# Behaviour-based Malware Detection in Mobile Android Platforms Using Machine Learning Algorithms

André Prata Ferreira[1], Chetna Gupta[2], Pedro R. M. Inácio[1,2], and Mário M. Freire[1,2*]
[1]Instituto de Telecomunicações, Universidade da Beira Interior, Covilhã, Portugal
{D1569, chetna.gupta}@ubi.pt
[2]Centro de Competências em Cloud Computing, Universidade da Beira Interior, Covilhã, Portugal
{inacio, mario}@di.ubi.pt

Received: July 2, 2021; Accepted: October 8, 2021; Published: December 31, 2021

**Abstract**

During the last few years, several approaches have been proposed for detection of Android malware Apps, each usually using its own dataset. Generating a representative Android malware dataset to evaluate malware detection approaches is a challenging task. Recently, the Canadian Institute for Cybersecurity released the CICAndMal2017 dataset, which includes recent and sophisticated Android samples spanning between five distinct categories: Adware, Ransomware, SMS malware, Scareware, and Benign. The best classification result obtained for this dataset was with a Precision of 95.3%, achieved with the Random Forest algorithm, using Permissions and Intents as static features. In this paper, we investigate the usage of nine machine learning algorithms to classify malware in the above mentioned dataset. The comparison of the obtained results is performed with the ones obtained with Random Forest, including performance evaluation (in terms of Precision, Recall, F-Measure, and Accuracy) and resource usage (in terms of execution time and CPU and memory consumption). Besides, we also investigate the use of a non-sliding Bag of System Calls algorithm with the above mentioned machine learning algorithms. It is shown that the Adaboost algorithm, using the Random Forest as a base estimator, leads to the best classification results with an Accuracy of 98.24%, a Precision of 99.31% (for malware), and an F1-Measure of 95.05% (for malware), at the cost of a larger execution time than Random Forest.

**Keywords**: Behaviour-Based Malware Detection, Static Malware Detection, Android Platforms, Machine Learnig Algorithms

## 1 Introduction

Cisco estimates that over 70% of the global population will have mobile connectivity by 2023 [1] and mobile devices using the Android operating system have been attracting a significant number of end-users. This trend lines up with the number of available devices increasingly attractive, either by its diversification of services, autonomy and capacity, whether by its use for personal, educational, commercial or professional. In fact, mobile devices are like microcomputer allowing users to store and access all necessary information anytime, anywhere. These devices are used to make and receive calls, send and receive messages, access personal mailbox and social networks, perform operations over bank accounts, and many times are used for dual authentication as well. It makes the use of mobile devices inevitable.

*Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, 12(4):62-88, Dec. 2021
DOI:*10.22667/JOWUA.2021.12.31.062*

*Corresponding author: Departamento de Informática, Faculdade de Engenharia, Universidade da Beira Interior, Rua Marques de Ávila e Bolama, 6201-001 Covilhã, Portugal, Web: `http://www.di.ubi.pt/~mario`, Tel: +351-275-242-081

According to recent report of Cisco [2] the number of mobile devices has grown to a larger extent not only among common users but also in industry/enterprise users.

As a consequence of their success, Android platforms have become one of the major target for attackers due to lack of expertise of end-users and malware detection has emerged as a growing area [3] in mobile platforms. During the last few years, several approaches have been proposed for detection of Android malware, each using their own dataset. Recently, the Canadian Institute for Cybersecurity released the CICAndMal2017 dataset [4], that includes recent and sophisticated Android samples until 2018 having samples spanning between five distinct categories: Adware, Ransomware, SMS malware, Scareware, and Benign. To the best of author's knowledge, to date the best classification result for detecting such malware in CICAndMal2017 dataset reached a Precision of 95.3% [4], using Random Forest algorithm with Permissions and Intents as static features.

In this paper, we investigate the usage of nine machine learning algorithms, namely AdaBoost, Logistic Regression, Decision Tree, Gaussian Naive Bayes, K-Nearest Neighbors, Multi-layer Perceptron, Multinomial Naive Bayes, Support Vector Machine, and Random Forest to classify malware in the above-mentioned dataset. The classification results obtained for these algorithms are compared with the results of Random Forest, including performance evaluation (in terms of Accuracy, Precision, Recall and F-Measure) and resource usage (in terms of execution time and CPU and memory consumption). Besides, we also investigate the use of a non-sliding Bag of System Calls algorithm with the above-mentioned machine learning algorithms.

The rest of the paper is organized in 4 Sections. Section 2 focus on the related work about the problem under study. Section 3 describes the dataset and classification approach, including how we extract, process and classify data. Section 4 presents the performance evaluation with classification results and consumption of used resources. Section 5 presents the main conclusions of this work.

## 2 Related Work

In this Section, we provide an overview of published research works related to the problem under study, focusing on works that used Permissions and Intents as features to perform Android malware detection. Feldman et al. [5] presented *Manilyzer*, a system that can be used to study the trends of Permission requests in malicious applications. It exploits information from manifest files and produces features vectors automatically. In their work, the authors used state-of-the-art machine learning algorithms (Naive Bayes, Support Vector Machine (SVM), K-Nearest Neighbors (KNN) and C4.5 Decision Tree) to perform the classification of the applications. Idrees and Rajarajan [6] proposed an approach that analyses the Permission and Intent patterns of Android applications. This approach was applied to samples collected from several sources, and the results were validated with the use of machine learning algorithms. Song et al. [7] proposed a static detection framework, consisting of four layers: the Message-Digest Algorithm 5 (MD5) values, the combination of malicious Permissions, the dangerous Permissions, and the dangerous Intent. The authors have also implemented a prototype system named ASE, and validated the feasibility, performance, and scalability of their proposed approach.

