Family Classification of Malicious Applications using Hybrid Analysis and Computationally Economical Machine Learning Techniques

Pushkar Kishore
CSE Dept.
NIT Rourkela,
Rourkela, Odisha, India.
monumit46@gmail.com

Swadhin Kumar Barisal
Siksha 'O' Anusandhan
Deemed to be University
Bhubaneswar, Odisha, India.
swadhinbarisal@soa.ac.in

Durga Prasad Mohapatra
CSE Dept.
NIT Rourkela,
Rourkela, Odisha, India.
durga@nitrkl.ac.in

***Abstract*—Most users utilize android smartphones for almost all activities. However, malicious attacks on these devices rose exponentially. Samples can be classified accurately, but earlier detection is challenging. So, we need a model that detects malicious applications before exploiting the data. This paper adopts computationally economical machine learning techniques to detect and determine the samples' families. Applications are analyzed to create static and dynamic datasets. Five data sampling techniques are used to fix the class imbalance. After data sampling, we apply four feature selection techniques to identify the most informative features. Then, four machine learning techniques are applied to detect malware and its family. In the case of static analysis, the highest mathews correlation coefficient (MCC) is 89% for malware classification, 86% for malware category classification, and 81% for malware family classification. In the case of dynamic analysis, the highest MCC is 81% for malware category classification and 62% for malware family classification. For hybrid analysis, we achieve 88% MCC for malware category classification and 82% for malware family determination. Our proposed model outperforms other state-of-the-art performance parameters named the area under curve, accuracy, F1-measure, and MCC.**

***Index Terms*—Android, Static analysis, Dynamic analysis, Data sampling, Feature selection**

I. Introduction

It is easier to access hardware, software, and IoT devices nowadays. The number of smartphones, smartwatches, and IoT devices has massively increased over the last four years [1]. A lot of these devices run on Android operating system (OS). Android applications consist of libraries and data. Android Manifest file contains the meta-information about the applications. "Res" directory contains pictures and applications' symbols. Other library resources contain non-aggregated assets [2]. Javed et al. [3] developed AlphaLogger that detected typed keys. These types of applications made cyber-attacks more challenging.

Android uses a module that verifies permission requested by applications. Wang et al. [2] discovered that android authorizations were divided into four categories. We consider the dataset having four malware categories: adware, ransomware, scareware, and SMS. As malware is evolving, we see the emergence of advanced detectors. There are two approaches for detecting malware: static analysis and dynamic analysis. In the case of static analysis, applications are analyzed without executions. However, for dynamic analysis, applications are executed in any environment. The surface area of the attacks is rising at a brisk and disturbing pace [4]. Many attacks will evolve in the future to attack android applications.

Some limitations in identifying android malware categories and families are:

1) Taheri et al. [5] and Tchakounte et al. [6] used static application programming interface (API) calls and permissions for detection. However, static analysis is insufficient for obfuscated samples.
2) Imtiaz et al. [7] used deep learning (DL) for detection. However, the malware detection, category classification and family classification rate are lower.
3) McLaughlin et al. [8] observed that the performance of the android malware detector and classifier decreased upon evaluating large-scaled datasets.
4) Sun et al. [9] observed that the training time might grow exponentially when the data grew linearly.

The primary objectives of our paper are:

1) Design a computationally economical system for detecting and identifying android malware, malware category and family on static, dynamic and hybrid layers.
2) Evaluate the effectiveness of our proposed system using the combination of data sampling, feature selection and machine learning (ML) techniques.
3) Compare ML techniques named naive bayes (NB), logistic regression (LR), support vector classifier (SVC) and random forest (RF).
4) Ensure a higher mathews correlation coefficient (MCC) for large-scale datasets.

The rest of the paper is organized as follows: Section II covers the related works, Section III presents the overview of the proposed model, Section IV covers experimental setup and results, Section V presents comparative analysis, and Section VI concludes with directions for future work.

II. Related Works

This section can be classified into three subsections: (1) API call based malware detection, (2) Intent based malware detection, and (3) Permission based malware detection.

A. API Call Based Android Malware Detection

Maiorca et al. [10] designed R-PackDroid for detecting ransomware. They used API packages that characterized applications context-free. As a result, R-PackDroid can detect obfuscated applications. Chan and Song [11] used a feature set containing permissions and API calls for static malware detection. However, they considered only 19 API calls and permissions. Taheri et al. [5] designed the CICAndMal2017 dataset, which included static permissions, intents and dynamic API calls. As a result, they enhanced the malware family classification performance. Zhang et al. [12] proposed a semantic-based approach that classified android malware using dependency graphs. They introduced graph-similarity metrics to find out similar application behaviors. A security-related weighted API graph for each application was created. Weighted contextual API dependency graph was extracted for feature set creation. They achieved 93% accurate anomaly detection on Genome dataset. Imtiaz et al. [7] proposed DeepAMD to detect real-world android malware. Their model can identify malware attacks on static and dynamic layers. DeepAMD is based on an artificial neural network and has 80.3% classification accuracy on dynamic layers.

B. Intent Based Malware Detection

Abuthawabeh et al. [13] designed a supervised model that enhanced malware detection accuracy using conversion-level traffic features. They employed the ensemble technique for selecting the most valuable features. They extracted conversion-level features with the PeerShark tool. They tested their model on a real-world dataset named CICAndMal2017 dataset. However, they tested static features only. Lashkari et al. [14] proposed a system that detected malicious or masquerading applications on mobile along with its specific family, i.e., adware. They used traffic features and classification techniques based on time, flow, and packets to identify malware families. Draper-Gil et al. [15] studied the effectiveness of flow-based time related features for detecting virtual private network (VPN) traffic. They categorized encrypted traffic into categories like browsing, streaming, etc. CICFlowMeter was used to generate and extract network traffic flow-level features. Feizollah et al. [16] evaluated the impact of explicit and implicit intents on identifying malicious applications. They proved that intents are better to encode malware's intentions than permissions. The detection rate was 83% with permissions, while 91% with intents. The detection rate ascended to 96% by merging intents and permissions.

