecting obfuscated malware. An imbalance malware dataset (11,678 malware and 3,963 benign ware) was obtained from virusTotal.com and preprocessed. Features were obtained based on the dynamic disassembly of the malware dataset. We extracted only fine-grained API (application programming interface) call features and DLL (dynamic link library) features using the IDA Pro and Volatility tools, respectively. We hybridized these features into an integrated feature set and used them to train Random Forest (RF), Gradient Boosting (GB), and eXtremeGradient Boosting (XGB) ensembles. As a dataset with an imbalance class, we applied Adaptive Synthetic Sampling (ADASYN) to rebalance the dataset to improve performance accuracy. We evaluated the accuracy of the models before and after applying the ADASYN technique to overcome the accuracy paradox. Similarly, we tested the resilience of the models against malware obfuscation by measuring the performance before and after obfuscating the malware dataset. The results show that using ADASYN reduced the accuracies of the models with RF from 99.94% without ADASYN to 99.86%, GB from 99.89% to 99.81%, and XGB from 99.95% to 99.87%. However, F1-Score and AUC appreciated: RF from 83% to 94%, GB from 72% to 83%, and XGB from 85% to 96%. AUC: RF from 86.36% to 95.56%, GB from 84.82% to 94.02%, and XGB from 89.39% to 98.59%. With resilience against obfuscated malware, accuracy, F1-Score, and AUC remain the same before and after malware obfuscation. We concluded that the approach improved classification accuracy and demonstrated resilience against malware obfuscation. This result implies that with the current exponential growth in malware volumes, variety and complexity, using the proposed novel fine-grained features with ensemble technique and ADASYN improved malware classification accuracy and resilience against malware obfuscation. Thus, it presents a huge potential for malware classification in general and obfuscated malware detection in particular.*

Keywords *- Data augmentation, Ensemble, Features, Hybrid features, Malware, Machine learning, Polymorphic Malware, Signature-based detection.*

1. Introduction

The growth and expansion of malware attacks remain a major problem for all cyber defense stakeholders as malware attackers adopt ways of circumventing signature-based defense methods requiring novel and innovative defense [1]; [2]; [3]; [4]. The mostly used cyber-attack technique used by attackers is malware and Potentially Unwanted Software (PUS) binaries [5]; [6]; [7]. Malware (Malicious Software) refers to software programs with malicious intent designed to cause damage and steal information without the victim's knowledge. They include but are not limited to viruses, worms, Trojans, rootkits, botnets, spyware, adware, and others [8]. Malware attacks mainly aim at illegally extracting confidential information to compromise information systems, disrupt information systems, and/or destroy such systems [8]. When this happens, the collateral damage to the victim is usually dire, including but not limited to reputational loss,

financial loss, regulatory issues, and compliance issues [9]. As a result, all stakeholders are required to adopt innovative techniques, processes and procedures to ensure optimum security of data resources and to prevent unauthorized access, disclosure and or modifications resulting in loss of confidentiality, integrity and availability.

However, defense against malware attacks remains a problem for industry and academia as malware authors develop innovative techniques to evade detection by the usual signature-based methods such as static, dynamic and hybrid techniques used mainly by the community's anti-virus companies [1]. They achieve this by exploiting the vulnerabilities or limitations inherent in these defense mechanisms leading to exposures and the consequential effect. These mechanisms include static, dynamic, hybrid and, recently, the use of ML techniques. Static analysis is an analysis of the suspected binary without the actual execution of the code and simply looking for signs and symptoms. Static analysis is susceptible to malware packing, encrypting, and other evasive techniques, hence the need for improved techniques [10]; [1].

Dynamic analysis is the execution of malware's binary code in a safe and isolated environment to monitor the behavior and runtime activities of the malware. [8]; [11];[12]. The dynamic analysis provides a limited view since it follows a single path, making the approach time-consuming. It is also limited due to time-dependent malware such as logic bombs and bots, which only execute after a timed event. Owing to the demerits of static and dynamic malware defense techniques, approaches involving integrating static and dynamic features, known as hybrid techniques, have been explored. This technique has been shown to improve malware detection as both weaknesses are compensated, leading to improved performance. Similarly, they are also known to be resource-intensive and have problems with feature redundancies and high computational resource requirements [13]. This call for innovative techniques with adaptability to handle the ever-changing malware strategies.

Consequently, the use of automated tools and techniques for malware classification was introduced with huge success using machine-learning algorithms where static, dynamic and hybrid features are used to train the algorithms for prediction and or classification. These include Decision trees, Neural Networks, Bayesian learning and others [1]. These techniques have shown to be adaptable and learnable and are able to detect not only known malware but also new and novel malware samples. Thus, with the growth in Artificial Intelligence (AI) and variants, including ML techniques, a number of these tools have been explored in defense of malware[14]. These techniques make use of static-based features, dynamic-based and hybrid-based features (combining static and dynamic features) in the training of the ML algorithms. Though the ML techniques appear to be a panacea for cybersecurity, there are also some limitations and gaps that need further attention to provide the required defense for the ever-growing, revolving and evolving malware ecosystem.

Thus, notwithstanding the reported improvement over the signature techniques, a number of limitations exist that are exploited, leading to compromise [15]. For instance, having an efficient and novel feature set for improved malware classification remains a challenge as authors tend to use all the features revealed during the analysis for the training of the ML. This includes noise in the features as some of them are redundant and do not add to the model's predictive capability, resulting in poor classification accuracy or performance in general.

Besides, determining the real accuracy of machine-learning models with imbalanced malware datasets is challenging. This is because using accuracy metrics with imbalanced dataset results in the “accuracy paradox”. To overcome the ‘accuracy paradox” requires data rebalancing techniques such as undersampling or oversampling methods. In addition, determining the resilience of malware detection models against malware obfuscation remains challenging. This is because, as the malware sample is obfuscated, the byte sequence changes, making it difficult for signature-based detection tools to accommodate, leading to exposures.

Moreover, the problem of conventional or traditional machine-learning algorithms using imbalanced datasets, which leads to poor and inaccurate performance accuracies, remains a major challenge for all malware defense stakeholders. These limitations form the bases for our study. Therefore, the study's objective was to propose a novel hybrid feature set with an ensemble algorithm and a data augmentation technique for improved malware detection accuracy and resilience against obfuscated malware attacks. This broad objective was broken into four sub-objectives as follows:

- To propose novel homogeneous hybrid features based on dynamic disassembly of malware dataset.
- Overcome the ‘accuracy paradox’ with an imbalanced malware dataset using a data augmentation technique.
- Demonstrate the resilience of the proposed technique against obfuscated malware attacks.
- Demonstrate the resilience of ensemble algorithmic techniques to overcome data imbalances.

The next section discusses the related literature, the identified research gaps, the prosed approach and the contribution of the study.

2. Related Works, Research Gaps, our Approach and Contribution

2.1. Related Literature

Recently, efforts to improve malware detection and analysis have adopted varied approaches by combining different features and other machine-learning techniques to improve detection and overcome the limitations of the traditional signature-based methods. [16] proposed a two-stage ransomware detection using dynamic analysis and machine learning based on suspicious activities, API calls, registry keys, and file extensions. They reported an accuracy of 97.3%, a false positive rate (FPR) of 4.8 percent, and a false negative rate of 1.5%.

Sing et.al. [16] detected malicious software by analyzing the behavioral artifacts using a machine-learning algorithm. They used printable strings, single values, and Shanon entropy with an integrated feature for improving performance accuracy. They reported an achieved accuracy of 99.5%. [17] proposed a new feature engineering approach for better classification of polymorphic malware combining hybrid structural and behavioral features and measured accuracy performance using recall, accuracy, precision, and F1Score. They reported a 12% improvement in accuracy between raw features and their features and a better feature selection technique. [18] used a heterogeneous hybrid based on machine learning by extracting features from input files, feature engineering the features, and applying machine learning for classification. They reported that the Random Forest model produced better results compared with other techniques but failed to provide the metric.

In [11], they conducted a memory forensics analysis of ransomware using digital forensic tools. They concluded that the decision tree presented a higher accuracy of 90.9%. Similarly, [19] conducted an experiment on a malware detection approach based on artifacts in memory images and dynamic analysis using API calls. They concluded that their model had an accuracy of 98.5% and a processing time of 1.7 ms. [13] They proposed a forensic analysis and concluded that the usual heterogeneous hybrid is resource-hungry, such as in processor time, memory consumption, and time-consuming venture.

Similarly, [20] proposed an improved detection and classification using API call sequence and sequence alignment algorithms and claimed an improvement over state of the art. [1] proposed a preliminary analysis technique for malware detection with IOT-generated data in the form of opcodes using Random Forest(RF), K-Nearest Neighbor (K-NN) and Support Vector Machine(SVM). They reported that the RF models outperformed the others at 98% while the two at 91%. Similarly, [1] proposed and rule-based technique to detect maliciousness in IOT malware files. They concluded that there was a need for new detection techniques with accuracy, customizability, and efficiency to run in less time and thus proposed their LOKI technique. A dynamic malware analysis with feature engineering and feature learning was proposed by [21] using API features with arguments and low-cost feature extraction technique, deep neural network and reported an accuracy of 98.80%. However, this accuracy was without data augmentation, which may make the models biased in the prediction.

Similarly, [22] presented a malware variant detection technique based on the use of static and dynamic features. They extracted features from malware files and used Deep Neural Network, XGBoost and RF. They reported their best accuracy of 96.3%. The dataset is imbalanced with a dataset of 7000 malware and 3000 benign ware. Thus, the accuracy reported was without data augmentation, and that might make the accuracy skewed.

Proposed[23] a dynamic feature dataset for the prediction of ransomware. They obtained an average accuracy of 0.99 for gradient-boosted regression trees, RF, and Neural Networks. However, the dataset size of 50 ransomware and 50 malware was woefully smaller and might lead to overfitting of the models. Thus, while recent literature addresses the topic, gaps need filling in, and this study sought to improve. These gaps are presented in the next subsection.

2.2. Identified Gaps in Current ML Techniques

Nevertheless, with all the efforts to ensure optimum security against malware attacks, challenges are still posed for both the research and industrial communities. A careful analysis of the literature shows that, despite the efforts by the research community and industry in finding efficient techniques for improving dynamic malware analysis in particular and general malware detections in general, the following gaps and limitations still exist in the studied works, which our study filled:

- The need for relevant features in malware classification remains a problem with the use of automated tools as the use of heterogeneous hybrid features from static and dynamic methods are shown to be computationally resource-hungry, with redundant, multi-collinear features and high computational complexity. We filled this gap by proposing and demonstrating a homogeneous hybrid feature for efficient and resilient malware classification.
- An analysis of the works demonstrates that the datasets used in their experiments are largely imbalanced in terms of malware and benign software combinations. However, most of the works used accuracy metrics as a performance measure. With such highly imbalanced datasets, the use of "accuracy" leads to the "accuracy paradox," where the algorithm is biased or skewed towards the majority sample at the expense of the minority, giving a false impression about the performance of the model. We filled

this gap by employing ADASYN oversampling technique to rebalance the data before the modelling.