Idrees et al. [8] presented *Pindroid*, a framework based on Permissions and Intents with Ensemble methods for identifying Android malware applications. *Pindriod* framework is effective in detection of malware apps with 99.8% Accuracy. Idrees et al. [9] in their other work, presented *AndroPin*, an Andrioid malware detection tool that uses Intents and Permissions. According to the author, the proposed framework overcomes the limitation of stealthy techniques used by malware to exploit the usage of Permissions and Intents. Furthermore, Feizollah et al. [10] evaluated the effectiveness of using Intents as a feature to identify malicious applications. The authors in their work shows that Intents are semantically rich features and compared the results with other know features. Their work concludes

Table 1: Summary of the classification results obtained in related works. ACC: Accuracy, FP: False Positive, FN: False Negative, TPR: True Positive Rate, FPR: False Positive Rate, PRE: Precision, F1: F1-Score, REC: Recall, AUC: Area Under Curve.

| Author | Year | Dataset Description | ACC | FP | FN | TPR | FPR | PRE | F1 | REC | AUC |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Feldman et al.[5] | 2014 | Partial | ∼90.00% | 10.10% | 10.00% | - | - | - | - | - | - |
| Idrees and Rajarajan [6] | 2014 | Partial | - | - | - | - | - | - | - | - | - |
| Song et al. [7] | 2016 | No | ∼99.00% | - | - | - | - | - | - | - | - |
| Idrees et al. [8] | 2017 | Yes, poor | - | - | - | 99.80% | 1.10% | 99.80% | 99.70% | 99.70% | 99.80% |
| Idrees et al. [9] | 2017 | Yes, poor | - | - | - | 98.00% | 2.00% | - | - | - | - |
| Feizollah et al. [10] | 2017 | Partial | - | - | - | 95.50% | 4.40% | - | - | - | - |
| Taheri et al. [4] | 2019 | Yes, rich | - | - | - | - | - | 95.30% | - | - | - |
| Sangal and Verma [11] | 2020 | Yes, rich | 96.05% | - | - | 96.10% | 8.80% | 96.00% | 96.00% | 96.10% | 98.50% |
| Kouliaridis et al. [12] | 2020 | Partial | 81.40% | - | - | - | - | - | - | - | 88.20% |
| Khariwal et al. [13] | 2020 | Partial | 94.73% | - | - | - | - | - | - | - | - |

that Intents should be used in conjunction with other features for effective malware detection. Furthermore, they present results of Intents and Permissions used together to classify applications. Taheri et al. [4] examined Permissions and Intents as static features from their CICAndMal2017 dataset using their two-layer Android malware analyzer. In this two-layer Android Malware analyzer, the first layer is a Static-Based binary Malware classification, the second layer is a Dynamic-Based Malware Family Classification. Authors claim to succeed in achieving higher Precision in both static and dynamic-based malware classification. Sangal and Verma [11] used the CICInvesAndMal2019 dataset and performed Principal Component Analysis, a feature reduction technique. The authors classified the samples using well-known machine learning models and concluded that Random Forest was the best classifier to achieve satisfactory results.

Kouliaridis et al. [12] surveyed the major datasets namely, *Drebin, VirusShare, and AndroZoo* used by existing state-of-the-art malware detection techniques dated from 2012 to 2020. Additionally, the authors ranked the importance of app classification features on above-mentioned datasets, related to Permissions and Intents, by using the Information Gain algorithm. Additionally, Khariwal et al. [13] proposed an algorithm that uses Information Gain to rank Permissions and Intents to find the best set of Permissions and Intents that can be used to detect Malware with better Accuracy. The authors claim that the proposed algorithm can find the best set of Permissions and Intents by applying several machine learning algorithms. Their results shows that the best set consists of 37 features 20 Intents and 17 Permissions with Random Forest classifier as best machine learning algorithm.

Table 1 presents a summary of the classification results obtained by existing state-of-art methods using Permissions and Intents as features for malware classification. Some of these methods present good classification results, but have limitations such as poorly or partially (*Partial*, in Table 1) described malware dataset or malware dataset described in detail, but quite basic and/or marginally representative of current malware (marked in Table 1 as *Yes, poor*). In the column Dataset Description, we represented the case of complete/detailed description of the malware dataset where it is representative of current malware as *Yes, rich*. Missing values or not mentioned by the authors in the papers are represented in Table 1 by the symbol "-".

# 3 Dataset and Classification Approach

## 3.1 Dataset

The dataset used in this work is fetched from CICandMal2017, the Android Malware Dataset [4]. This database contains not only Android applications (APKs), but also CSV's and PCAP's files. These files contain network traffic features that were gathered in the scenario built for populating the database. For

this work, we used CICAndMal2017 Dataset [4] that has Benign and Malware applications. Malware is the short for malicious software and is software specially designed to harm users by stealing data or harm, damage or destroy systems [14]. Benign samples are harmless applications, the opposite of malicious. The CICAndMal2017 Dataset consists of five categories, one Benign and four Malignant. There are 1700 Benign applications, and 426 Malignant applications divided into four different categories: Adware –104 applications; Ransomware – 101 applications; Scareware – 112 applications; SMS Malware – 109 applications. Within the four different categories, there are 42 unique malware families.

### 3.2 Data Extraction

The static analysis consists in the analysis of an Android application, without executing the application, through reverse engineering, to verify their reliability and safety. In this way the application is scanned on a server and information such as hash, Permissions and source code are collected and processed by a classification algorithm that indicates whether it is or not a malware.

The static features of Android applications extracted in this work are based on the information provided by the manifest file. The application file APK is the package file format used by the Android operating system on mobile platform, that may contain the files, classes.dex, AndroidManifest.xml, directories and resource files. The plaintext file AndroidManifest.xml obtained through decompilation is a manifest file that defines the running information of the entire application.

Permissions and Intents were the characteristics chosen to define and characterize each of the applications. Permissions and Intents are defined in the manifest, Permissions refer to assigning or denying access to content. In Android 6.0+ versions these are requested to the user when some action needs Permission to be executed [15]. Intent is an asynchronous messaging object that can be used to request an action from other Android components [16].

With Permissions and Intents, it is possible to classify the samples as benign or malware. To access the manifest file, it is necessary to unpack the application. For this task the apktool [17] tool was used. Apktool is a Java-based tool that allows users to compile and decompile .APK files. After unpacking the applications, Permissions and Intents were extracted from the manifest of each application, as shown in Figure 1.

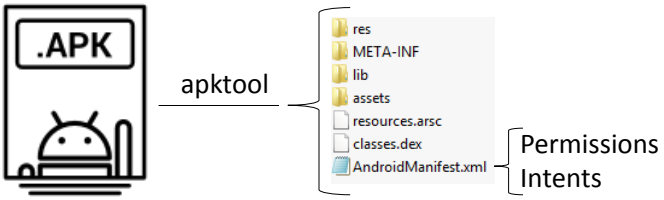


Figure 1: Data Extraction model of an APK.