C. Permission Based Android Malware Detection

Aung and Zaw [17] proposed an ML framework that detected malicious applications by monitoring permissions and event features. They applied K-means, random forest (RF), and decision tree (DT) techniques and obtained a 90% average detection rate. Huang et al. [18] extracted easily retrievable features from applications to detect malicious applications. They used techniques named Adaboost, naive bayes (NB), DT, and support vector machine (SVM). They guaranteed that a permission-based approach could achieve 81% detection accuracy. The above observations make it clear that network traffic flow-level features are helpful for detection. Canfora et al. [19] proposed a malware detector that worked on sequences of system calls. Specific system calls representing malicious behaviors like sending high premium rate short message service (SMS), botnet capabilities, etc., were implemented. They collected malicious applications from CICAndMal2017 with the alignment of permissions. The classification threshold was equal to the similarity score between the DNA of families and tested applications. Alzaylaee et al. [20] investigated ML-based detection using dynamic analysis on real devices. They had shown that several features could be effectively extracted from devices compared to emulators. An additional 24% of total applications were successfully analyzed on the phone. Some researchers applied deep learning techniques to identify the presence of malware in the android applications [21] but had lower malware detection, identification rate, dynamic analysis family detection, and dynamic analysis category detection. These limitations will rise and get complex as time passes.

III. Proposed Model

This section discusses about our proposed model and its description. Figure 1 illustrates our proposed framework for malware detection, category and family classification. As shown in Figure 1, our approach consists of multiple steps. First, samples are classified as malware or benign in the static layer. Next, the malware is classified into four categories and subclassified into 40 families. We apply several sampling techniques to balance class distribution. Next, several feature selection techniques identify the relevant features and remove the redundant ones. Then, classifiers are applied and compared. Finally, we evaluate the performance of classifiers using some performance metrics.

A. Static Layer Application's Label Detection

Applications are statically analyzed to create a dataset. In this case, the objective is to determine whether the android sample is malicious or benign. The dataset has an imbalanced class distribution. The distribution of benign samples to malicious samples is 0.74:0.26. There is much inconsistency between the number of benign and malicious samples. Dealing with an imbalanced dataset having an unequal number of classes will create bias in the result. We use four different types of sampling techniques to overcome the imbalance problem. The first technique is the synthetic minority over-sampling technique (SMOTE) [DS1], which is based on the combination of oversampling the majority and minority class [22]. The second technique is random undersampling (RNUNDER) [23] [DS2], which randomly removes instances from the majority class until a more balanced distribution is achieved. The third

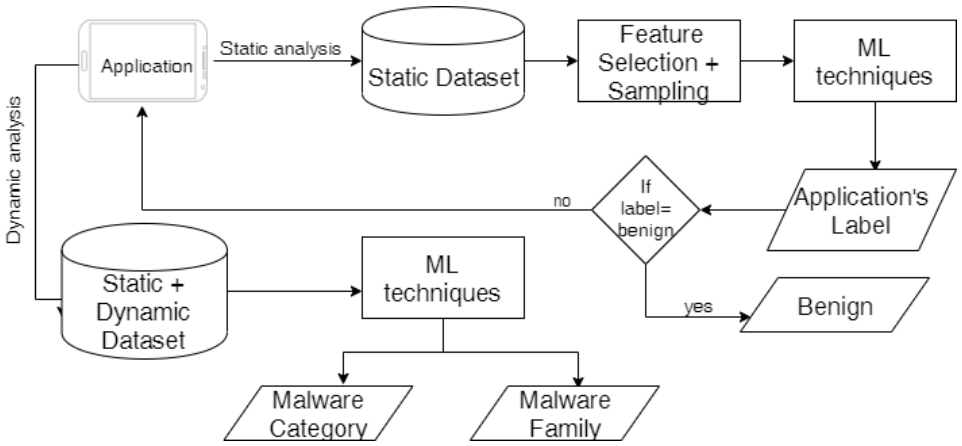

Fig. 1. Graphical representation of proposed android malware detection and identification approach

technique is random oversampling (RNDOVER) [24] [DS3], in which a random set of copies of minority class is added to the dataset. The fourth technique is tabular generative adversarial networks (tGAN) [25], [DS4], in which new instances can be created from actual data having numerical or categorical columns. We also consider the case, i.e., without sampling (WTOS) [DS5].

Then, we apply feature selection techniques. Liu et al. [26] surveyed several feature selection algorithms. We apply two feature selection techniques: Analysis of Variance (ANOVA) [FS1] and Kendall [FS2]. Apart from the above two, we apply one metaheuristic algorithm, the whale optimization algorithm [FS3]. Mirjalili et al. [27] surveyed many metaheuristic algorithms named particle swarm optimization (PSO), gravitational search algorithm (GSA), dolphin echolocation (DE), fast evolutionary programming (FEP), evolution strategy with covariance matrix adaptation (CMA-ES), genetic algorithm (GA) and evolution strategy (ES). They found that whale optimization is the best one. We also consider a case where all features are included [FS4].

After applying sampling and feature selection, computationally economical ML techniques named NB, LR, SVC, and RF are applied. Zhang et al. [28] compared predictive accuracy and area under curve (AUC) of 11 algorithms. Those computationally economical algorithms which examined the generalizability of our results are considered.

Few performance metrics are considered for evaluating the ML techniques' performance. Sokolova et al. [29] presented a detailed study of performance measures for the classification task. We use F1-measure, accuracy, AUC and MCC. If the ratio of classes is highly inconsistent, then F1-measure is preferred. F1-measure is the harmonic mean of precision and recall. AUC is used when the class distribution is balanced. Finally, MCC measures the performance of both positive and negative classes.