- Limited works focused on experimentally demonstrating the resilience of malware classification approaches towards obfuscated malware, which is the method mainly used by malware attackers to exploit dynamic or signature-based detection methods. We filled this gap by evaluating our proposed technique with ensembles on normal malware datasets and obfuscated malware samples as an innovation towards changes in malware binaries.
- The base or traditional machine learning algorithms give inaccurate performance measures when the dataset is imbalanced. Standard classification models, including logistic regression, decision trees, and others, show bias towards the majority class and treat the minority class as mere noise, given the high probability of misclassification of the minority class compared with the majority. We filled this gap by using ensemble algorithmic techniques with data augmentation, given a true performance by our models.

2.3. The Proposed Approach and Contribution of the Study

Consequently, this paper proposed a novel hybrid feature technique with an ensemble algorithm and data augmentation technique for improved malware detection against obfuscated malware samples of binaries. We filled these gaps by dynamic disassembly of malware samples and extracting fine-grained features from the runtime behavior of the malware to train ensembles resulting in improvement in malware classification. We did this based on a dynamic disassembly of the malware dataset, extraction of features, and hybridization of extracted features (API calls and DLL functions). We conducted dynamic disassembly of a malware sample using dynamic analysis tools. We extracted API Call features using the IDA Pro tool and Dynamic Link Library (DLL) features using the volatility tool. The extracted features were feature-engineered to obtain relevant features for the training of the ensembles. We hybridized or integrated the two features using binary hybridization, or feature presentation technique, and used it to train and test ensembles. We obtained an imbalanced dataset made up of malware and benign software. The imbalanced dataset needed some techniques to reduce bias and skewness in the classification accuracy and present the true performance of the models. Thus, we applied the Adaptive Synthetic Sampling technique (ADASYN) to increase the minority sample in the training data. We integrated the individual features into an integrated feature set and used it to train and test the ensembles' performances before and after obfuscation, first without ADASYN and later with the ADASYN technique. In addition, we evaluated the models' performances before and after obfuscation without ADASYN and with the ADASYN data augmentation technique. The performance of the models' accuracy without ADASYN was compared with that of the one with the ADASYN technique. Similarly, the performance of the models on normal malware and obfuscated malware was determined to show the resilience of the approach in handling malware obfuscation and, above all, the efficiency of the proposed features in classifying malware determined.

From the study results, our approach demonstrated the efficiency of the proposed homogeneous features in classifying malware samples as the classification performance of the models using the features improved malware detection compared with the state-of-the-art. The performance accuracy obtained with our approach is true classification accuracy without bias or skewedness due to the use of the ADASYN data augmentation technique. Hence, we avoided the "accuracy paradox," which is common with imbalanced data in general and malware datasets in particular when the accuracy is determined without the application of data augmentation techniques. The approach showed resilience or robustness in detecting obfuscated malware, which means variants of both known and unknown malware, were efficiently detected and classified compared with the current approaches. Besides, the limitations with conventional or traditional machine learning algorithms with imbalance datasets were overcome by using hybrid ensemble algorithmic techniques with the use of the ADASYN technique leading to improved classification performance compared with the cited literature.

2.3.1. Contribution of the Paper

Overall, we have demonstrated considerable novelty and or originality, validity, reliability, and clarity of presentation through our rigorous scientific and experimental processes. Our results are exciting and of modest but significant value, and moderately contribute to knowledge in malware analysis in particular and automated tools in cybersecurity using artificial intelligence (AI) in general. Specifically, the paper highlighted or contributed to knowledge as follows:

- We proposed and demonstrated an efficient homogeneous hybrid malware feature set comparable with the usual heterogeneous hybridization with reduced computational complexity, feature redundancy, and computational resource requirements resulting in improved malware and variant classification performance. Thus, the paper provides novel malware features for efficient malware classification and variants of known and unknown malware.
- By using the data augmentation technique (ADASYN), we overcome the problem of the ‘accuracy paradox’ that occurs when the dataset is imbalanced. When the accuracy metric is used as a performance measure without data augmentation, it leads to skewedness and/or bias of the models in prediction. Hence, by our approach, we have shared light on the fact that when the malware dataset is imbalanced. Without data augmentation, other metrics such as precision, recall, AUC and F1 scores

should be used instead of accuracy. In addition, the use of data augmentation is relevant when dealing with imbalance classes in malware datasets.

- Moreover, the study also demonstrated the resilience and robustness of the proposed approach against malware obfuscation. Hence, providing a novel technique for detecting known, unknown, and zero-day malware and their variants. This resilience means known, unknown and zero-day malware can be detected, resulting in improved performance against advanced malware attacks.
- Finally, using an ensemble algorithmic technique with the ADASYN data augmentation provides a hybrid technique that improves the data imbalance problem over the use of conventional or traditional methods.

We organized the rest of the paper as follows:

In Section 2, we present the Related Literature, Research Gaps and our description of our proposed approach and the paper's contribution. Section 3 presents the Methods and Materials following a methodological framework; the study results are presented in Section 4, while we present the study's discussion in Section 5. The conclusions and future work of the study are in Section 6.

3. Materials and Methods

In this section, we present the key methodological items, processes, experimentations, procedures and methods employed in the study following the architectural-framework shown in Figure 1.

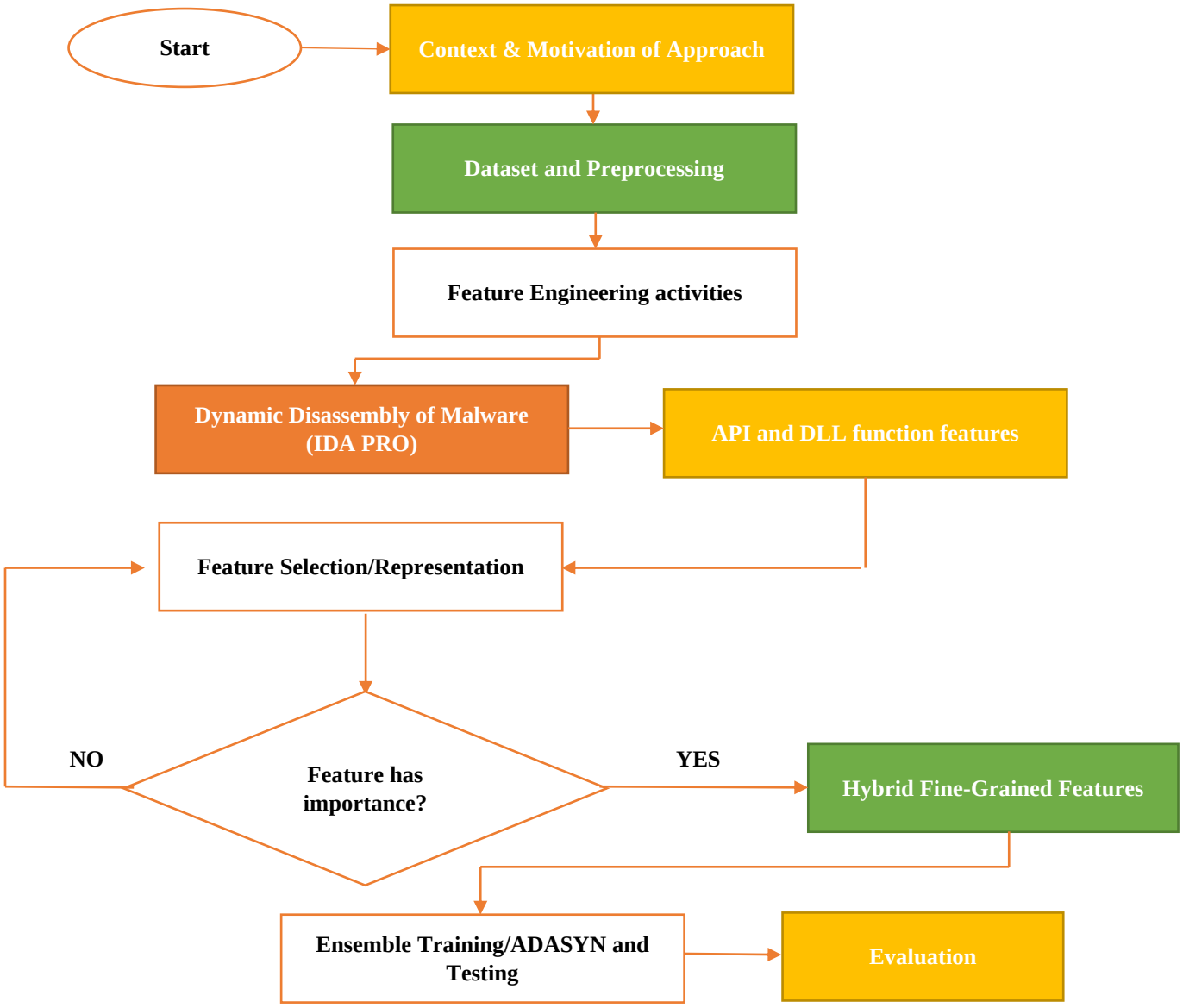

Fig. 1 The architecture of our proposed approach

In this section, the methodological framework of the study is presented. Stages are follows:
1. Description of the context and motivation of the proposed approach
2. Dataset and Preprocessing activities.
3. Dynamic disassembly of malware obtain DLL and API Call Features
4. Feature selection
5. Feature integration
6. Feature Representation
7. Model Selection and Training
8. Performance Evaluation.

3.1. Context and Motivation for the Approach

In this subsection, we present a brief discussion on dynamic malware analysis, its limitations, and how dynamic malware disassembly can reveal relevant features that can be used to train ML to improve detection performance.

3.1.1. Dynamic Malware Analysis

Dynamic analysis is the execution of malware binary code in a safe and isolated environment to monitor the behavior and runtime activities of the malware. [8]*;* [11]*;*[12]. While the malware is executing, some runtime features or behaviors can be observed that express the malware's intent. These may include *GetProcessId, CreateFile, DeleteFile,* etc. Such features can be used to train machine-learning models for automated detection purposes. Even though this technique is a relative improvement over static analysis, malware authors have adopted innovative ways that render this technique suboptimal when malware is obfuscated, or other encryption techniques are used. Such variants include polymorphic and other mutating malware. This requires the use of innovative techniques to ensure efficient and effective protection of information assets against such attacks.