### 3.3 Data processing

The data processing aims to improve the quality and organization of the data. For this, the data needed to build the comma-separated value file (CSV) was collected. We present an example using five Permissions. To perform network operations the manifest includes the Permissions INTERNET and ACCESS_NETWORK_STATE. WAKE_LOCK Permission allows the application to keep the device ON. READ_PHONE_STATE allows read only access to information regarding the Phone, for example cellular network information. ACCESS_WIFI_STATE Permission allows applications to consult information regarding Wi-Fi networks [18].

Two datasets were build:

- A CSV where the accounting of all Permissions and Intents for each of the applications was made. An example of this kind of datasets can be seen in Table 2.
- A CSV where it is validated if the Permission or Intent in question is present in the application. An example example of this kind of datasets can be seen in Table 3.

The CSV also contains the category where the application falls, benign or malware.

Table 2: Example of Accounted CSV Permissions.

| INTERNET | WAKE_LOCK | READ_PHONE_STATE | ACCESS_NETWORK_STATE | ACCESS_WIFI_STATE |
|---|---|---|---|---|
| 6 | 4 | 0 | 3 | 0 |

Table 3: Example of Normalized CSV Permissions.

| INTERNET | WAKE_LOCK | READ_PHONE_STATE | ACCESS_NETWORK_STATE | ACCESS_WIFI_STATE |
|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0 |

### 3.4 Experimental Environment and Classifiers

The experimental environment consists of a virtual machine (VM) with CentOS 7 operating system. It was installed using VMware Workstation Player on a Host with Windows 8.1 Pro. We present in Table 4 the components and characteristics of the experimental environment. Classifiers were implemented in this VM.

The experiments were run in this experimental environment. After the extraction process, explained in Section 3.2, we proceeded with the classification process. In order to allow a performance comparison with the method proposed by Taheri et al. [4], we used 60% of samples for training and the remaining 40% to test. The samples in the pre-processing step are randomly divided. Therefore, we performed fifty classification experiments for each algorithm in order to find the best result.

Table 4: Hardware characteristics.

| Resources | Characteristics |
|---|---|
| Host CPU | Intel® Core™ i7-5820k 3.30GHz |
| Host RAM | 32GB |
| HardDisk | 1TB SSD |
| Host Operating System | Windows 8.1 Pro |
| VM Resourses | 12 Cores, 8GB RAM |
| VM Operating System | Centos 7 |

The classification is performed on a CSV dataset using machine learning algorithms. During the classification, the measures about used RAM and CPU resources are collected and execution time is measured. The classification performance is measured using Accuracy, F1-Measure, Precision and Recall. The classification of applications as benign or malware is done using nine algorithms available at Scikit-learn library [19, 20]: AdaBoost (AB), Logistic Regression (LR), Decision Tree (DT), Gaussian Naive Bayes (GNB), K-Nearest Neighbors (KNN), Multi-layer Perceptron (MLP), Multinomial Naive Bayes (MNB), Support Vector Machine (SVM), and Random Forest (RF). In addition to the description

of these algorithms on the Scikit-learn web page, a detailed description of how these algorithms work can be found in textbooks, such as in [21]. In the implementation of each of these algorithms, the default settings provided by scikit-learn are used with some changes (optimizations) that are shown in Table 5. In this table, we replaced the setting values by the symbol "-", when the default settings values are used by algorithms without any change.

The Bag of System Calls (BoSC) algorithm is defined by Kang et al. in [22] and is used in this paper in order to investigate its impact on the classification results. The features in our work are represented by a vector Vi=(f1,f2,f3...fr) where r is the total number of distinct features. Given the size s of the bag of the BoSC algorithm, a new vector Vj=(f1,f2,f3...fs) is created. Since our feature vector consists of Permissions and Intents, we can represent it by V1=(p1,p2,p3...pm,i1 12,i3...in) where pk (k=1 ... m) represents a given Permission, and il (l=1...n) represents a given Intent and m is the total number of Permissions and n is the total number of Intents. When the BoSC is used, a new vector is created, V2=(p1,p2,p3...ps,i1,12,i3...is) where s represents the size of the bag of features that will be used. For classification using the non-sliding BoSC algorithm, the size of the bag is defined and then the classification is done using only the features in the Bag and without sliding the Bag over the full set of Permissions and Intents. In figure 2 we present an overview of the BoSC implementation in our work.

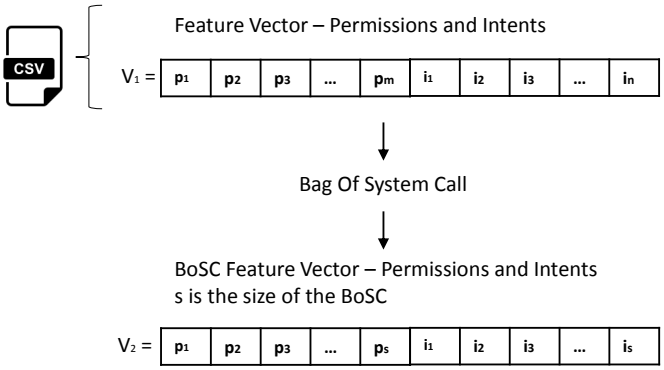


Figure 2: Schematic representation of the implementation of the Bag of System Calls algorithm.

Figure 3 outlines the four scenarios investigated in this paper, considering Accounted CSV and Normalized CSV datasets with and without the use of the BoSC algorithm.

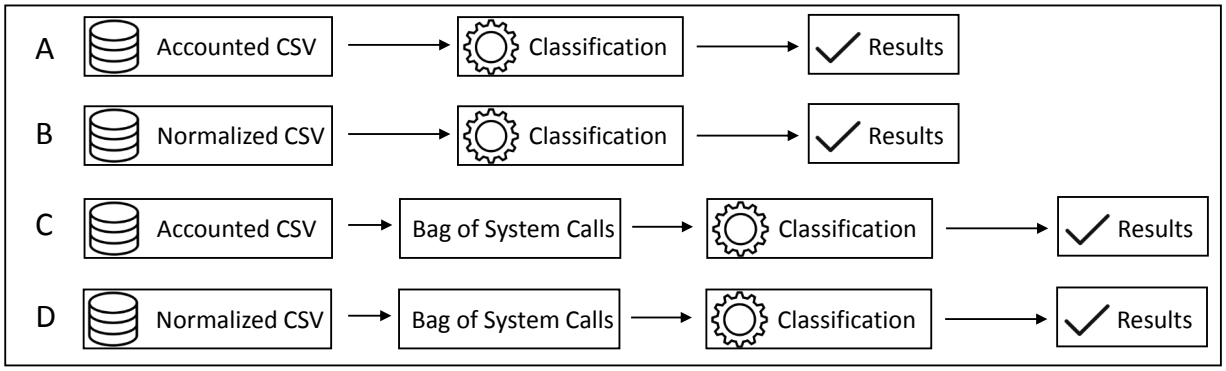


Figure 3: Classification scenarios considered along this paper.