After finding the label of the applications using static analysis, we have two options. First, the malicious samples can be further sub-classified using static or dynamic analysis only. Second, the hybrid analysis can be used for the above objective.

B. Malware's Category Classification

There are four categories of malware in the dataset. The distribution between them is like 24% adware, 27% SMS, 24% ransomware and 25% scareware. First, data sampling techniques [DS1-DS5] are applied to balance the number of classes. After sampling, the feature selection techniques [FS1-FS4] are applied. Then, ML techniques are applied to find the static, dynamic and hybrid analysis performance.

C. Malware's Family Classification

There are 40 malware families in the dataset. The number of samples in any family ranges from 3 to 16. Since there is inconsistency in the number of samples belonging to a class, data sampling techniques [DS1-DS5] are applied. DS1's default parameters are changed since the number of neighbors is less than three sometimes for some training points. Example - simplocker has only one training instance; thus, *k_neighbors* parameter in DS1 is set as 1. After data sampling, feature selection techniques [FS1-FS4] are applied. At last, ML techniques are applied to find the performance of the static, dynamic and hybrid analysis.

TABLE I
MALWARE FAMILIES WITH THEIR COUNTS

Family	CS	Family	CS	Family	CS	Family	CS
Dowgin	6	Ewind	10	Feiwo	5	Gooligan	10
Kemoge	10	Mobidash	8	Selfmite	3	Shuanet	7
Youmi	6	Beanbot	10	Fakemart	9	Jifake	10
Mazarbot	10	Zsone	10	Charger	10	Jisut	8
koler	7	Lockerpin	3	Pletor	8	Porndroid	7
Ransom	10	Slocker	3	Svpeng	6	Wlocker	6
Spy.277	6	Defender	16	Av4And	10	Avpass	10
Fakeapp	4	Fakeav	10	Fakejob	8	FakeTB	7
Penetho	10	Vshield	9	Biige	3	Fakeinst	6
Nandrobx	10	Plankton	6	Smssnif	9	Fakenoti	12

TABLE II
PERFORMANCE PARAMETERS FOR STATIC BINARY DETECTION

DS	FS	F1-measure (%)				Accuracy (%)				AUC (%)				MCC (%)			
		NB	LR	SVC	RF	NB	LR	SVC	RF	NB	LR	SVC	RF	NB	LR	SVC	RF
WTOS	FS1	79	90	54	**92**	90	**95**	83	**95**	95	97	94	**98**	73	87	52	**88**
WTOS	FS2	27	51	85	58	77	83	75	84	89	90	80	88	25	49	0	53
WTOS	FS3	78	88	11	90	90	94	76	**95**	94	96	89	97	71	84	21	86
WTOS	FS4	77	90	7	91	90	**95**	76	**95**	95	97	90	97	71	87	17	**88**
SMOTE	FS1	84	87	81	**89**	92	93	91	**94**	96	97	95	98	79	83	75	**85**
SMOTE	FS2	72	73	50	72	82	83	75	83	91	91	87	89	62	64	14	62
SMOTE	FS3	78	87	66	86	87	93	86	93	95	96	93	98	70	82	59	82
SMOTE	FS4	79	86	65	**89**	87	92	86	**94**	96	97	94	**99**	72	81	60	**85**
RNDOVER	FS1	83	88	76	**90**	91	94	89	**95**	95	97	94	**98**	77	84	70	86
RNDOVER	FS2	72	74	7	71	82	83	76	82	90	91	66	88	63	65	17	60
RNDOVER	FS3	78	88	61	**90**	87	94	84	**95**	95	96	91	**98**	70	83	55	**87**
RNDOVER	FS4	82	88	50	**90**	90	94	82	**95**	96	97	91	**98**	76	84	48	**87**
RNDUNDER	FS1	84	86	63	85	92	**93**	84	92	95	96	92	**97**	78	82	54	80
RNDUNDER	FS2	70	72	85	71	81	82	75	82	90	90	15	88	60	62	0	60
RNDUNDER	FS3	73	87	31	86	82	**93**	79	92	95	96	89	**97**	64	82	34	81
RNDUNDER	FS4	75	87	26	**87**	84	**93**	79	**93**	95	**97**	89	**97**	67	**83**	34	**83**
TGAN	FS1	80	91	61	90	91	95	85	95	95	97	95	**98**	74	88	58	87
TGAN	FS2	24	20	85	48	69	78	74	82	66	92	49	87	6	29	0	46
TGAN	FS3	78	88	12	**91**	90	94	76	**96**	95	97	91	97	71	85	22	**89**
TGAN	FS4	83	88	12	88	92	94	76	94	94	96	91	97	77	84	22	84

IV. EXPERIMENTAL SETUP AND RESULTS

We execute all our experiments on Windows 10 professional 21H2. The CPU is Intel(R) Core(TM) i5-8250U with 16 GB RAM and NVIDIA GeForce 1060. Python 3.8 is used for the experiment.

The second version of the dataset CICAndMal2017 [30] named CICInvesAndMal2019 [5] is considered. The above dataset has numerous android malware families, permissions, intents, and API calls. The dataset contains other features like process logs, packages, log states, battery states, etc. The captured static features are symbolic, permission, intent, and components. In addition, dynamic features named API call and network flows are captured. Table I contains the malware families' counts (CS). Table I indicates that the maximum number of samples (16) is from the defender while the lowest (3) is from lockerpin, slocker, selfmite and biige. Only fifteen families have at least ten or more samples.