3.1.2. Limitations of Dynamic Malware Analysis

Notwithstanding some gains with dynamic malware detection techniques, it has shown to have some limitations; dynamic analysis provides a limited view since it follows a single path, making the approach time-consuming. It is limited due to time-dependent malware such as logic bombs and bots, which only execute after a timed event; some malware is also environment-aware, refusing to execute in a simulated environment, and is resource-intensive. In addition, sophisticated malware such as polymorphic, metamorphic, and oligoorphic malware, and their variants, using obfuscation techniques, renders dynamic analysis suboptimal, resulting in exposures. [23]; [8];[25]. In addition, sophisticated malware obfuscation and encryption techniques render dynamic malware analysis inefficient and ineffective. Obfuscation in malware defense is the act of concealing the identity of malware from detection tools to evade detection by anti-virus scanners, leading to exposure [20]. With the increased volumes, variety, and complexity of malware, the use of traditional signature-based methods such as dynamic analysis is shown to be prone to such obfuscation and encryption techniques, including dead-code insertion, code transposition, register reassignment, sub-routine reordering, instruction substitution, and code integration techniques [19]. Since the signature-based defense techniques operate on the principle that, once identified, the malware remains the same throughout its lifespan, the current obfuscation techniques that change the malware binaries render the dynamic defense techniques suboptimal. The apparent weaknesses in both static and dynamic malware analysis prompted the need for hybrid techniques for compensatory effects.

3.1.3. Heterogeneous Hybrid Techniques

A heterogeneous hybrid is made up of a combination of static and dynamic approaches [[18]. In this, static techniques can be used and followed by dynamic techniques, or vice versa, in a machine-learning environment where static features can be used for training and dynamic features for testing, or a combination of these features can be used to train a model [14];[23]. These combinations are found to improve the performance of the models over the individual techniques [26]; [27]. Thus, though heterogeneous static-dynamic hybridization or mix offers some advantages over the individuals, they also possess some limitations with the dexterity of the evolving malware variants. Therefore, even though heterogeneous hybrid improves performances, its limitations stand as an obstacle to its efficiency and robustness as a technique. They are known to be resource-hungry in terms of processor time and memory consumption and time-consuming, including high execution times, high resource requirements (such as processors and memory), and poor analysis. The method is susceptible to high obfuscation techniques, such as polymorphic and metamorphic malware, which have poor and redundant features and lack adaptability with obfuscated malware [18];[17][13]. In addition, they use machine-learning approaches in a bit to achieve the performance, which also presents some limitations [26];[27]. To improve upon these limitations, the use of automated ML techniques has been explored widely in the literature with some success [14]; [17];[25]. Even though ML techniques have introduced adaptability, learnability and other positives in taming the malware onslaught, a critical review of the works shows that gaps need to be filled. Thus, we are motivated by the fact that some gray areas exist that needs novel and innovative techniques to fill the gaps [1]

3.1.4. Motivation for our Approach

- Notwithstanding the limitations of dynamic malware analysis, the dynamic disassembly of malware samples with dynamic tools reveals relevant features that can be used to train machine-learning algorithms with improved performance compared with other hybrid techniques. However, little or no work has focused on using API calls and DLL function features extracted from dynamic environments and used with ensemble techniques to

improve malware detection and classification performance. We explored this gap by using the hybridization of API call features and DLL function features with the ensemble for improved malware classification.

- We are motivated by the fact that recent works used accuracy as a performance measure with highly imbalanced malware datasets. This phenomenon leads to the "accuracy paradox," where the models are skewed or biased towards the majority class and may treat the minority as noise, leading to misclassification of the minority class. This does not present true performance accuracy, leading to reported false classification accuracy. Overcoming this phenomenon requires using class imbalance rectification techniques and data augmentation methods to increase the minority class for better classification. Thus, using the ADASYN data augmentation technique with the ensemble provides a reliable performance measure.
- Malware obfuscation and encryption are the methods and techniques used to exploit the weaknesses of dynamic and other signature-based detection methods. Little Works focuses on or demonstrates the resilience of their approaches against real, obfuscated malware. Thus, by testing the proposed approach against obfuscated malware, we built a robust, resilient, and efficient technique against obfuscated malware attacks.
- Finally, conventional or traditional machine-learning algorithms perform poorly with imbalanced data that malware authors mostly use. Overcoming this requires using algorithmic techniques with data augmentation methods to rebalance the data before classifications or instead use other metrics such as precision, recall, AUC and/or F1 score. Hence, we proposed to use a hybrid approach based on an ensemble with the ADASYN data augmentation method.

3.2. Dataset and Pre-Processing

For this experiment, we collected malware samples from two main sources: VirusTotal.com and malwr.com, for a four-year period 2017–2019 and 2019–2021. This was necessary because the malware landscape is evolving and revolving, and new, novel, known malware variants are emerging. By including these malware samples, we are sure to have almost all new and known malware variants. To obtain benign samples for this study, we extracted these files from the Windows operating system by collecting the most used Windows files found in the Windows file system. Using virusTotal.com, we checked whether a sample was benign or malicious when all the virus scanners flagged it as malware or not. Consequently, we combined this malware and benign ware to form our experimental dataset, which comprises 11,678 malware and 3,963 benign ware made up of different malware families. Tables 1 and 2 show the malware categories and the total dataset size used for the study, respectively.

Table 1. Malware categories and samples

Malware category	Samples
Backdoors	998
Downloaders	797
Viruses	883
Trojan Spy	1123
Trojan Droppers	234
Worms	1732
Spyware	974
Others	4937
Sub-Total	**11,678**
Benign ware	3963
Total Dataset	**15,641**

Table 2. Composition of the experimental dataset

Dataset type	Number of samples	Category
Malware	11678	Varied (from 2017-2021)
Benign ware	3963	Varied (from 2017-2021)
Total Dataset	15,641	Mixed

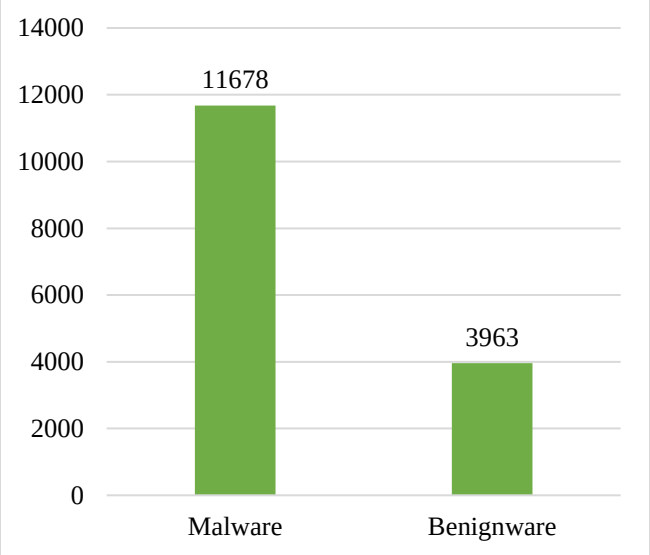

Fig. 2 Imbalance dataset

From the data exploration stage and preprocessing, it was realized that the malware dataset was imbalanced. This phenomenon, if not managed well, may result in what is usually known as the "accuracy paradox," where the prediction would be skewed or biased towards the majority sample as the dataset shows 74.66% and 25.34%, making it mildly skewed, as shown in Figure 2. Thus requiring data augmentation.

In addition, to overcome the problem of outliers, which might result in poor model prediction, we checked for outliers and removed them before we applied the technique. This is because the presence of outliers leads to poor model predictions.

3.2.1. Adaptive Synthetic Sampling (ADASYN)

To smoothen data imbalances, a number of undersampling and oversampling techniques are used, such as Synthetic Minority Oversampling Technique (SMOTE) which has its variant ADASYN. In this technique, synthetic data points are generated at the lower-density areas. To put it another way, more synthetic data is produced in areas with lower minority population densities. Otherwise, less fake data is used.

To smoothen out the imbalance in the dataset and improve the prediction of the models, we applied the ADASYN technique using the algorithm. It is a variation of the Synthetic Minority Oversample Technique (SMOTE), where it creates synthetic data according to the data density, which is inversely proportional to the density of the minority class. Low-density areas are where the synthetic data are created in the regions of the feature space. Simply put, in places where the density is lower in minority areas, synthetic data are created to augment the data points. Consequently, we applied the ADASYN technique to the training data to rebalance it. The code snippet for the ADASYN approach is as shown.

```
From the imblearn, oversampling import ADASYN,
Adasyn=ADASYN (random state=4208) X oversample ada,
Y oversample ada=
Adasyn.fit resample (X train, y train) Classifier ada= ensemble
(),
Classifier.ada.fit(X oversample ada, y oversample ada),
Print (classification report(y test, classifier ada. predict(X
test))).
```

Fig. 3 Code snippert for ADASYN algorithm

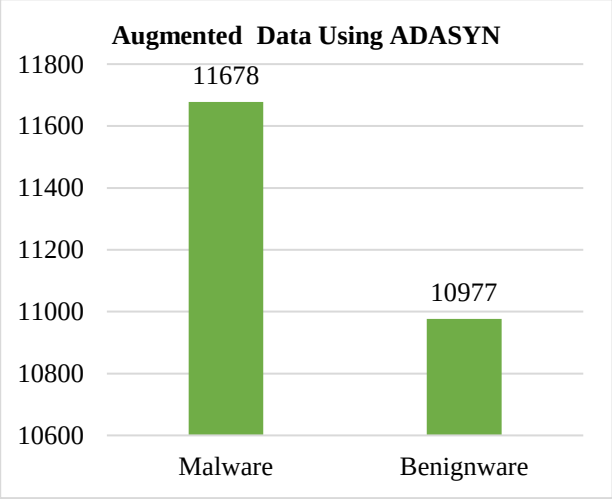

Fig. 4 Rebalanced/Augmented dataset

We explain the algorithm of the ADASYN technique as follows: From a given dataset, the majority class is N^- and the minority is N^+, respectively, captured—a preset threshold value, dth, for the maximum or highest degree of class imbalance. Therefore the total number of synthetic samples to be generated is $G=(N^- - N^+) * \beta$ ……. *(1)* . Where $\beta = (N^- / N^+)$. Thus, for every minority sample x_i, k-nearest neighbors are obtained using the Euclidian distance and the ratio $r_i/\sum r_i$. Thereafter, the total synthetic samples for each x_i are given as $g_i = r_x Xg$. We then iterate from 1 to g_i to generate the samples the same way as the Smote technique. Thus the minority sample is raised to inch close to the majority class to avoid the bias and skewedness of the models in predicting. Figure 4 shows the data after augmentation.

3.3. Dynamic Disassembly of Malware, Feature Selection and the Hybridization of Features

This section describes how we extracted API Calls and DLL function Features and how we integrated these features into a hybrid feature set to train the ensembles.