Table 5: Settings of classification algorithms.

| Algorithm | Settings |
|---|---|
| AB | n_estimators=50,learning_rate=1, base_estimator=(default/DT/GNB/LR/MNB/RF) |
| DT | - |
| GNB | - |
| KNN | n_neighbors=2,3,5 and 10 |
| LR | max_iter=10000000000 |
| MLP | hidden_layer_sizes=(64,64,64), activation=(identity/logistic/tanh/relu), random_state=1, max_iter=2000<br>hidden_layer_sizes=(100,100,100), activation=(identity/logistic/tanh/relu), random_state=1, max_iter=2000 |
| MNB | - |
| RF | - |
| SVM | Kernel=(linear/poly/rbf/sigmoid), C=(0.1...1,1.5,2...30) |

# 4 Performance Evaluation

## 4.1 Performance Metrics and Resource Usage

For performance evaluation of the classification approaches applied to the CICandMal2017 dataset, as described in the Section 3, we use Accuracy (Ac), Precision (Pr), Recall (Re) and F1-Measure (F1) or F1-Score as defined in [23] and summarized in Table 6.

Table 6: Performance metrics and their definitions.

| | |
|---|---|
| $ACC = \frac{TP+TN}{TP+TN+FP+FN}.$ | Accuracy represents the proportion between the correctly classified cases and the total case. |
| $PRE = \frac{TP}{TP+FP}.$ | Precision is the proportion of the classified positive cases that are correctly classified. |
| $REC = \frac{TP}{TP+FN}.$ | Recall is the Proportion of positive cases that are correctly classified as positive. |
| $F1 = 2 * \frac{PRE * REC}{PRE + REC}.$ | F-Measure represents the harmonic mean of Precision and Recall. |

During each classification process we gather measures about the resources that are being used by the Virtual Machine (VM), namely CPU and RAM consumption and the execution time. The CPU usage and execution time are obtained using the /usr/bin/time command. The CPU usage is shown by process and in Linux can reach values above 100%, since the Time command presents the CPU usage as percentage of a single CPU and in multi-core systems the percentage of CPU usage is the sum of the percentage of all used cores. Measures of used RAM were collected using /usr/bin/ps_mem command. Using these metrics, it is possible to evaluate and compare the classification results and the corresponding resource consumption.

## 4.2 Classification of Accounted CSV

The Accounted CSV is the database gathered where each feature is counted the number of times that it appears. In tables 7, 8, 9 and figures 4 and 5, we present the results obtained with the Accounted CSV. Since the best result for malware classification in the CICAndMal2017 dataset is a Precision of 95.3%

[4], obtained for the Random Forest algorithm in binary classification using Permissions and Intents, this algorithm is herein used as our point of reference for comparison of the classification results.

The Random Forest algorithm for the Accounted CSV leads to an Accuracy of 97.06% and a Precision of 96.67% (malware), being these values only surpassed by the Adaboost algorithm using Random Forest (Adaboost-RF) as base estimator. For the Adaboost-RF we obtained an Accuracy of 97.76% and a Precision of 98.7% (malware). However, Adaboost-RF requires a larger execution time (94.485 s) than RF (11.964 s). Figures 4 and 5 illustrate the impact of the self C parameter of Support Vector Machine (SVM) on the classification Accuracy for Accounted CSV. We also show in table 9 the best classification results obtained for the best value of the C parameter for each Kernel.

Table 7: Classification results for Accounted CSV using DT, GNB, KNN, LR, MNB, RF, and AB algorithms.

| Algorithm | Ex. Time (s) | RAM (MB) | CPU (%) | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|
| DT | 9.8568 | 330.63 | 114.04 | Benign | 97.93 | 95.95 | 96.93 | 95.06 |
| | | | | Malware | 83.82 | 91.19 | 87.35 | |
| GNB | 10.1524 | 316.98 | 114.06 | Benign | 96.51 | 54.89 | 69.98 | 63.76 |
| | | | | Malware | 38.28 | 93.37 | 54.30 | |
| KNN-2 | 9.6302 | 310.23 | 146.72 | Benign | 95.35 | 99.56 | 97.41 | 95.76 |
| | | | | Malware | 97.86 | 80.59 | 88.39 | |
| KNN-3 | 9.6348 | 281.31 | 145.98 | Benign | 97.77 | 97.05 | 97.41 | 95.88 |
| | | | | Malware | 88.70 | 91.28 | 89.97 | |
| KNN-5 | 9.6612 | 329.08 | 145.9 | Benign | 93.80 | 98.93 | 96.30 | 94.12 |
| | | | | Malware | 95.54 | 77.72 | 85.71 | |
| KNN-10 | 9.7176 | 329.81 | 146.8 | Benign | 93.13 | 99.14 | 96.04 | 93.29 |
| | | | | Malware | 94.44 | 66.67 | 78.16 | |
| LR | 12.2876 | 398.24 | 287.92 | Benign | 96.70 | 98.83 | 97.76 | 96.35 |
| | | | | Malware | 94.74 | 86.23 | 90.28 | |
| MNB | 9.7072 | 280.3 | 123.16 | Benign | 93.89 | 97.43 | 95.63 | 93.06 |
| | | | | Malware | 89.57 | 77.66 | 83.19 | |
| RF | 11.9644 | 451.91 | 111.86 | Benign | 97.14 | 99.27 | 98.19 | 97.06 |
| | | | | Malware | 96.67 | 87.88 | 92.06 | |
| AB | 20.936 | 73.76 | 104.2 | Benign | 96.73 | 97.70 | 97.22 | 95.41 |
| | | | | Malware | 89.04 | 84.97 | 86.96 | |
| AB-DT | 16.4304 | 51.13 | 109.14 | Benign | 96.84 | 97.26 | 97.05 | 95.18 |
| | | | | Malware | 87.66 | 85.99 | 86.82 | |
| AB-GNB | 73.9054 | 878.83 | 101.76 | Benign | 94.48 | 84.59 | 89.26 | 83.53 |
| | | | | Malware | 54.70 | 79.01 | 64.65 | |
| AB-LR | 83.9316 | 928.92 | 395.84 | Benign | 95.84 | 98.38 | 97.09 | 95.29 |
| | | | | Malware | 92.81 | 83.04 | 87.65 | |
| AB-MNB | 71.6894 | 845.45 | 295.08 | Benign | 83.57 | 95.28 | 89.04 | 81.29 |
| | | | | Malware | 58.44 | 26.16 | 36.14 | |
| AB-RF | 94.4852 | 852.02 | 100.98 | Benign | 97.56 | 99.71 | 98.62 | 97.76 |
| | | | | Malware | 98.70 | 89.94 | 94.12 | |