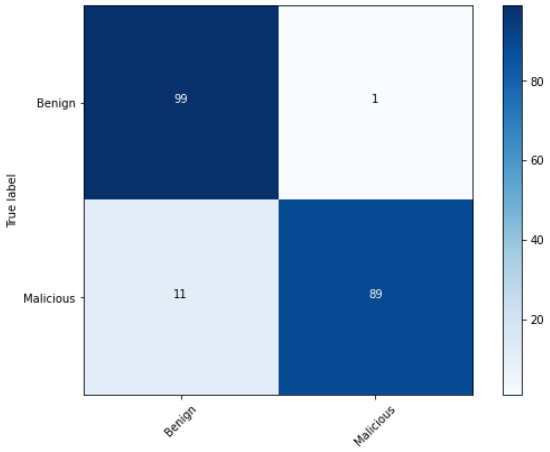

Fig. 2. Confusion matrix for static binary detection

A. Static Layer Application's Label Detection

Table II presents the evaluated performance metrics of static binary detection upon applying the combination of data sampling, feature selection and ML techniques. For DS5, the sampling is not done. Since the dataset is imbalanced, F1-measure and MCC are the best parameters. The highest F1-measure is 92% when FS1 and RF are used. The highest MCC is 88% when FS1 and RF are used. **Overall, FS1+RF is the best combination when no sampling is performed.** In the case of DS1, the sampling is performed. For a balanced dataset, AUC and MCC are the best parameters. The highest AUC is 99% when FS4+RF is used. The highest MCC is 85% whenever FS1+RF or FS4+RF are used. **All-around, FS4+RF is the best combination when SMOTE is used.** In the case of DS3, the highest AUC is 98% when FS1+RF, FS3+RF or FS4+RF are used. The highest MCC is 87% when FS3+RF or FS4+RF are used. **Thus, FS4+RF is the best combination for RNDOVER.** In the case of DS2, the highest AUC is 97% when FS1+RF, FS3+RF or FS4+RF are utilized. The highest MCC is 83% when FS4 and RF are used. **Hence, FS4+RF is the most suitable combination for RNDUNDER.** Lastly, for DS4, the highest AUC is 98% when FS1 and RF are used. The highest MCC jump to 89% when FS3 and RF are used. **Accordingly, FS3+RF is the most suitable combination for tGAN. Overall, the combination of whale optimization, tGAN and RF delivers the best result for static binary detection with 91% F1-measure, 96% accuracy, 97% AUC and 89% MCC**. Figure 2 depicts the best model's (RF) confusion matrix for static layer binary classification. Only 1% benign applications are misclassified, while 11% malicious samples are misclassified as benign.

TABLE III
PERFORMANCE PARAMETERS FOR STATIC CATEGORY CLASSIFICATION

DS	FS	F1-measure (%)				Accuracy (%)				AUC (%)				MCC (%)			
		NB	LR	SVC	RF	NB	LR	SVC	RF	NB	LR	SVC	RF	NB	LR	SVC	RF
WTOS	FS1	69	84	26	87	87	93	76	**94**	94	96	92	**98**	68	83	24	85
WTOS	FS2	17	50	17	61	74	81	75	83	81	89	79	90	-1	46	0	53
WTOS	FS3	45	82	20	86	79	92	75	**94**	87	96	91	97	40	80	13	85
WTOS	FS4	71	87	12	**89**	71	87	26	89	94	97	80	**98**	63	83	7	**86**
SMOTE	FS1	64	78	74	84	82	88	87	**92**	94	95	94	**98**	63	74	71	82
SMOTE	FS2	81	86	14	**89**	81	86	28	89	95	97	85	**98**	75	82	12	**85**
SMOTE	FS3	67	78	50	84	77	87	60	91	94	96	90	**98**	61	74	41	80
SMOTE	FS4	70	86	18	**89**	69	86	31	89	93	97	65	**98**	62	82	10	**85**
RNDOVER	FS1	64	82	40	85	85	92	81	93	93	96	93	**98**	63	81	47	84
RNDOVER	FS2	26	48	2	53	74	74	6	75	82	90	70	90	28	45	2	47
RNDOVER	FS3	42	80	32	87	75	91	79	**94**	85	96	91	97	45	77	42	**85**
RNDOVER	FS4	69	87	18	**88**	70	87	28	88	93	97	61	**98**	62	83	13	84
RNDUNDER	FS1	66	71	26	78	80	79	21	85	94	95	91	97	61	64	20	72
RNDUNDER	FS2	28	45	2	51	40	58	6	61	84	86	77	87	25	39	0	44
RNDUNDER	FS3	50	73	4	78	39	82	7	86	90	95	79	97	36	67	6	74
RNDUNDER	FS4	69	85	17	**88**	68	85	31	**88**	93	97	39	**98**	61	80	14	**83**
TGAN	FS1	70	88	49	88	71	88	51	88	93	97	89	**99**	62	84	39	83
TGAN	FS2	22	48	10	33	32	49	26	33	77	76	60	77	18	32	0	11
TGAN	FS3	81	86	14	**89**	81	86	28	**89**	95	97	85	98	75	82	12	**85**
TGAN	FS4	71	83	13	87	71	83	28	87	93	95	85	98	63	78	3	83