3.3.1. API Call feature extraction using IDA Pro

Dynamic disassembly of malware samples reveals relevant features that can be used to train machine-learning algorithms.[8]; [11]. IDA Pro is one of the disassembly tools used to generate accurate codes from any executable during analysis. It is used mainly for manual and automated reverse engineering extracting features. It also works well as a debugger and supports instruction tracing, scripting, tracing of functions, and logging of instructions. We adopted this tool to analyse from which we extracted Windows API Calls. To obtain the needed Windows API, we loaded the binary into the IDA Pro and analyzed the malware with the information stored in the database in .ido, .idl, .nam. We inspected this repository to identify the relevant features for the study from the import functions found in the import windows. We ignored some of the features that were not explicit in their behavior and only extracted Fine-Grained API features for the study. This is because the use of the entire features revealed in the malware analysis contains noise, which can affect the predictive capability of the models when used for classification. Through this process, the API call features were extracted for the study. The major challenge in using these features found in the database or repository without selecting relevant features is that it loads to poor prediction, and redundant features weigh heavily on computational resource requirements and other metrics. Thus, to improve upon this, as used by many authors, we selected only fine-grained features with high feature importance for the study.

3.3.2. DLL Function Calls Extraction

Volatility tools can be used to investigate malware artifacts from devices, networks, file systems and the registry of running processes. To obtain DLL features, we used the Cuckoo Sandbox (www.cuckoosandbox.com). The malware samples were run, and the DLL functions features were extracted. Using the volatility tool from Http://www.volatilityfoundatio.org), an open-source memory analysis framework for Windows 32-bit and 64-bit operating systems, we augmented the cuckoo sandbox and based on that, we extracted DLL features to hybridize with the API features for the training of the machine-learning algorithms.

The registry contains files that include DLL functions. Only fine-grained features (without noise) were selected for the training and testing of the MLS. Other artifacts that were not clear features of maliciousness were ignored. This approach ensured we obtained only relevant features as much as possible to improve our classification. Thus, following the analysis trail, we inspected the repository to identify relevant features without ambiguity that were selected in the form of DLL. The major challenge in using these features found in the repository without selecting relevant features is that it leads [13] to poor prediction and redundant features, which weigh heavily on computational resource requirements and other metrics. Thus, to improve upon this, as used by many authors, we selected only fine-grained features with high feature importance for the study.

3.3.3. Feature Selection

After the initial exploration, the results showed redundant features that negatively influenced the performance. Obtaining the 'fine-grained' features with high feature importance required the use of feature selection. In ML parlance, feature selection refers to the process of choosing features with high feature importance from the main feature set extracted [29]. This mainly improves model performance by eliminating redundant features that likely introduce noise, bias and variance into the model. By using only relevant features, the training model can concentrate on the features with high impact, thereby improving training and testing time and accuracy and minimizing error, resulting in robust models. Feature selection methods include wrappers such as forward selection, backward selection and recursive elimination methods. Filters are usually independent of the algorithm and are used for correlation; examples include Pearson correlation, ANOVA, chi-square etc. The other types include embedded methods such as LI regularization and others. Thus, we reduced the features by applying a feature selection to remove noise, improve prediction accuracy, and reduce computational cost and resources using one of the filter methods of feature selection. Feature selection is the process of weeding out irrelevant features that interfere with the predictive capacity of the models [29]. We used the Information Gain (MI) method, one of the filter techniques for feature selection. We chose it because these methods are model-agnostic in nature. This is calculated as the reduction in entropy by transforming a dataset.

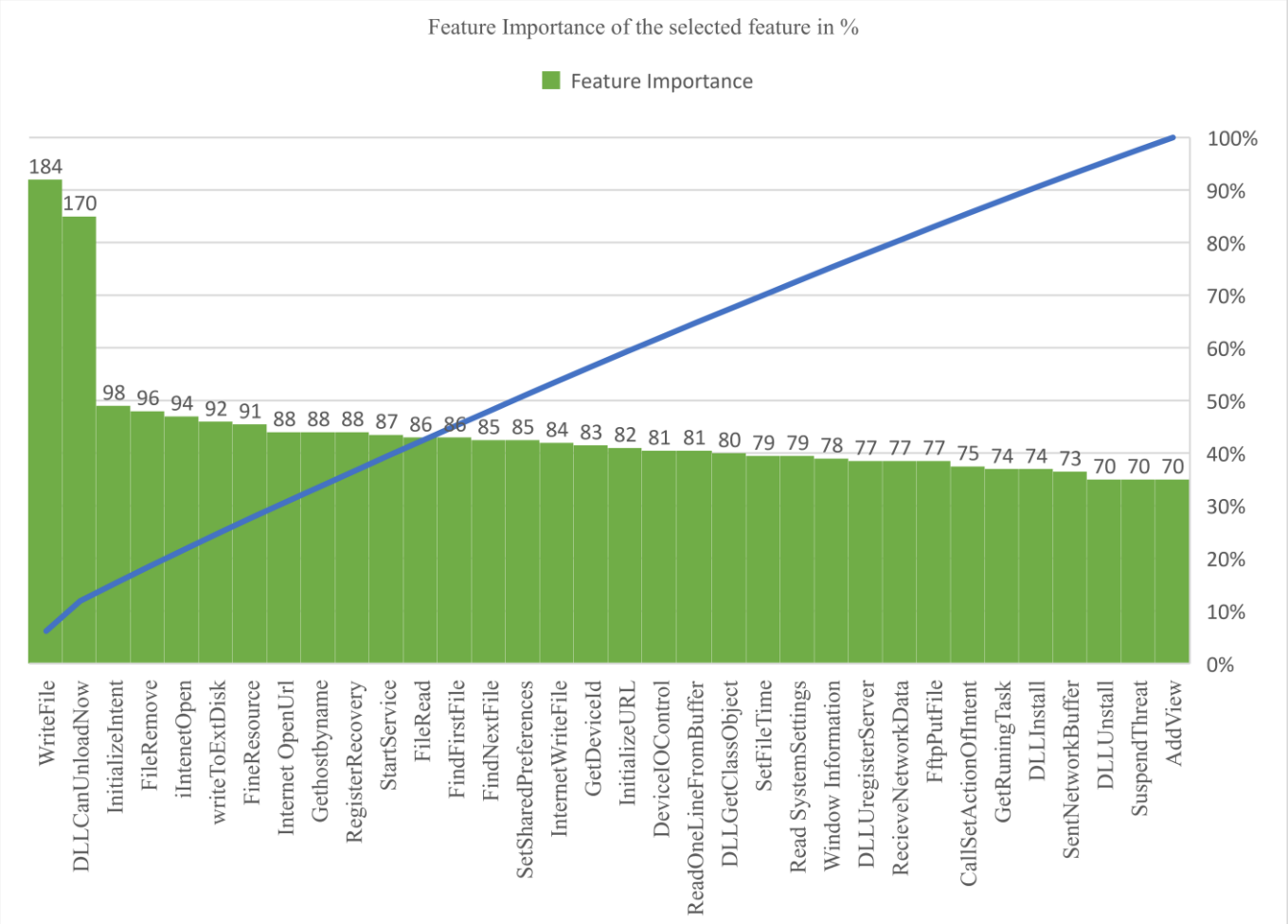

Fig. 5 Hybrid features and their importance

$$Entropy = -\sum p(x) Log2\ p(x) \quad (2)$$

The higher the entropy value, the more information the attribute has. Thus, we used this technique to evaluate the information gain of all the variables of features that have high information and that contribute to the classification capability of the models; based on this, we ordered the features based on their strength. The information gain technique was used because they are faster and avoid overfitting models [30]. Following this, we selected 36 features with the highest importance, as shown in Figure 5.

3.3.4. Feature Hybridization

To train a machine-learning model, features of the instance under modelling are important [29]. API Calls were extracted, and the DLL function features were integrated into a feature set and factorized for the use of the ensemble algorithms. Figure 6 shows the architecture of the hybridization process. This was relevant because using individual features presents some limitations [8]. He suggested that the individual techniques' limitations could be compensated when these techniques are integrated. The integration ensures that the other resulting in improved classification, compensates for the limitations of each. It also helps in the detection of broad categories of malware families.

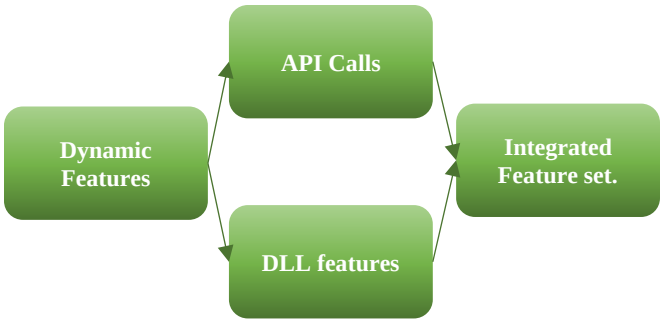

Fig. 6 The architecture of the hybrid feature integration

Thus, to combine the DLL features with API Calls, we converted each of them into a binary vector. The vectors were integrated to form an integrated feature vector for the training and testing of the models, as shown under feature representation in subsection 3.3.5 below.

3.3.5. Feature Representation

For machine-learning algorithms to be able to learn the patterns, the extracted features need to be converted into a vector form; for sequential features such as API Calls and DLL Functions, the creation of a representative vector can be done using a number of ways, including frequency feature vector, frequency-weighted feature extraction and or binary feature extraction [20]. For the purpose of this work, we used the binary frequency vector approach s described by [20]; [29]. In binary representation, the features are represented as binary features. F signifies the presence or absence of a malware binary given a resulting feature vector as V_{Fb}= (bs_1, bs_2……bs_n), where bs_i is 1 if F contains an instance of s_i or 0 if otherwise, n is the size of the sample or dataset. The figure below shows a sample feature vector for API Calls and DLL functions, as shown in figure 7.