Table 8: Classification results for Accounted CSV using Multi-layer Perceptron.

| Algorithm | Activation | H.L. | Ex. Time (s) | RAM (MB) | CPU (%) | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|---|---|
| MLP | Identity | 64 | 82.966 | 941.98 | 808.78 | Benign | 96.38 | 97.94 | 97.15 | 95.41 |
| | | | | | | Malware | 91.25 | 85.38 | 88.22 | |
| | | 100 | 153.669 | 991.68 | 852.26 | Benign | 96.78 | 97.06 | 96.92 | 95.06 |
| | | | | | | Malware | 88.02 | 86.98 | 87.50 | |
| | Logistic | 64 | 108.4408 | 971.37 | 865.7 | Benign | 97.49 | 97.63 | 97.56 | 96.12 |
| | | | | | | Malware | 90.80 | 90.29 | 90.54 | |
| | | 100 | 210.6308 | 1009.29 | 858.44 | Benign | 97.87 | 95.97 | 96.91 | 95.18 |
| | | | | | | Malware | 86.01 | 92.22 | 89.01 | |
| | Relu | 64 | 57.6158 | 897.16 | 783.46 | Benign | 97.53 | 97.10 | 97.31 | 95.65 |
| | | | | | | Malware | 87.73 | 89.38 | 88.54 | |
| | | 100 | 93.3772 | 955.25 | 813.64 | Benign | 96.68 | 98.10 | 97.38 | 95.76 |
| | | | | | | Malware | 91.72 | 86.23 | 88.89 | |
| | Tahn | 64 | 64.3188 | 905.46 | 802.4 | Benign | 96.68 | 98.10 | 97.38 | 95.41 |
| | | | | | | Malware | 91.72 | 86.23 | 88.89 | |
| | | 100 | 154.0948 | 992.57 | 851.8 | Benign | 97.01 | 96.14 | 96.57 | 94.59 |
| | | | | | | Malware | 85.71 | 88.64 | 87.15 | |

Table 9: Classification results for Accounted CSV using Support Vector Machine (Best Accuracy for each kernel).

| Algorithm | Ex. Time (s) | RAM (MB) | CPU (%) | Kernel | C | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|---|---|
| SVM | 24.18 | 947.09 | 600.97 | Linear | 4 | Benign | 97.98 | 97.84 | 97.91 | 96.59 |
| | | | | | | Malware | 90.45 | 91.03 | 90.73 | |
| | 50.13 | 442.98 | 807.15 | RBF | 29 | Benign | 96.22 | 98.71 | 97.45 | 95.76 |
| | | | | | | Malware | 93.33 | 82.35 | 87.50 | |
| | 27.74 | 954.93 | 618.93 | Poly | 30 | Benign | 85.05 | 99.85 | 91.86 | 85.88 |
| | | | | | | Malware | 98.15 | 30.81 | 46.90 | |
| | 26.74 | 641.82 | 618.93 | Sigmoid | 3 | Benign | 92.67 | 98.24 | 95.37 | 92.35 |
| | | | | | | Malware | 90.55 | 68.45 | 77.97 | |

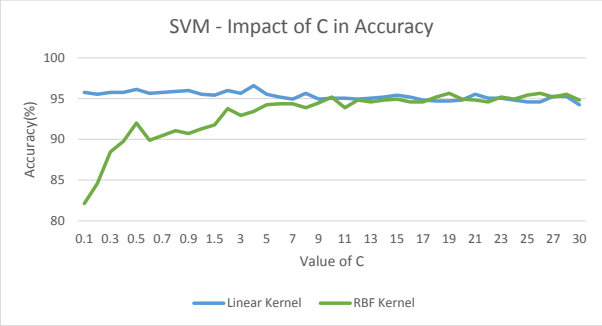


Figure 4: SVM classification results for Accounted CSV with Linear and RBF Kernels.

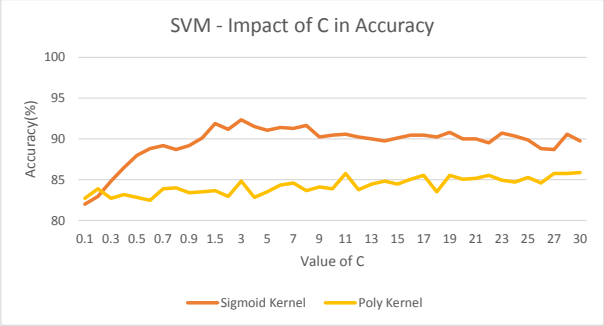


Figure 5: SVM classification results for Accounted CSV with Sigmoid and Poly Kernels.

### 4.3 Classification of Normalized CSV

The Normalized CSV is the database gathered where each feature is only counted once, if it appears in the sample or not. In tables 10, 11, 12 and figures 6 and 7 we present the classification results obtained with the Normalized CSV. We got slightly better results in terms of Accuracy (between 0.12% and 1.88%) using the Normalized CSV than the Accounted CSV, for the following algorithms: KNN2, KNN5, RF, AB, AB-DT, AB-GNB, AB-LR and AB-RF, MLP (except using Logistic activation function with hidden layer of size 64), SVM (except for Linear Kernel with C value of 0.4). For Normalized CSV, Random Forest and Adaboost-RF increased the Accuracy in 0.35% and 0.48%, respectively, leading to an Accuracy of 97.41% for RF and 98.24% to Adaboost-RF. Despite the classification improvement, the differences between the resource utilization of these two algorithms remains similar.