TABLE IV
PERFORMANCE PARAMETERS FOR DYNAMIC CATEGORY CLASSIFICATION

DS	FS	F1-measure (%)				Accuracy (%)				AUC (%)				MCC (%)			
		NB	LR	SVC	RF	NB	LR	SVC	RF	NB	LR	SVC	RF	NB	LR	SVC	RF
WTOS	FS1	12	59	11	**79**	32	59	28	**79**	50	86	50	95	0	47	0	**73**
WTOS	FS2	11	11	11	57	28	28	28	59	50	62	50	79	0	0	0	45
WTOS	FS3	8	44	11	67	20	44	28	67	50	70	50	89	0	25	0	56
WTOS	FS4	16	62	11	78	26	62	28	78	53	87	50	**96**	9	50	0	71
SMOTE	FS1	12	59	12	77	32	59	32	78	50	85	50	94	0	46	0	70
SMOTE	FS2	11	11	12	52	28	28	32	54	50	67	50	79	0	0	0	39
SMOTE	FS3	8	42	12	68	20	43	32	69	50	70	50	88	0	24	0	59
SMOTE	FS4	11	56	12	**82**	22	56	32	**81**	51	85	50	**97**	8	43	0	**75**
RNDOVER	FS1	12	63	11	73	32	63	28	74	50	88	50	94	0	51	0	66
RNDOVER	FS2	11	11	11	56	28	28	28	57	50	58	50	79	0	0	0	42
RNDOVER	FS3	11	40	11	68	21	41	28	68	51	72	50	89	2	22	0	58
RNDOVER	FS4	9	60	11	**81**	21	60	28	**80**	50	68	50	**97**	0	49	0	**74**
RNDUNDER	FS1	12	63	8	**76**	32	63	20	**77**	50	86	50	94	0	51	0	**70**
RNDUNDER	FS2	8	8	8	54	20	20	20	55	50	65	50	77	0	0	0	40
RNDUNDER	FS3	8	43	8	68	20	43	20	69	50	69	50	89	0	24	0	59
RNDUNDER	FS4	11	65	8	**76**	22	66	20	**77**	51	89	50	**96**	8	54	0	**70**
TGAN	FS1	11	29	8	82	28	33	20	82	50	74	50	95	0	14	0	76
TGAN	FS2	9	9	9	58	21	21	21	58	50	57	50	78	0	0	0	44
TGAN	FS3	11	28	9	65	28	30	21	66	50	71	50	88	0	12	0	54
TGAN	FS4	11	29	8	**85**	28	33	20	**85**	50	73	50	**97**	0	13	0	**81**

B. Malware's Category Classification

Table III presents the evaluated performance metrics for static category classification using the combination of data sampling, feature selection and ML techniques. For DS5, the highest F1-measure is 89% whenever FS4+RF is utilized. The highest MCC ascends to 86% whenever FS4+RF is used. **Overall, FS4+RF is the best combination when no sampling is employed.** For DS1, the highest AUC is 98% when FS1+RF, FS2+RF, FS3+RF or FS4+RF is used. The highest MCC is 85% when FS2+RF or FS4+RF is used. **Overall, FS4+RF is the most suitable combination for SMOTE.** In the case of DS3, the highest AUC is 98% when FS1+RF or FS4+RF is used. The highest MCC is 85% when FS3 and RF are used. **Thus, FS3+RF is the best combination for RNDOVER.** In the case of DS2, the highest AUC is 98% when FS4 and RF are used. The highest MCC is 83% when FS4 and RF are used. **Consequently, FS4+RF is the best combination for RNDUNDER.** Lastly, for DS4, the highest AUC is 99% when FS1+RF is used. The MCC extends to 85% when FS3 and RF are used. **Therefore, FS3+RF is the best combination for tGAN. Overall, the combination of whale optimization, RNDOVER and RF provides the most reasonable result for static malware category classification with 87% F1-measure, 94% accuracy, 97% AUC and 85% MCC**.

Table IV presents the estimated performance metrics for dynamic category classification using the combination of

TABLE V
PERFORMANCE PARAMETERS FOR STATIC FAMILY CLASSIFICATION

DS	FS	F1-measure (%)				Accuracy (%)				AUC (%)				MCC (%)			
		NB	LR	SVC	RF	NB	LR	SVC	RF	NB	LR	SVC	RF	NB	LR	SVC	RF
WTOS	FS1	53	79	9	79	42	79	6	92	89	97	64	97	42	78	11	80
WTOS	FS2	6	8	4	53	4	41	2	43	78	87	8	86	1	42	0	44
WTOS	FS3	43	79	6	**81**	32	78	3	92	84	97	86	**98**	33	78	6	79
WTOS	FS4	47	79	6	**81**	36	79	3	**93**	84	97	86	97	37	79	6	**81**
SMOTE	FS1	66	81	67	80	62	80	61	90	93	97	89	**98**	61	**80**	60	79
SMOTE	FS2	24	52	42	53	17	40	28	41	82	87	16	85	19	40	28	41
SMOTE	FS3	50	81	59	**82**	39	79	52	90	89	97	11	97	39	79	52	**80**
SMOTE	FS4	56	80	65	81	44	79	56	**91**	89	97	9	97	44	79	55	**80**
RNDOVER	FS1	53	80	13	**82**	42	79	8	90	89	97	45	**98**	42	79	13	79
RNDOVER	FS2	6	51	4	53	4	39	1	42	79	87	71	86	1	40	0	43
RNDOVER	FS3	43	80	6	80	32	79	2	**92**	84	97	76	97	33	79	1	79
RNDOVER	FS4	48	80	6	**82**	36	80	2	90	85	97	84	97	37	**80**	1	**80**
RNDUNDER	FS1	50	79	10	**81**	39	78	8	**90**	89	**97**	49	**97**	41	78	11	**80**
RNDUNDER	FS2	6	51	4	53	4	41	4	42	77	87	67	85	1	41	0	43
RNDUNDER	FS3	41	78	6	**81**	30	78	5	89	83	**97**	58	**97**	31	77	6	78
RNDUNDER	FS4	46	78	6	78	34	78	5	**90**	84	**97**	80	**97**	35	77	6	**80**
TGAN	FS1	48	71	9	79	36	67	7	**88**	86	96	87	97	37	66	10	**78**
TGAN	FS2	4	33	4	50	4	22	4	39	61	85	75	84	0	25	0	40
TGAN	FS3	13	78	6	**80**	10	78	4	**88**	69	97	87	**98**	11	77	6	**78**
TGAN	FS4	8	59	6	60	1	14	1	18	57	88	84	93	1	12	4	27