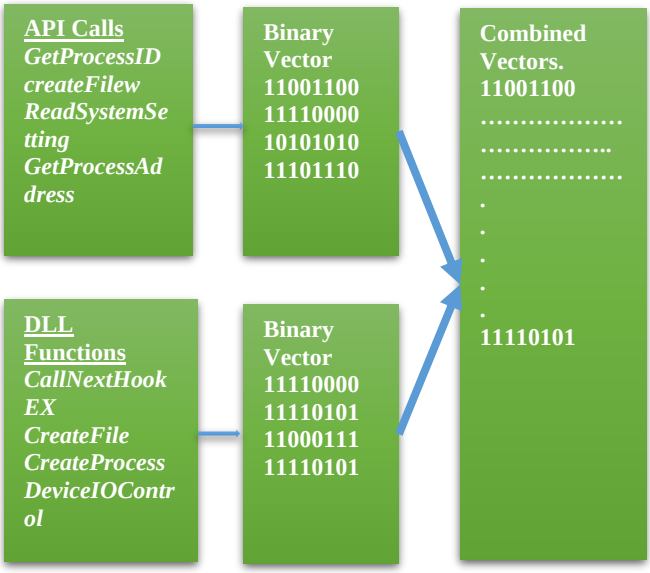

Fig. 7 Feature representation strategy

3.4. Training of the Models

One main aspect of ML techniques is how to validate the models. A number of validation techniques exist. For the purpose of this study, we used the split/test approach. This approach is where the dataset is divided into two unequal parts for training and testing purposes. The approach enables observation of the performance of the models on unseen data. Thus, we split the dataset into 70% for training and 30% for testing the models. The aim is to ensure randomness or stratification in the data so as to minimize bias or skewedness in the data that can lead to over- or under-fitting of the algorithm [19]. Though one of the simplest techniques, there may be sampling bias which is usually a systematic error that occurs due to non-random sampling of a population where some members are less likely to be included leading to the bias. However, the advantage of using this technique outweigh the demerits since the models are already ensemble-based and use data augmentation methods. Consequently, the three ensemble algorithms were trained by passing the training dataset into the models to learn patterns and later can predict unseen data. The Random Forest (RF), GB, and XGB were accordingly trained. These models were chosen because they are not prune to overfitting and can handle both linear and non-linear data. They are good at classification and have high stability [31]. Similarly, the RF is cost-effective, robust against overfitting, handles missing values efficiently, has little or no bias, and is easily interpretable [31]. These strengths far outweigh its limitations of being complicated relative to decision trees, slower, and usually working well with large datasets. The ADASYN technique was applied to the training data to rebalance the dataset at this stage.

- Thus, we first trained and tested the models with the hybrid features without obfuscating the malware and, similarly, without the application of the ADASYN technique. This was mainly to set a baseline performance of the models.
- After this, we trained and tested the performance of the models with obfuscated malware to observe the effect of obfuscation on the performance of the models.
- Similarly, to observe the effect of the data augmentation technique (ADASYN) on the performance of the models, we applied the ADASYN technique only to the training data because we were not using the entire dataset for training and testing, and we tested or evaluated the performance of the algorithms using accuracy metric, before and after the data augmentation. The performance of each of the models was observed, and hyperparameters were tuned to achieve the desired results. The description of the various parameter tuning and optimization processes of the models follows in the next subsection.

3.4.1. Random Forest (RF)

RF model is used in many fields such as banking, health care, stock market prediction, e-commerce and cybersecurity with very good success [31]. The security community has used a decision tree-based algorithm, but it is not always used alone. Many trees are trained and used together to predict in a fashion known as Random Forest. This ensures that each tree sees the data differently to improve detection outcomes. To make the prediction as to whether a binary is malware or benign, the trees are allowed to vote, and the most popular vote wins, as depicted in Figure 8 below.

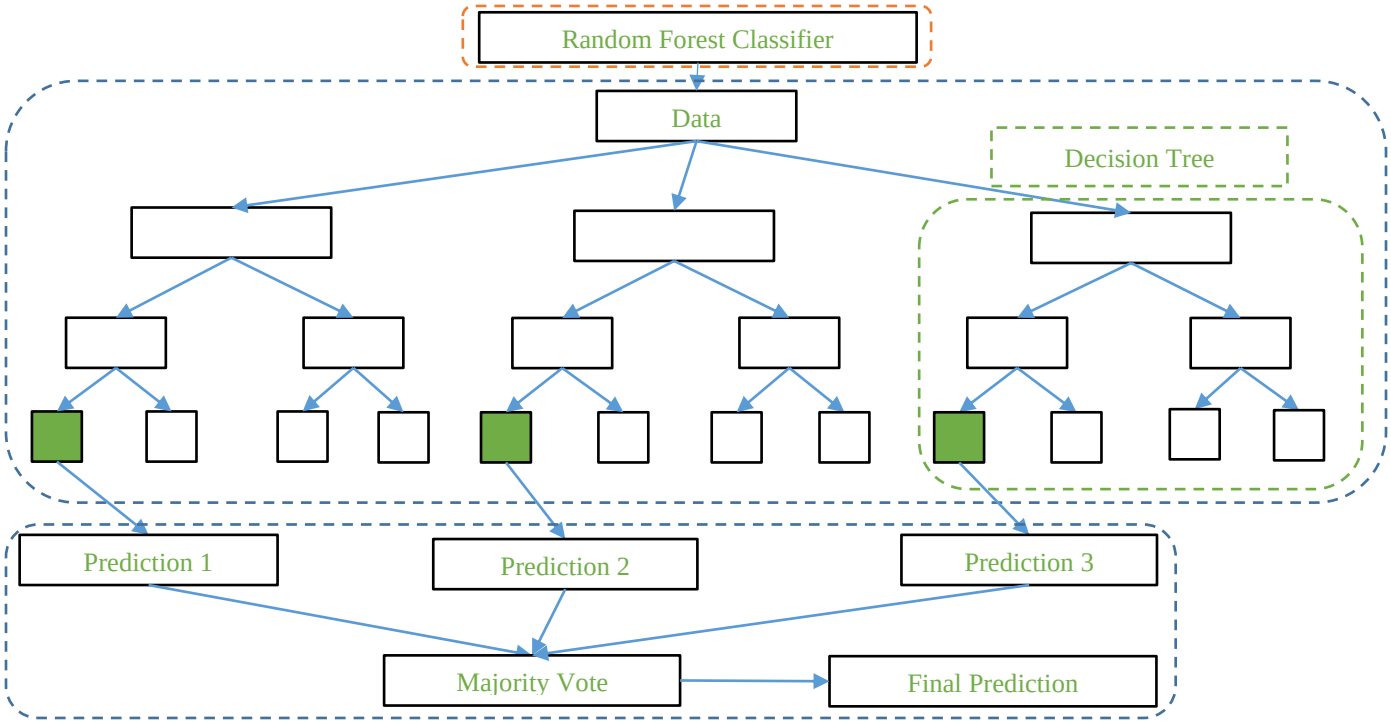

Fig. 8 Architecture of random forest

The description of the RF approach on how it classifies dataset is as explained: precisely, suppose $S = \{(x_1,y_1), \dots, (x_n, y_n)\}$ is data sampled from $R^D \times \{-1,1\}$-valued random vector (X, Y) with possibly known distribution for some positive integer $D \geq 1$. The objective of the random forest (RF) algorithm is to predict y from x using an ensemble $h = \{h_1(x), \dots, h_k(x)\}$ of classifiers for decision trees. The decision tree for the classifier $h_k(x)$ is determined by the parameter $\theta_k = \left(\theta_{k_1}, \dots, \theta_{k_p}\right)$ that is a realization of a known random variable θ, that describes the subsets of the data set S that constitute the decision trees of the classifiers in h. Thus

$$h_k(x)\ h_k(x|\theta_k). \tag{3}$$

Each tree in the random forest voted for the most popular class $y \in \{-1,1\}$ for the input data x. The vote of the k^{th} the tree is the output of the classifier $h_k(x)$. The class with the most votes wins. Therefore, the decision function is then given as

$$H(x) = arg \max_{y \in S} \sum_{i=1}^{k} I\,(h_i(x) = y), \tag{4}$$

Where $I(h_i(x) = y)$ is the indication function. The margin function for the best class $y \in S$ of a random forest is given.

$$m(x,y) = P_\theta(h(x|\theta) = y) - \max_{u \neq y} P_\theta(h(x|\theta) = u),$$

where P_θ is the probability distribution of the decision tree generating random variable θ. Note that $m(x,y)$ is the measure of the extent the probability of votes for the best class exceeds the probability for the next-best class. Therefore, the generalization error e takes the $e = P_{x,y}(m(x,y) < 0) \leq \frac{Var_{x,y}(m(x,y))}{\left(\mathbb{E}_{x,y}(m(x,y))\right)^2}$ (5)

where $P_{x,y}$, $\mathbb{E}_{x,y}$ and $Var_{x,y}$ are respectively the probability distribution, expectation and variance of the random vector (X,Y). The inequality follows from Chebyshev's inequality. Figure 9 below shows the code snippet for the RF algorithm.

```
From sklearn. Ensemble Import RandomForestClaasifier
Form sklearn Import Metrics
Rfmodel= RandomForestClassifier()
Rfmodel.fit (xtrain, ytrain)
ypredrf=rfmodel.predict(xtest)
print('Accuracy": %f%(metrics.accuracy.score(ytest,ypredrf)))
```

Fig. 9 Code snippet for RF

3.4.2. Gradient Boosting Boost

This boosting algorithm is similar to AdaBoost. It uses gradient boosting in an iterative manner to correct the estimators based on the returned values *(Parisi, 2020)*. The adjustment is made based on the residual error and not the weights. The default estimators in this model are decision trees specified in parameters.

To motivate, let us look at AdaBoost. It tries to find a classifier $f^*(x) \in \{+1,-1\}$ such that

$$f^*(x) = \text{argmin} E[e^{-Y f(X)} | X = x] \quad (6)$$

$f^*(x) = \text{argmin}[e^{f(x)} P(Y = 1|X = x) e^{f(x)} P(Y = -1|X = x)]$

$\left[e - ^a (P(Y = 1|X = x) e^{(a)} P(Y = -1|X = x)\right] = \frac{1}{2}\log\left(\frac{(P(Y=1|X=x)}{P(Y=1|X=x)}\right)$

Since argmin, α, we have

$f^*(x) = \frac{1}{2}\log\left(\frac{(P(Y=1|X=x)}{P(Y=1|X=x)}\right)$ from this, it is easy to see that,

$$P(Y = -1|X = x) = \frac{1}{1+e^{-2f^*(x)}} = \frac{e^{-f^*(x)}}{e^{-f^*(x)} + e^{-f^*(x)}} \quad (7)$$

To cast this in the Bernoulli framework, define $Y^0 = (Y + 1)/2 \in \{0,1\}$. Let $f(x) \in \{0, 1\}$ be a binary classifier and let $p(x) = P(Y = 1|X = x)$. Then the log-likelihood of Y^0 given $p(x)$ is easily seen to be $l(Y^0, f(x)) = Y^0 \log p(x) + (1 - Y^0)\log(1 - p(x)) = -\log(1 + e^{-2f(x)})$. Equivalently the cross entropy, which is equal to the negative log-likelihood, is $H(Y^0, f(x)) = \log(1 + e^{-2f(x)})$.