Figures 6 and 7 illustrate the impact of the self C parameter of Support Vector Machine (SVM) on the classification Accuracy for Normalized CSV. We also show in table 12 the best classification results obtained for the best value of the C parameter for each Kernel.

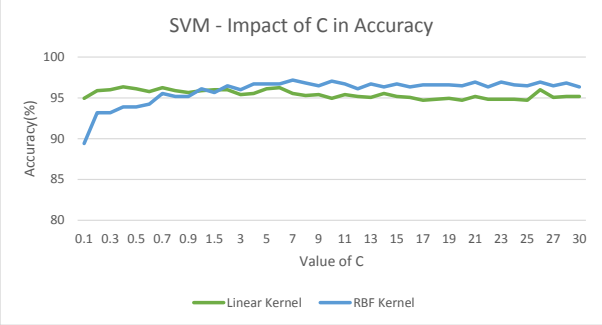


Figure 6: SVM classification results for Normalized CSV with Linear and RBF Kernels.

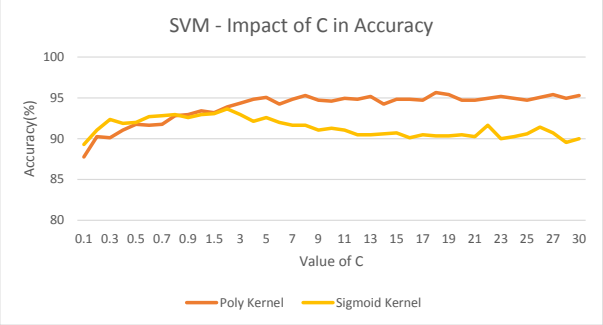


Figure 7: SVM classification results for Normalized CSV with Sigmoid and Poly Kernels.

### 4.4 Classification Using the Bag of System Calls Algorithm

The Bag of System Calls (BoSC) algorithm allows to use each time a part and not all classification characteristics. As such, bags with sizes of 500, 750 and 1000 were used. This means that, for example, in the bag of 500 we use only 500 Permissions and 500 Intents as characteristics to carry out the classification procedure. We compare in each Section the results obtained with the ones obtained in Sections 4.2 and 4.3 for Accounted CSV and Normalized CSV.

#### 4.4.1 Classification Using the Bag of System Calls with Size of 500 for Accounted CSV

Considering the results obtained in table 13 and comparing with our first test scenario (Section 4.2) we got nine increases of Accuracy in a total of 15 classifications. We highlight the increase of 26.36% and 8.82% for the algorithms GNB and Adaboost-GNB respectively, being them the best improvements of this subset of results. We also observed an increase of the Accuracy by 0.12% for Random Forest by using this technique. For MLP results we can obverse three improvements, two with Relu activation function for both defined Hidden Layer and with Activation Function Tahn with a Hidden Layer of size 100. These improvements are of 0.47%, 0.83% and 0.35% respectively, as we can see in table 14. The results for SVM are presented in table 15 and in figures 8 and 9, where we can see an increase of the Accuracy of Poly and Sigmoid Kernels from 85.88% to 88.82% and from 92.35% to 92.59%, respectively.

Table 10: Classification results for Normalized CSV using DT, GNB, KNN, LR, MNB, RF, and AB algorithms.

| Algorithm | Ex. Time (s) | RAM (MB) | CPU (%) | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|
| DT | 9.244 | 315.44 | 115.3 | Benign | 95.70 | 97.58 | 96.63 | 94.71 |
| | | | | Malware | 90.86 | 84.57 | 87.60 | |
| GNB | 9.7386 | 362.54 | 114.48 | Benign | 97.33 | 54.57 | 69.93 | 63.18 |
| | | | | Malware | 36.34 | 94.54 | 52.50 | |
| KNN2 | 9.3562 | 299.79 | 145.76 | Benign | 96.15 | 98.97 | 97.54 | 96.00 |
| | | | | Malware | 95.27 | 83.93 | 89.24 | |
| KNN3 | 9.4094 | 328.67 | 145.3 | Benign | 96.74 | 96.59 | 96.66 | 94.71 |
| | | | | Malware | 86.93 | 87.43 | 87.18 | |
| KNN5 | 9.4 | 311.15 | 144.84 | Benign | 96.80 | 96.52 | 96.66 | 94.59 |
| | | | | Malware | 85.19 | 86.25 | 85.71 | |
| KNN10 | 9.397 | 309.81 | 146.74 | Benign | 94.09 | 97.85 | 95.93 | 93.18 |
| | | | | Malware | 87.80 | 71.52 | 78.83 | |
| LR | 11.2456 | 385.46 | 262.76 | Benign | 96.19 | 98.84 | 97.49 | 95.88 |
| | | | | Malware | 94.37 | 83.23 | 88.45 | |
| MNB | 9.2968 | 314.8 | 123.98 | Benign | 93.00 | 96.37 | 94.65 | 91.18 |
| | | | | Malware | 81.62 | 68.94 | 74.75 | |
| RF | 11.6306 | 458.95 | 111.96 | Benign | 97.71 | 99.13 | 98.41 | 97.41 |
| | | | | Malware | 96.03 | 90.06 | 92.95 | |
| AB | 16.381 | 573.08 | 109.64 | Benign | 96.22 | 98.37 | 97.28 | 95.65 |
| | | | | Malware | 93.21 | 85.31 | 89.09 | |
| AB-DT | 16.3624 | 577.04 | 109.6 | Benign | 96.97 | 97.53 | 97.25 | 95.53 |
| | | | | Malware | 89.17 | 86.96 | 88.05 | |
| AB-GNB | 77.665 | 868.61 | 101.95 | Benign | 95.78 | 85.76 | 90.49 | 85.41 |
| | | | | Malware | 58.12 | 83.95 | 68.69 | |
| AB-LR | 78.1466 | 912.04 | 412.52 | Benign | 95.28 | 99.28 | 97.24 | 95.41 |
| | | | | Malware | 96.15 | 78.62 | 86.51 | |
| AB-MNB | 66.4724 | 829.03 | 307.96 | Benign | 98.20 | 74.43 | 84.68 | 79.06 |
| | | | | Malware | 51.58 | 95.24 | 66.91 | |
| AB-RF | 95.547 | 843.32 | 101.28 | Benign | 98.01 | 99.86 | 98.93 | 98.24 |
| | | | | Malware | 99.31 | 91.14 | 95.05 | |