TABLE VI
PERFORMANCE PARAMETERS FOR DYNAMIC FAMILY CLASSIFICATION

DS	FS	F1-measure (%)				Accuracy (%)				AUC (%)				MCC (%)			
		NB	LR	SVC	RF	NB	LR	SVC	RF	NB	LR	SVC	RF	NB	LR	SVC	RF
WTOS	FS1	0	4	0	1	5	4	5	2	50	50	50	53	0	2	0	0
WTOS	FS2	0	0	0	3	5	5	5	4	50	51	50	56	0	0	0	2
WTOS	FS3	0	13	0	48	3	20	5	59	50	78	50	90	0	18	0	59
WTOS	FS4	0	14	0	**54**	3	21	5	**63**	50	80	50	**92**	0	19	0	**62**
SMOTE	FS1	0	5	0	1	2	5	3	2	50	50	50	50	0	3	0	0
SMOTE	FS2	0	0	0	3	2	2	3	3	50	50	50	50	0	0	0	1
SMOTE	FS3	0	13	0	46	3	16	3	58	50	50	50	80	0	14	0	57
SMOTE	FS4	0	19	0	**52**	3	23	3	**62**	50	50	50	**90**	0	22	0	**61**
RNDOVER	FS1	0	4	0	3	5	4	5	4	50	51	50	54	0	2	0	2
RNDOVER	FS2	0	0	0	4	3	3	5	6	50	50	50	54	0	0	0	3
RNDOVER	FS3	0	14	0	45	3	22	5	58	50	80	50	90	0	20	0	57
RNDOVER	FS4	0	15	0	**52**	3	21	5	**62**	50	50	50	**90**	0	19	0	**61**
RNDUNDER	FS1	0	4	0	5	3	5	2	5	50	49	50	57	0	3	0	3
RNDUNDER	FS2	0	0	0	3	3	3	2	4	50	48	50	55	0	0	0	2
RNDUNDER	FS3	0	13	0	41	3	20	2	53	50	79	50	87	0	18	0	52
RNDUNDER	FS4	0	17	0	**49**	3	23	2	**59**	50	79	50	**94**	0	22	0	**59**
TGAN	FS1	0	2	0	2	5	4	5	2	50	52	50	50	0	0	0	0
TGAN	FS2	0	0	0	3	5	5	5	5	50	50	50	57	0	0	0	2
TGAN	FS3	0	6	0	46	5	12	5	59	50	78	50	91	0	10	0	58
TGAN	FS4	0	16	0	**50**	5	25	5	**61**	50	81	50	**93**	0	23	0	**61**

sampling, feature selection and ML techniques. For DS5, the highest F1-measure is 79% whenever FS1 and RF are used. The highest MCC is 73% whenever FS1 and RF are used. **Overall, FS1+RF is the best combination with no sampling requirement.** For DS1, the highest AUC is 97% whenever FS4+RF is used. The highest MCC is 75% whenever FS4 is used with RF. **Overall, FS4+RF is the best combination for SMOTE.** In the case of DS3, the highest AUC is 97% when FS4+RF is used. The highest MCC is 74% when FS4+RF is used. **Thus, FS4+RF is the best combination for RNDOVER.** In the case of DS2, the highest AUC is 96% when FS4 and RF are used. The highest MCC is 70% when FS4+RF or FS1+RF are employed. **Therefore, FS4+RF is the most suitable combination for RNDUNDER.** Lastly, for DS4, the highest AUC is 97% when RF and FS4 are used. The highest MCC is 81% when FS4 and RF are used. **Accordingly, FS4+RF is the best combination for tGAN. Overall, the combination of no feature selection, tGAN and RF provides the most reasonable result for dynamic malware category classification with 85% F1-measure, 85% accuracy, 97% AUC and 81% MCC**.

On the hybrid dataset, we try two combinations named (TS4+FS4+RF) and (RNDOVER+FS3+RF). The best results are obtained from (RNDOVER+FS3+RF). The updated highest accuracy is 90%, F1-measure is 90%, AUC is 96%, and MCC is 88%. **Overall, MCC improves by 2% compared to**

MCC obtained for static category malware classification. Figure 3 depicts the best model's (RF) confusion matrix for

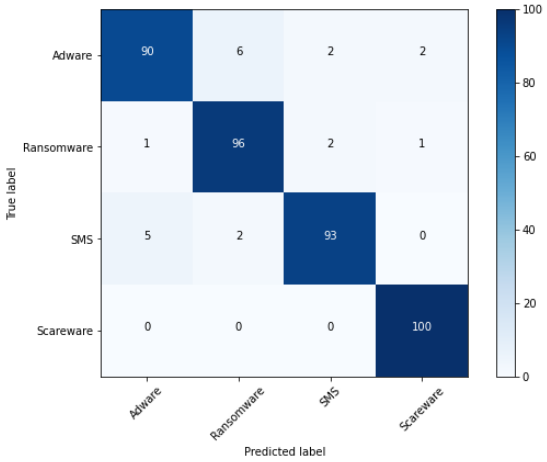

Fig. 3. Confusion matrix for hybrid category classification

the hybrid layer malware category classification. All scareware samples are correctly classified, while the lowest classification accuracy (90%) is for adware.