Therefore, for gradient boost, we use this cross entropy as a loss function for binary gradient boost. The code snippet from the sklearn is shown in 10.

```
From sklearn Import ensemble
Params={n_estimators: 500, max-depth:3, subsample: 0.5,
learning rate:0.01, min sample_leaf:1random_state:3}
Clf+ensemble.gradientBoostingClassifier(**params)
Clf.fit(xtrain,ytrain)
Ypred_clf.predct(xtest)
Print "Accuracy is:"
Print(metrics.accuracy_score(ytest,ytest)
Print ("Area under the curve:%?%
(metrics.roc_auc_score(ytest, ypredrf)
```

Fig. 10 Code snipet for GB algorithm

3.4.3. eXtreme Gadient GBoost (XGBoost)

The eXtremeGradientBoosting is one of the ensemble learners that is similar to XGBoost. It efficiently manages large datasets and extends the XGBoost's functionality. It is scalable and good for parallel computing because of the minimized residual error and less overfitting [31]. The approach is just an extension of the gradient boost, as described earlier. Therefore, to predict with the XGBoost, which is an improved version of the gradient descent, we aimed to optimize the performance. This model uses parallel computing, resulting in reduced overfitting. The code snippet is shown *below.*

```
From sklearn ImportMetrics
From xgboost.sklearn import GBClassifier
Xgb_model=XGBClassifier()
Xgb_model.fit(xtrain,ytrain,
eval_metrics=[error], eval_set=[((xtrain,
ytrain)),(xtest,ytest)])
Y_pred_xgb.model.predict(xtest)
Print ("Accuracy is:"
Print (metrics.accuracy_score(test, y_test)
Print ("Area under the curve:%?%
(metrics.roc_auc_score(ytest, ypredrf)
```

Fig. 11 Code snippets for XGB

3.5. Evaluating Performance Metrics

To determine a model's efficiency, it has to be evaluated using performance metrics. These refer to standards used to assess the characteristic and the behavior of an artifact [32]. For the purpose of this study, the following performance metrics were used**;** Accuracy, Sensitivity/Hit Rate/Recall/True Positive Rate (TPR), False positive Rate(FPR)/Precision, and True Negative Rate/specificity(TNR). These measures are obtained from the confusion matrix or contingency table, as shown below.

Table 3. Confusion matrix

Predicted	Actual Malware	Non-Malware
Malware	True Positive(TP)	False Positive(FP)
Non-Malware	False Negative(FN)	True Negative(TN)

For the purpose of this paper, we used Accuracy, Recall, F1-Score and Auc. Sensitivity is the proportion of correctly classified malware samples in the dataset. False Positive Rate = 1-specificity. Accuracy is the proportion of correctly classified observations in the dataset and is represented as shown in equation 17. Other metrics, such as True Positive Rate (TPR), False Positive Rate (FPR), and True Negative Rate (TNR), are shown in equations 18-20.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (8)$$

$$Precision = \frac{TP}{TP + FP} \quad (9)$$

$$Recall = \frac{TP}{TP + FN} \quad (10)$$

$$F1 - Score = \frac{2 * TP}{2 * TP + FP + FN} \quad (11)$$

4. Results of the Study

The study proposed a novel hybrid feature set with ensemble and data augmentation for efficient and resilient malware classification and obfuscated or malware variant detection. Specifically, to address the accuracy paradox in imbalanced data, demonstrate resilience in handling malware obfuscation, and overcome the limitations in using conventional ML techniques with class imbalances. This section presents the study's results using tables, graphs, and charts.

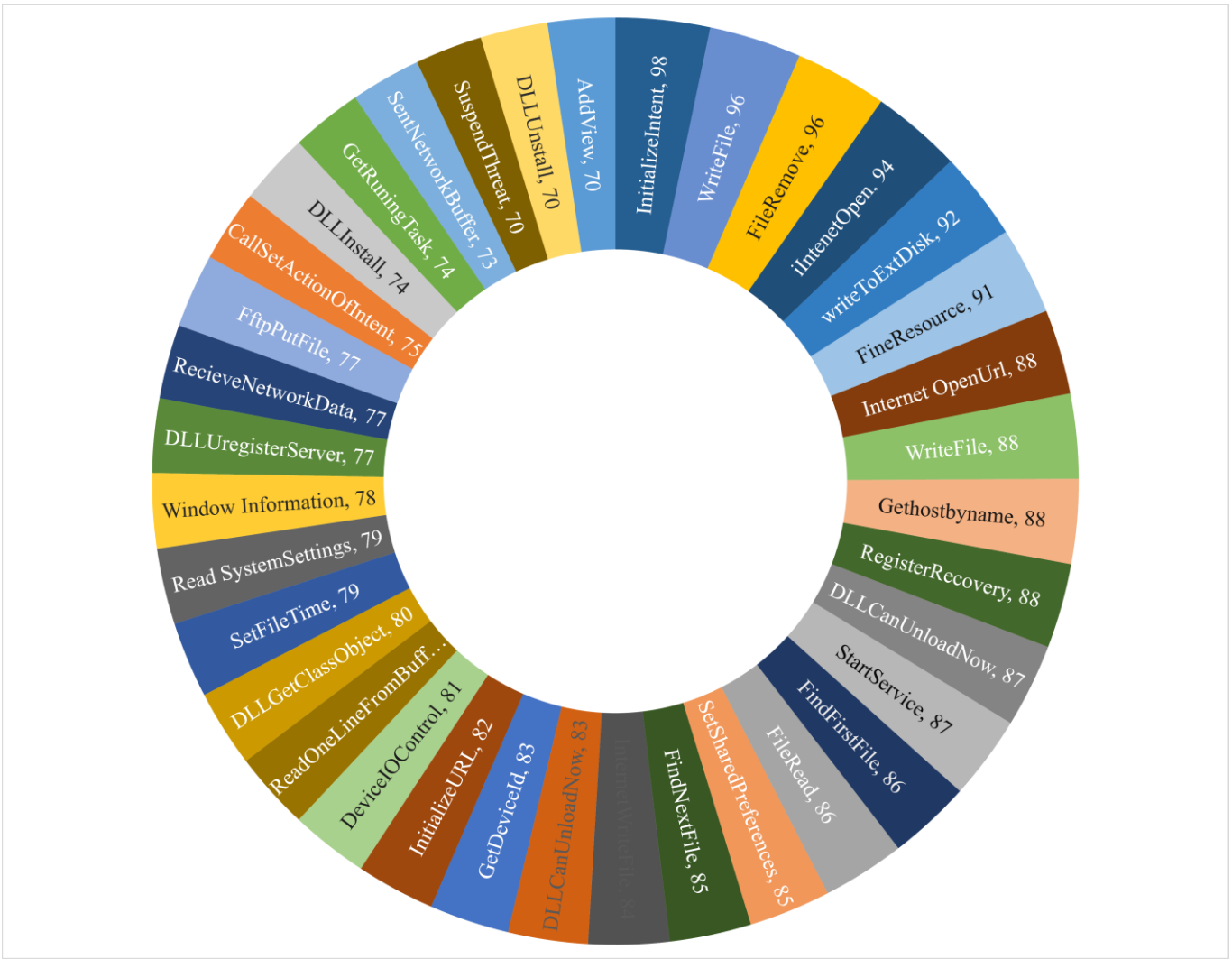

Fig. 12 Selected features

4.1. Proposed Novel Hybrid Features for Efficient Malware Classification

Relevant features for malware classification remain a challenge in malware defense. The study proposed a novel feature set based on the dynamic disassembly of malware datasets and extracted fine-grained features for the training and classification of malware and obfuscated malware samples. The novel feature set, as shown in figure 12, efficiently classified malware, including obfuscated samples, with a level of accuracy both before and after data augmentation methods.

Stated [29] that machine learning algorithms start and die on the bases of the features used in the training and testing. Thus, to have a good prediction, there is a need for efficient features that are relevant to the problem under investigation. Hence, with the variety of malware, selecting features highly representative of the problem domain under study is critical. Thus, with this feature set, broader malware families could be predicted or classified with high accuracy without bias and skewedness when in imbalanced environments.

4.2. Overcoming the 'Accuracy Paradox' in Imbalanced Malware Dataset

The use of an imbalanced dataset and using accuracy as a performance metric leads to the 'accuracy paradox', where the models are likely to be biased or skewed towards the majority class. Overcoming this requires the use of data augmentation or rebalancing techniques. With many highly imbalanced malware environments, using accuracy metrics as a performance measure without data augmentation results in the skewedness of malware detectors. We demonstrated and overcame this by classifying the malware initially without using the data augmentation and later with the use of the ADASYN technique. As depicted in Table 3, the models performed relatively better with the imbalanced dataset. This might be because the models are biased or skewed towards the majority class because the dataset is mildly skewed. As shown in Table 4, the accuracy metric before the data augmentation was slightly higher compared to the metric after the ADASYN technique, which presupposes some level of bias.

Table 3. Performance of models on the imbalance data before and after the ADASYN technique in percentages

Ensemble	Acc before ADASYN	Acc after ADASYN
RF	99.94	99.86
GB	99.89	99.81
XGB	99.95	99.87

Consequently, under the imbalanced data circumstances, it is good to use F1-score and the AUC metrics since these metrics provide stability in predicting the models. As depicted in Table 4, there was an appreciation of the F1-score and that of AUC after the application of the ADASYN technique. This appreciation in performance shows that precision should be used instead in the data imbalance environment, the use of AUC and or F1 score. As depicted in Table 5, the F1 score and AUC both appreciated before and after data augmentation.

Table 4. Performance metrics of the models before and after ADASYN technique

Ensemble	Accuracy B/A		Recall B/A		F1 Score B/A		AUC B/A	
RF	99.94	99.86	96	91	83	94	86.36	95.56
GB	99.89	99.81	70	69	72	83	84.82	94.02
XGB	99.95	99.87	91	87	85	96	89.39	98.59

**B/A. Before and After * All metrics in %*

4.3. Demonstrating the Resilience of Our Proposed Approach against Malware Obfuscation

Malware authors mainly use obfuscation techniques to change the binary sequence of malware to evade detection by signature-based detectors. To overcome this, we tested our proposed technique on the classification of malware before obfuscation and after obfuscation of the malware dataset. Figure 13 shows the performance of the models before obfuscation.