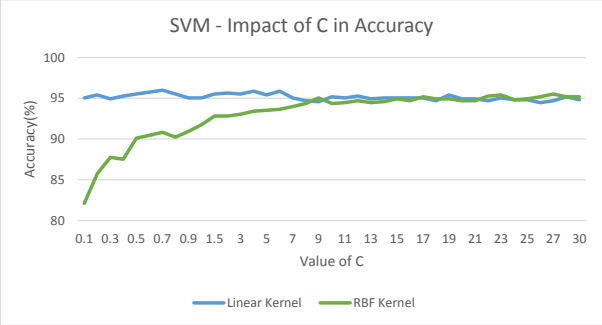


Figure 8: SVM Classification results for Accounted CSV using Linear and RBF Kernels with a Bag of System Call of size 500.

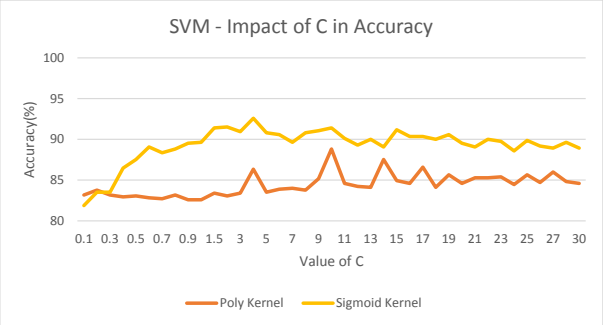


Figure 9: SVM Classification results for Accounted CSV using Poly and Sigmoid Kernels with a Bag of System Call of size 500.

Table 11: Classification results for Normalized CSV using Multi-layer Perceptron.

| Algorithm | Activation | H.L. | Ex. Time (s) | RAM (MB) | CPU (%) | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|---|---|
| MLP | Identity | 64 | 101.7424 | 945.03 | 844.38 | Benign | 97.97 | 97.13 | 97.55 | 96.00 |
| | | | | | | Malware | 87.42 | 90.85 | 89.10 | |
| | | 100 | 185.7222 | 981.12 | 867.02 | Benign | 98.34 | 96.17 | 97.24 | 95.65 |
| | | | | | | Malware | 86.10 | 93.60 | 89.69 | |
| | Logistic | 64 | 157.5304 | 979.75 | 888.54 | Benign | 97.96 | 97.11 | 97.53 | 96.00 |
| | | | | | | Malware | 87.88 | 91.19 | 89.51 | |
| | | 100 | 331.759 | 1014.58 | 855.9 | Benign | 96.57 | 97.83 | 97.19 | 95.41 |
| | | | | | | Malware | 90.07 | 85.00 | 87.46 | |
| | Relu | 64 | 99.3712 | 947.49 | 812.78 | Benign | 96.99 | 98.54 | 97.76 | 96.35 |
| | | | | | | Malware | 93.46 | 87.20 | 90.22 | |
| | | 100 | 116.9352 | 1101.84 | 818.1 | Benign | 96.32 | 99.24 | 97.76 | 96.47 |
| | | | | | | Malware | 97.06 | 86.84 | 91.67 | |
| | Tahn | 64 | 106.8874 | 930.29 | 846.22 | Benign | 96.93 | 97.78 | 97.35 | 95.76 |
| | | | | | | Malware | 91.02 | 87.86 | 89.41 | |
| | | 100 | 236.998 | 1006.08 | 883.64 | Benign | 96.10 | 98.23 | 97.16 | 95.41 |
| | | | | | | Malware | 92.36 | 84.30 | 88.15 | |

Table 12: Classification results for Normalized CSV using Support Vector Machine (Best Accuracy for each kernel).

| Algorithm | Ex. Time (s) | RAM (MB) | CPU (%) | Kernel | C | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|---|---|
| SVM | 17.82 | 573.52 | 518.7785 | Linear | 0.4 | Benign | 97.85 | 97.71 | 97.78 | 96.35 |
| | | | | | | Malware | 89.54 | 90.13 | 89.84 | |
| | 34.82 | 752.04 | 751.5495 | RBF | 7 | Benign | 98.40 | 98.12 | 98.26 | 97.18 |
| | | | | | | Malware | 91.93 | 93.08 | 92.50 | |
| | 21.92 | 642.65 | 604.795 | Poly | 29 | Benign | 96.91 | 97.63 | 97.27 | 95.65 |
| | | | | | | Malware | 90.59 | 88.00 | 89.28 | |
| | 18.06 | 581.95 | 509.3925 | Sigmoid | 2 | Benign | 95.00 | 97.22 | 96.10 | 93.65 |
| | | | | | | Malware | 87.33 | 78.92 | 82.91 | |

### 4.4.2 Classification Using the Bag of System Calls with Size of 500 for Normalized CSV

The classification for the Normalized CSV using the Bag of System Calls technique culminated in obtaining the results presented in the tables 16, 17 and 18. Analyzing the results and comparing them with those obtained in the Section 4.3 we can see that we obtained improvements in eight algorithms namely in DT, GNB, KNN(2,3,5,10), MNB and Adaboost-MNB. Similarly to what happened for the results with the Bag of System Calls presented in the Section 4.4.1, we highlight the algorithm GNB and Adaboost-GNB that obtained improvements of 26.58% and 6.35%, respectively. For the MLP we did not get any improvements. For the SVM classification we present table 18 and figures 10 and 11 where we can see an improvement for Poly Kernel. We also point out that SVM with the RBF kernel reached the Accuracy of the RF algorithm.