C. Malware's Family Classification

Table V presents the evaluated static family classification performance metrics upon applying the combination of sampling, feature selection and ML techniques. For DS5, the highest F1-measure is 81% whenever FS3+RF or FS4+RF is used. The highest MCC is 81% whenever FS4 and RF are used. **For no data sampling, FS4+RF is the best combination.** For DS1, the highest AUC is 98% when FS1 and RF are used. The highest MCC is 80% when FS3+RF or FS4+RF are used. **FS3+RF is the best combination for SMOTE.** In the case of DS3, the highest AUC is 98% when FS1 and RF are used. The highest MCC is 80% when FS4 and RF are used. **FS4+RF is the most suitable combination for RNDOVER.** In the case of DS2, the highest AUC is 97% when FS1+RF, FS3+RF or FS4+RF are used. The highest MCC is 80% when FS1+RF or FS4+RF is used. **FS1+RF is the best combination for RNDUNDER.** Lastly, for DS4, the highest AUC is 98% when RF and FS3 are used. The highest MCC is 78% when FS1+RF or FS3+RF is used. **Therefore, FS3+RF is the best combination for tGAN. Without feature selection and data sampling, RF provides the most reasonable result for static malware family classification with 81% F1-measure, 93% accuracy, 97% AUC and 81% MCC**.

Table VI presents the evaluated performance metrics for dynamic family classification upon applying the combination of data sampling, feature selection and ML techniques. For DS5, the highest F1-measure is 54% whenever FS4+RF is used. The highest MCC is 62% whenever FS4+RF is used. **For no data sampling, FS4+RF is the best combination.** For DS1, the highest AUC is 90% whenever FS4+RF was used. The highest MCC is 61% whenever FS4+RF is used. **Thus, FS4+RF is the best combination for SMOTE.** In the case of DS3, the highest AUC is 90% when FS4+RF is used. The highest MCC is 61% when FS4+RF is used. **Thus, FS4+RF is the best combination for RNDOVER.** In the case of DS2, the highest AUC is 94% when FS4+RF is used. The highest MCC is 59% when FS4+RF is used. **Therefore, FS4+RF is the most suitable combination for RNDUNDER.** Lastly, for DS4, the highest AUC is 93% when FS4+RF is used. The highest MCC is 61% when FS4+RF is used. **Hence, FS4+RF is the best combination for tGAN. Overall, the combination of no feature selection, no data sampling and RF provide the best result for dynamic malware family classification with 54% F1-measure, 63% accuracy, 92% AUC and 62% MCC**.

We infer from the above observations that FS4+WTOS+RF is the most suitable combination. We try the above combination on the hybrid dataset. The updated accuracy is 83%, F1-measure is 76%, AUC is 94%, and MCC is 82%. **Overall, MCC improves by 1% compared to MCC obtained for static malware family classification.**

V. Comparison with related state-of-the-arts

Table VII, VIII and IX present the comparison of our proposed methodology with existing state-of-the-art. In Table VII, highest F1-measure [92%] is obtained using DNN [7]. However, our proposed methodology has the highest values for all parameters. MCC is the best parameter among all, which is considered for performance comparison. There is a 5% improvement in MCC score from the second-best. In Table VIII, highest F1-measure [91%] and accuracy [91%] is obtained using DNN [7]. Our proposed one has over 4% improvement in MCC score compared to the second-best. In Table IX, highest F1-measure [80%] is obtained using DNN [7]. DT and KNN perform poorly due to the highly imbalanced dataset. However, RF performs much better due to the ensemble. Our proposed methodology surpasses the values of other techniques' performance parameters. Here, MCC is 3% higher than the second-best. **Overall, our proposed methodology achieves the highest MCC for binary detection, category and family classification using the proper combination of data sampling, feature selection and computationally economical technique (RF).**

TABLE VII
Performance comparison for static binary detection

Technique	F1-measure (%)	Accuracy (%)	AUC (%)	MCC (%)
RF [5]	86	92	95	81
KNN [14]	87	93	96	83
DT [14]	87	93	96	83
DNN [7]	**92**	93	**97**	84
Proposed	91	**96**	**97**	**89**

VI. Conclusion and Future Work

Attacks on the android operating system are rising; thus, a computationally efficient malware detector is needed. Applications are analyzed on three layers: static, dynamic and hybrid(static+dynamic). The combination of data sampling,

TABLE VIII
PERFORMANCE COMPARISON FOR CATEGORY CLASSIFICATION

Technique	F1-measure (%)	Accuracy (%)	AUC (%)	MCC (%)
RF [5]	82	81	95	75
DT [13]	77	78	94	70
KNN [14]	49	54	79	39
DNN [7]	**91**	**91**	**96**	84
Proposed	90	90	**96**	**88**

TABLE IX
PERFORMANCE COMPARISON FOR FAMILY CLASSIFICATION

Technique	F1-measure (%)	Accuracy (%)	AUC (%)	MCC (%)
RF [5]	60	43	86	44
KNN [14]	25	21	50	19
DT [14]	23	16	50	14
DNN [7]	**80**	80	92	79
Proposed	76	**83**	**94**	**82**

feature selection and techniques are applied to detect malware and classify its category and family. Upon experimentation, it is observed that a combination of whale optimization, tabular GAN and the random forest (RF) is best for static malware category detection. The highest malware category classification is achieved by replacing tabular GAN with RNDOVER and static with the hybrid. Similarly, RF is applied with no sampling and avoids feature selection for malware family classification.
In the future, we will test the combinations on more datasets. In addition, we will use other representations of the application's features to enhance the performance parameters.

REFERENCES

[1] I. Gartner, "Gartner says worldwide sales of smartphones recorded first ever decline during the fourth quarter of 2017," 2018.

[2] C. Wang, Q. Xu, X. Lin, and S. Liu, "Research on data mining of permissions mode for android malware detection," *Cluster Computing*, vol. 22, no. 6, pp. 13 337–13 350, 2019.

[3] A. R. Javed, M. O. Beg, M. Asim, T. Baker, and A. H. Al-Bayatti, "Alphalogger: Detecting motion-based side-channel attack using smartphone keystrokes," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–14, 2020.