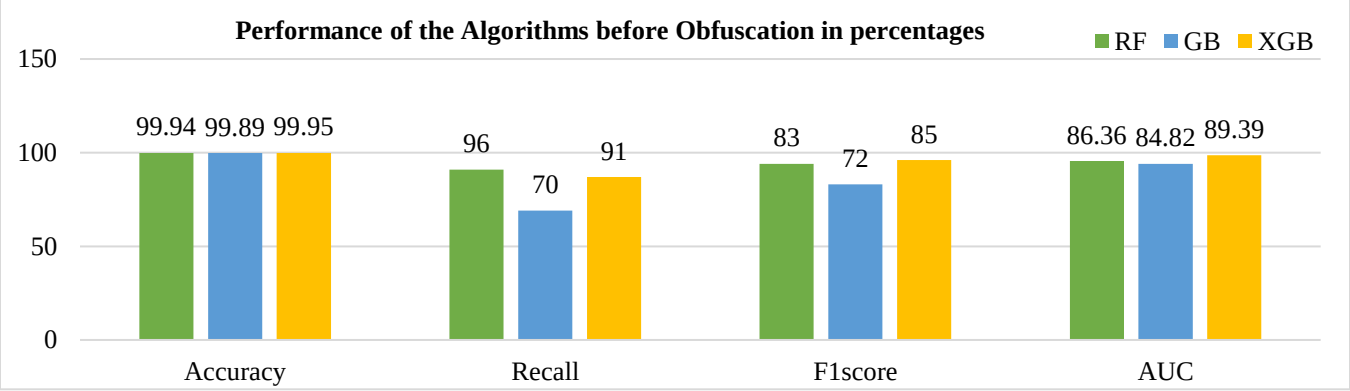

Fig. 13 Models performances before obfuscation

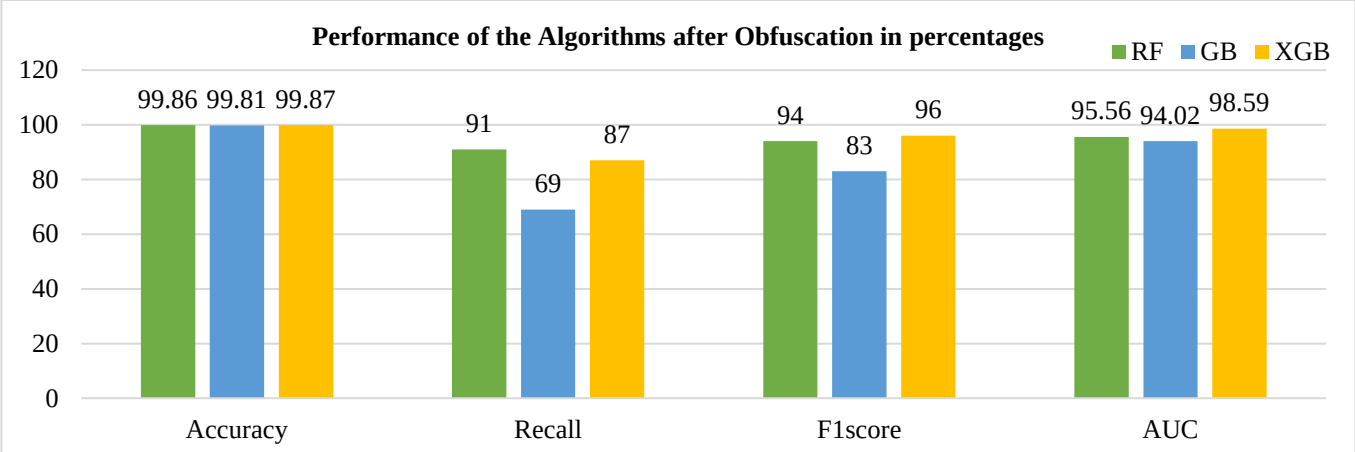

Fig. 14 Models performances after obfuscation

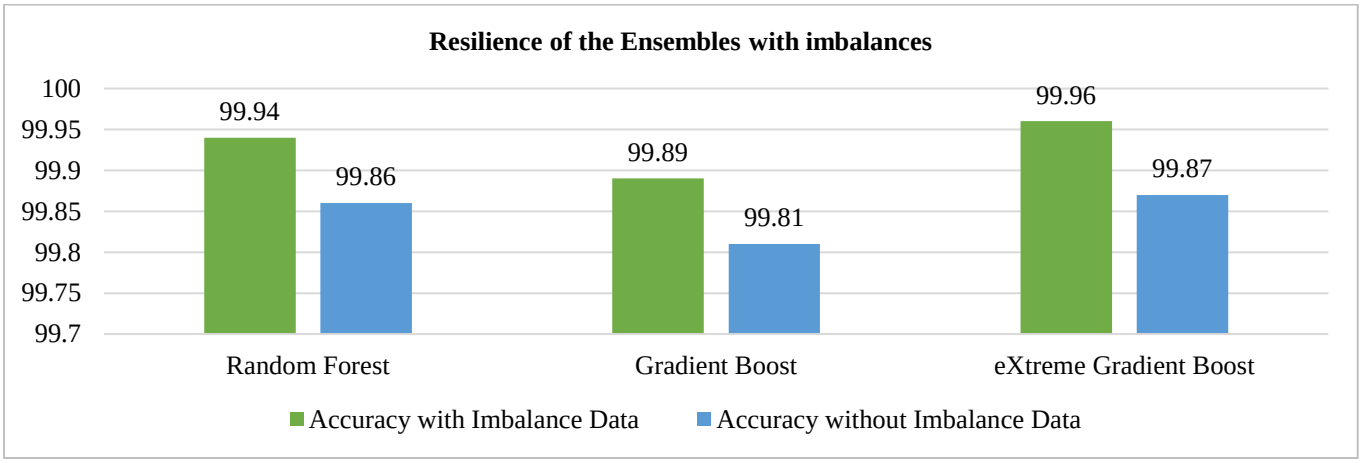

Fig. 15 Accuracy of the models with Imbalance data and augmented data

Similarly, when the malware samples were obfuscated, the performance of the models was again tested. As depicted in Figure 14, the performance of the models after obfuscating remains the same, demonstrating the resilience of the approach to malware obfuscation.

4.4. Overcoming Imbalance Malware Data using Ensemble Algorithmic Techniques

Conventional or traditional machine-learning algorithms are susceptible to data imbalances. They produce unsatisfactory classifiers as they show bias towards the majority classes and tend to treat the minority classes as noise or ignore them altogether. To overcome this, the use of undersampling and oversampling techniques has been recommended. The other approach is modifying the algorithm to make it appropriate for handling the imbalances using algorithmic techniques such as ensembles or committee-based learners [29]; 30]. This is mainly done to improve the performance of the techniques or classifiers. Thus, we combined ensemble algorithmic techniques with the data augmentation technique to improve the performance. As shown in Figure 15, the models performed relatively well.

5. Discussion of the Study

The study proposed and novel hybrid feature technique with ensemble and data augmentation for efficient and resilient detection of malware variant detection. The purpose was to propose novel features, overcome the accuracy paradox, demonstrate resilience against malware obfuscation and improve the performance of the imbalance dataset using an ensemble algorithmic technique. This section discusses the results of the study according to the research objectives.

5.1. Performance of the Proposed Homogeneous Hybrid Feature Set

Features or attributes that play a critical role in the machine-learning environment [29] are the foundation on which machine-learning algorithms start and die. In obtaining features, the need for efficiency reduced computational complexity, and resource requirements is critical. One identified gap or challenge in malware detection is relevant computational resource-efficient features. While using heterogeneous hybrid features seems to improve malware detection, the high computational resource requirements remain a problem. To overcome this, we proposed a novel

homogeneous hybrid feature extracted from dynamic malware disassembly. We extracted API call features and DLL function features and hybridized them for efficient and effective classification. The high performance of the models with the proposed features suggests the efficiency of the feature set in classifying malware before the application of the data augmentation and after. Similarly, the approach showed resilience against obfuscation by presenting high classification performance. This is consistent with [21], who did a similar study and reported an accuracy of 98.80%. However, their approach used features and malware datasets without the use of data augmentation, compared with our approach, where we used ADASYN data augmentation. The improved performance of our proposed technique demonstrates superior performance in both accuracy and resilience to obfuscated malware detection. The implication is that, with the proposed novel features, the detection and classification of obfuscated malware in imbalanced environments would improve the detection of new, novel, and zero-day malware.

5.2. Overcoming the Accuracy Paradox in Imbalance Class Problem

One of the major challenges facing machine learning is the class imbalance problem [30]. Class imbalance occurs when there is wide variation between the classes of items to be classified. It is classified as mild, moderate and extreme depending on the level of imbalance. The problem is that ML algorithms tend to be biased towards the majority sample of class and treat the minority sample or class as noise, which leads to misclassification or bias or skewedness of the models. Thus, when the malware dataset is imbalanced, using accuracy as a performance measure or metric without data augmentation creates biased accuracy. This is known as the 'accuracy paradox. To overcome the accuracy problem, the use of under-sampling, oversampling, and algorithmic techniques have been proposed. In undersampling, there is a decrease in the majority sample of the dataset until the number becomes near the minority class, whiles in oversampling, the numbers in the minority class are resampled to come near the majority class. Each of them has demerits, such as loss of vital information in the case of oversampling and synthetic data points created in the case of undersampling. Other techniques, such as the use of ensemble algorithmic techniques, have been explored. Thus, to overcome this, this study proposed and used the ADASYN technique as a means of overcoming the problem in that dataset by observing the accuracy of the models before and after the application of the technique. The results showed a reduced accuracy of the models after the ADASYN technique.

The reduced accuracy of the models after the application of the data augmentation demonstrates the imperfection of the accuracy metric when the malware dataset is imbalanced. This shows that the imbalance class creates some bias towards the majority class, causing the models to tend to ignore the minority or misclassify well in the minority class. The change in accuracy of the models is consistent with the assertion by [31], who suggested that imbalanced data tends to lead to the accuracy paradox, where the models tend to be biased towards the minority class and suggested the use of undersampling and oversampling techniques. Consequently, our use of the ADASYN technique on the training data rebalanced the dataset, resulting in a drop in the accuracy values. This alleviates the problem of data imbalances, which is prevalent in areas such as pilfering in the electricity industry, fraudulent bank transactions, the identification of uncommon or rare diseases, and the prediction of earthquakes.

5.3. Demonstrating Resilience against Obfuscated Malware

The growth in malware volumes, variety and complexity, continues challenging malware defenders. Malware attackers are using obfuscation techniques to evade detection from signature-based detection techniques. Anti-static techniques such as dead code insertion, code transposition, code reordering and others are used to evade detection. Advanced techniques such as encryption, polymorphic, metamorphic and oligomorphic techniques are used to create malware variants [16]. This variation in the byte sequences of the binaries makes it highly difficult for signature-based detectors leading to exploitation and exposure. To overcome this, the ML technique is needed to test in real-time malware obfuscation to ascertain the level of resilience of their models against different malware obfuscation methods.

To this end, we explored this by testing the models without obfuscation and after the samples were obfuscated. The result of the baseline classification before obfuscation and application of ADASYN demonstrates that the models performed relatively well in classifying malware, both before obfuscation and after application of the ADASYN technique. The results of the models remain the same in both cases. This means that the insertion or change in code sequence (obfuscation) had no impact on the performance of the models. The ability of the models to be able to classify malware that was obfuscated using both static and advanced obfuscation methods implies that the models are resilient, immune or resistant to malware obfuscation. Hence, when automated, the models can detect and classify known, unknown and zero-day vulnerabilities. Therefore, with the current malware variants making up about 80% of malware, having models that are resilient and immune to malware obfuscation is of the essence, and our approach presents a good tool for malware variant detection.

5.4. The Use of Algorithmic Ensemble Techniques in Improving Imbalance Data Problem

Apart from using resampling techniques to overcome the imbalance data problem, the alternative technique is to modify existing classifiers to make them appropriate for handling

imbalanced datasets. This modification involves the aggregation of weaker classifiers to form a stronger classifier, known as ensemble algorithms. The fundamental goal of ensemble techniques is to improve the performance of single classifiers [32]. These techniques can be bagging (Aggregation) with 'n' different boostrap training samples with replacement. They reduce overfitting and improve classification accuracy, improve stability, reduce variance and perform well in noise environments. Boosting, on the other hand, combines weak learners to create strong learners leading to improved prediction or classification. They are also very simple to implement, good at generalizations, and not prone to overfitting. Thus, to overcome the problem of class imbalance in the dataset, with sought to use these techniques to explore the performance. The result showed the resilience of the ensembles in handling the imbalanced data, as the margin of variation was minimal before the application of the data augmentation. This is consistent with the view of [32], who was of the view that ensembles always outperformed their base classifier and are stable in handling data. Similarly, [31]; [23] were of the view that, with most malware systems being in imbalanced settings, the use of data augmentation and ensembles are recommended. Thus, ensemble methods perform relatively well, which might not have been the case with conventional or traditional machine learning methods such as decision trees, logistic regression, and others that show bias and predict from the majority class. This relative performance might be due to ensemble algorithmic techniques improving the class imbalance problem. Besides, using the ensemble models in conjunction with the data augmentation provided stability and led to the classification performance.

Consequently, the proposed approach provided improved homogeneous features compared with those from the heterogeneous family with the added advantage of low computational complexity due to the homogeneity of features, improved accuracy due to the overcoming of the accuracy problem using the ADASYN technique, resilience and or immunity to malware obfuscation which were the gaps identified in the literature. Hence, it provides superior performance over cited literature.

6. Conclusion and Future Works

Malware attacks are increasing in volume, variety and complexity. The existing signature-based techniques have limitations in the face of the current malware environment. Though ML techniques are employed, their application has shown some gaps, such as poor and redundant features, data imbalance problems when using accuracy as a metric, the use of obfuscation techniques and the over-dependence of conventional machine-learning methods in imbalance dataset environments. Thus, this paper explored using an efficient hybrid feature technique with ensemble methods and a data augmentation technique to classify an imbalanced malware dataset. The purpose of the study was to propose an improved and efficient hybrid technique for the efficient classification of malware, to overcome malware obfuscation using the technique, to overcome the 'accuracy paradox' with imbalanced malware data using the data augmentation technique, and to show the resilience of the ensemble techniques with the data augmentation technique. The results showed the efficiency of the approach with XGB at 99.89% accuracy with ADASYN and AUC at 98.59% at detecting malware and obfuscated malware and improving the performance's reliability. Based on the study's achieved objectives, we conclude that our proposed approach provides improved fine-grained features for malware classification, overcomes the accuracy paradox, and demonstrates resilience against malware obfuscation. This has resulted in improving the detection and classification of known, unknown and variants of malware, including zero-day vulnerabilities. Thus, the study presents a huge potential for malware variant detection and classification. Notwithstanding the improvement in malware classification, less vigorous feature dimensionality reduction techniques and ensemble algorithms were used in the study. Future works will adopt the technique with robust feature dimensionality reduction techniques such as Principal Component Analysis (PCA) and compare the results of the homogeneous features with those of heterogeneous features using deep learning techniques such as Convulsionary Neural Networks (CNN).

Funding

Self-Financing/ Departmental support

Contributions

Azaabi Cletus: Work on the experiments, analysis, and write-up of the script.
Alex Akwasi Opoku: Review of equations and structure.
Benjamin Asubam Weyori: Review of the paper for experimental accuracy.

References

[1] Firas Shihab Ahmed et al., "Preliminary Analysis of Malware Detection in Opcode Sequences within IOT Environment," *Journal of Computer Science*, vol. 16, no. 9, pp. 1306-1318, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[2] M. Manjula, Venkatesh, and K. R. Venugopal, "Cyber Security Threats and Countermeasures using Machine Learning and Deep Learning Approaches: A Survey," *Journal of Computer Science*, vol. 19, no. 1, pp. 20-56, 2023. [CrossRef] [Publisher Link]

[3] Pedro Ramos Brandao**,** "Advanced Persistent Threats (APT)-Attribution-MICTIC Framework Extension," *Journal of Computer Science*, vol. 17, no. 5, no. 470-479, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[4] Faisal Alsattam et al., "Rule-Based Approach to Detect IOT Malicious Files," *Journal of Computer Science*, vol. 16, no. 9, pp. 1203-1211, 2020. [CrossRef] [Publisher Link]

[5] AV-Test Institute, Annual Malware statistics, 2023. [Online]. Available: https://www.av-test.org/en/statistics/malware

[6] DBIR Team, Verizon Data Breach Investigations Report 2022. Results and Analysis, 2023. [Online]. Available: https://www.verizon.com/business/resources/reports/dbir/

[7] Asma A. Alhashmi, Abdulbasit Darem, and Jemal H. Abawajy, "Taxonomy of Cybersecurity Awareness Delivery Methods: A Countermeasure for Phishing Threats," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 10, pp. 29-35, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[8] K. A. Monnappa, *Learning Malware Analysis: Explore the Concepts, Tools and the Techniques*, Packt Publishing, 2018. [Google Scholar]

[9] Hussein Aldawood, and Geofrey Skinner, "Reviewing Cyber Security Social Engineering Training and Awareness Programs-Pitfalls and Ongoing Issues," *Future Internet*, vol. 11, no. 3, p. 73, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[10] Joshua Saxe, and Hillary Sanders, *Malware Data Science: Attack Detection and Attribution*, Starch Press, Packt Publishing, 2018. [Google Scholar]

[11] Paul Joseph, and Jasmine Norman, "Systematic Memory Forensic Analysis of Ransomware using Digital Forensic Tools," *International Journal of Natural Computing Research*, vol. 9, no. 2, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[12] Berkant Düzgün et al., "New Datasets for Dynamic Malware Classification," *Cryptography and Security*, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[13] Yus Kamalrul Bin Mohamed Yunus, and Syahrulanuar Bin Ngah, "Review of Hybrid Analysis Technique for Malware Detection," *IOP Conference Series: Materials Science and Engineering*, vol. 769, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[14] Saleh Alyahyan, "*Machine Learning Ensemble Methods for Classifying Multi-Media Data*," A Thesis Submitted for the Degree of Doctor of Philosophy at the University of East Anglia, 2020. [Google Scholar] [Publisher Link]

[15] Azaabi Cletus, Alex Opoku, and Benjamin Weyory, "Exploring the Performance of Feature Dimensionality Reduction Technique Using Malware Dataset," *International Journal of Computer Science and Network Security*, vol. 22, no. 6, pp. 690-696, 2022. [Publisher Link]

[16] Jagsir Singh, and Jaswinder Singh, "Challenges of Malware Analysis: Malware Obfuscation Techniques," *International Journal of Information Security Science*, vol.7, no. 3, pp. 100-110, 2018. [Google Scholar] [Publisher Link]

[17] Emmanuel Masabo et al., "Improvement of Malware Classification Using Hybrid Feature Engineering," *SN Computer Science, Springer Nature*, vol. 1, no. 17, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[18] Hoda El Merabet, and Abderrahmane Hajraoui, "A Survey of Malware Detection Techniques Based on Machine Learning," *(IJACSA) International Journal of Advance Computer Science and Applications*, vol. 10, no. 1, pp. 366-373, 2019. [Google Scholar] [Publisher Link]

[19] Rami Sihwail, Khairuddin Omar, and Khairul Akram Zaninol Ari Sanad Al-Afghani, "Malware Detection Approach Based on Artefacts in Memory Image and Dynamic Analysis," *Applied Sciences*, vol. 9, no. 18, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[20] Danny Kim, "*Improving Existing Static and Dynamic Malware Detection Techniques with Intrusion-level Behaviour*," Theses and Dissertations from UMD, 2019. [Google Scholar] [Publisher Link]

[21] Cho Do Xuan et al., "Malicious URL Detection based on Machine Learning," *International Journal of Advance Computer Science and Application*, vol. 11, no. 1, 2020. [Google Scholar] [Publisher Link]

[22] Jinsu Kang, and Yoojae Won, "A Study on Variant Malware Detection Techniques using Static and Dynamic Features," *Journal of Information Processing Systems*, vol. 16, no. 4, pp. 882-895, 2020. [CrossRef] [Google Scholar] [Publisher Link].

[23] Hamid Darabian et al., "Detecting Cryptomining Malware: Deep Learning Approach for Static and Dynamic Analysis," *Journal of Grid Computing*, vol. 18, pp. 293-303, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[24] R.Surendiran, and K.Alagarsamy, "A Critical Approach for Intruder Detection in Mobile Devices," *SSRG International Journal of Computer Science and Engineering*, vol. 1, no. 4, pp. 6-14, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[25] Javier Bermejo Higuera et al., "Systematic Approach to Malware Analysis (SAMA)," *Applied Science*, vol. 10, no. 4, p. 1360, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[26] R C. Veena, and S H. Brahmananda, "A Significant Detection of APT using MD5 Hash Signature and Machine Learning Approach," *International Journal of Engineering Trends and Technology*, vol. 70, no. 4, pp. 95-106, 2022. [CrossRef] [Publisher Link]

[27] Sumit S. Lad, and Amol C. Adamuthe, "Improved Deep Learning Model for Static PE files malware Detection and Classification," *International Journal of Computer Network and Information Sciences*, vol. 14, no. 2, pp. 14-26, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[28] Pablo Duboue, *The Art of Feature Engineering, Essentials for Machine Learning*, Cambridge University Press, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[29] Ravi Diwakar, *Handling Imbalance Data with Imbalance-Learn in Python*, Data Science Blogathon, 2022. [Publisher Link]

[30] Alexsandro Parisi, *Hands-on Artificial Intelligence for Cybersecurity, Implement Smart AI System for Preventing Cyber-Attacks and*

Detecting Threats and Network Anomalies, Packt Publishing, 2020. [Google Scholar]

[31] Soma Halder, and Sinan Ozdemir, *Hands-on Machine Learning for Cybersecurity, Safeguard your Systems but Making Your Machines Intelligent Using the Python Ecosystem*, Packt Publishing, 2018. [Google Scholar]

[32] S. Sumathi et al., *Advance Decision Sciences Based on Deep Learning Algorithms: A Practical Approach Using Python*, Computer Science, Technology and Applications, Nova Science Publishers, 2020. [CrossRef] [Publisher Link]