[4] C. Iwendi, Z. Jalil, A. R. Javed, T. Reddy, R. Kaluri, G. Srivastava, and O. Jo, "Keysplitwatermark: Zero watermarking algorithm for software protection against cyber-attacks," *IEEE Access*, vol. 8, pp. 72 650–72 660, 2020.

[5] L. Taheri, A. F. A. Kadir, and A. H. Lashkari, "Extensible android malware detection and family classification using network-flows and api-calls," in *2019 International Carnahan Conference on Security Technology (ICCST)*. IEEE, 2019, pp. 1–8.

[6] F. Tchakounté, A. Djakene Wandala, and Y. Tiguiane, "Detection of android malware based on sequence alignment of permissions," *Int. J. Comput.(IJC)*, vol. 35, no. 1, pp. 26–36, 2019.

[7] S. I. Imtiaz, S. ur Rehman, A. R. Javed, Z. Jalil, X. Liu, and W. S. Alnumay, "Deepamd: Detection and identification of android malware using high-efficient deep artificial neural network," *Future Generation computer systems*, vol. 115, pp. 844–856, 2021.

[8] N. McLaughlin, J. Martinez del Rincon, B. Kang, S. Yerima, P. Miller, S. Sezer, Y. Safaei, E. Trickel, Z. Zhao, A. Doupé *et al.*, "Deep android malware detection," in *Proceedings of the seventh ACM on conference on data and application security and privacy*, 2017, pp. 301–308.

[9] B. Sun, T. Takahashi, T. Ban, and D. Inoue, "Detecting android malware and classifying its families in large-scale datasets," *ACM Transactions on Management Information Systems (TMIS)*, vol. 13, no. 2, pp. 1–21, 2021.

[10] D. Maiorca, F. Mercaldo, G. Giacinto, C. A. Visaggio, and F. Martinelli, "R-packdroid: Api package-based characterization and detection of mobile ransomware," in *Proceedings of the symposium on applied computing*, 2017, pp. 1718–1723.

[11] P. P. Chan and W.-K. Song, "Static detection of android malware by using permissions and api calls," in *2014 International Conference on Machine Learning and Cybernetics*, vol. 1. IEEE, 2014, pp. 82–87.

[12] M. Zhang, Y. Duan, H. Yin, and Z. Zhao, "Semantics-aware android malware classification using weighted contextual api dependency graphs," in *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, 2014, pp. 1105–1116.

[13] M. K. A. Abuthawabeh and K. W. Mahmoud, "Android malware detection and categorization based on conversation-level network traffic features," in *2019 International Arab Conference on Information Technology (ACIT)*. IEEE, 2019, pp. 42–47.

[14] A. H. Lashkari, A. F. A. Kadir, H. Gonzalez, K. F. Mbah, and A. A. Ghorbani, "Towards a network-based framework for android malware detection and characterization," in *2017 15th Annual conference on privacy, security and trust (PST)*. IEEE, 2017, pp. 233–23 309.

[15] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and vpn traffic using time-related," in *Proceedings of the 2nd international conference on information systems security and privacy (ICISSP)*, 2016, pp. 407–414.

[16] A. Feizollah, N. B. Anuar, R. Salleh, G. Suarez-Tangil, and S. Furnell, "Androdialysis: Analysis of android intent effectiveness in malware detection," *computers & security*, vol. 65, pp. 121–134, 2017.

[17] W. Z. Zarni Aung, "Permission-based android malware detection," *International Journal of Scientific & Technology Research*, vol. 2, no. 3, pp. 228–234, 2013.

[18] C.-Y. Huang, Y.-T. Tsai, and C.-H. Hsu, "Performance evaluation on permission-based detection for android malware," in *Advances in intelligent systems and applications-volume 2*. Springer, 2013, pp. 111–120.

[19] G. Canfora, E. Medvet, F. Mercaldo, and C. A. Visaggio, "Detecting android malware using sequences of system calls," in *Proceedings of the 3rd International Workshop on Software Development Lifecycle for Mobile*, 2015, pp. 13–20.

[20] M. K. Alzaylaee, S. Y. Yerima, and S. Sezer, "Emulator vs real phone: Android malware detection using machine learning," in *Proceedings of the 3rd ACM on International Workshop on Security and Privacy Analytics*, 2017, pp. 65–72.

[21] Z. Yuan, Y. Lu, and Y. Xue, "Droiddetector: android malware characterization and detection using deep learning," *Tsinghua Science and Technology*, vol. 21, no. 1, pp. 114–123, 2016.

[22] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[23] J. Prusa, T. M. Khoshgoftaar, D. J. Dittman, and A. Napolitano, "Using random undersampling to alleviate class imbalance on tweet sentiment data," in *2015 IEEE international conference on information reuse and integration*. IEEE, 2015, pp. 197–202.

[24] A. Moreo, A. Esuli, and F. Sebastiani, "Distributional random oversampling for imbalanced text classification," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 2016, pp. 805–808.

[25] L. Xu and K. Veeramachaneni, "Synthesizing tabular data using generative adversarial networks," *arXiv preprint arXiv:1811.11264*, 2018.

[26] H. Liu and L. Yu, "Toward integrating feature selection algorithms for classification and clustering," *IEEE Transactions on knowledge and data engineering*, vol. 17, no. 4, pp. 491–502, 2005.

[27] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in engineering software*, vol. 95, pp. 51–67, 2016.

[28] C. Zhang, C. Liu, X. Zhang, and G. Almpanidis, "An up-to-date comparison of state-of-the-art classification algorithms," *Expert Systems with Applications*, vol. 82, pp. 128–150, 2017.

[29] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information processing & management*, vol. 45, no. 4, pp. 427–437, 2009.

[30] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *computers & security*, vol. 31, no. 3, pp. 357–374, 2012.