### 4.4.3 Classification Using Bag of System Calls with Size of 750 for Accounted CSV

Comparing the results with the ones obtained in Section 4.2, we can see that we have nine improvements, the same number as in Section 4.4.1 but for different algorithms. We observed an increase of the Accuracy of the Random Forest algorithm by 0.59%, from 97.06% to 97.65%. The biggest improvement was observed in Adaboost-GNB that increased the Accuracy by 8.59%, from 83.53% to 92.12%. MLP improved for both Hidden Layer sizes with Relu activation function and with Tahn with Hidden Layer size of 100. The same has seen in Section 4.4.1 but with smaller improvements, as we can see in table

Table 13: Classification results using Bag of System Calls with size of 500 for Accounted CSV with DT, GNB, KNN, LR, MNB, RF, and AB algorithms.

| Algorithm | Ex. Time (s) | RAM (MB) | CPU (%) | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|
| DT | 1.79 | 116.46 | 187.35 | Benign | 97.41 | 95.81 | 96.60 | 94.71 |
| | | | | Malware | 85.49 | 90.66 | 88.00 | |
| GNB | 1.76 | 118.87 | 189.2 | Benign | 92.39 | 95.63 | 93.98 | 90.12 |
| | | | | Malware | 78.57 | 67.07 | 72.37 | |
| KNN2 | 1.84 | 121.8 | 241.29 | Benign | 96.47 | 98.70 | 97.57 | 96.00 |
| | | | | Malware | 93.62 | 84.08 | 88.59 | |
| KNN3 | 1.84 | 129.07 | 240.54 | Benign | 97.85 | 98.55 | 98.20 | 97.06 |
| | | | | Malware | 93.46 | 90.51 | 91.96 | |
| KNN5 | 1.84 | 54.95 | 241.54 | Benign | 96.26 | 97.81 | 97.03 | 95.18 |
| | | | | Malware | 90.26 | 84.24 | 87.15 | |
| KNN10 | 1.83 | 125.33 | 243.08 | Benign | 93.22 | 98.99 | 96.02 | 93.29 |
| | | | | Malware | 93.81 | 67.95 | 78.81 | |
| LR | 1.93 | 122.43 | 297.62 | Benign | 96.93 | 97.79 | 97.36 | 95.76 |
| | | | | Malware | 90.91 | 87.72 | 89.29 | |
| MNB | 1.79 | 118.65 | 235.79 | Benign | 95.31 | 96.96 | 96.13 | 93.65 |
| | | | | Malware | 85.71 | 79.25 | 82.35 | |
| RF | 2.09 | 130.04 | 175.68 | Benign | 97.58 | 98.99 | 98.28 | 97.18 |
| | | | | Malware | 95.27 | 89.24 | 92.16 | |
| AB | 2.31 | 135.09 | 167.83 | Benign | 95.03 | 97.45 | 96.23 | 94.00 |
| | | | | Malware | 89.76 | 81.42 | 85.39 | |
| AB-DT | 2.31 | 129.71 | 168.54 | Benign | 95.38 | 98.27 | 96.80 | 94.71 |
| | | | | Malware | 91.18 | 78.98 | 84.64 | |
| AB-GNB | 5.07 | 161.29 | 130.91 | Benign | 93.55 | 97.23 | 95.35 | 92.35 |
| | | | | Malware | 86.13 | 71.95 | 78.41 | |
| AB-LR | 4.39 | 163.1 | 749.89 | Benign | 96.03 | 98.83 | 97.41 | 95.76 |
| | | | | Malware | 94.44 | 82.93 | 88.31 | |
| AB-MNB | 3.66 | 123.46 | 681.02 | Benign | 91.90 | 90.28 | 91.08 | 85.88 |
| | | | | Malware | 63.93 | 68.42 | 66.10 | |
| AB-RF | 19.75 | 218.56 | 107.57 | Benign | 97.16 | 99.56 | 98.35 | 97.29 |
| | | | | Malware | 97.95 | 87.73 | 92.56 | |

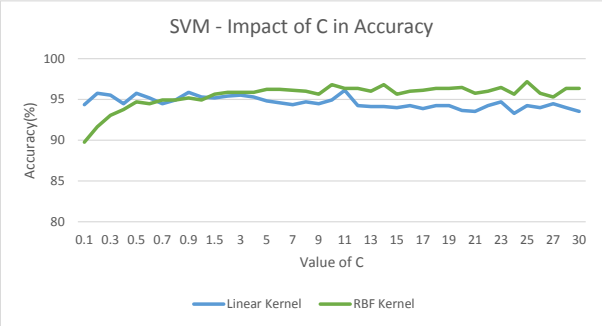


Figure 10: SVM classification results for Normalized CSV using Linear and RBF Kernels with a Bag of System Call of size 500.

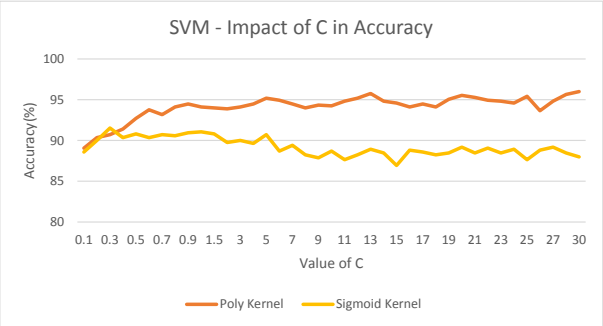


Figure 11: SVM classification results for Normalized CSV using Poly and Sigmoid Kernels with a Bag of System Call of size 500.

Table 14: Classification results using Bag of System Calls with size of 500 for Accounted CSV with Multi-layer Perceptron.

| Algorithm | Activation | H.L. | Ex. Time(s) | RAM (MB) | CPU (%) | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|---|---|
| MLP | Identity | 64 | 5.95 | 165.82 | 826.77 | Benign | 97.53 | 96.13 | 96.83 | 94.82 |
| | | | | | | Malware | 83.33 | 88.82 | 85.99 | |
| | | 100 | 10.05 | 168.51 | 943.12 | Benign | 97.31 | 95.60 | 96.45 | 94.35 |
| | | | | | | Malware | 83.33 | 89.29 | 86.21 | |
| | Logistic | 64 | 12.26 | 169.17 | 971.16 | Benign | 97.53 | 97.10 | 97.31 | 95.65 |
| | | | | | | Malware | 87.73 | 89.38 | 88.54 | |
| | | 100 | 22.17 | 171.35 | 1027.09 | Benign | 96.77 | 97.06 | 96.92 | 95.06 |
| | | | | | | Malware | 88.10 | 87.06 | 87.57 | |
| | Relu | 64 | 7.62 | 163.82 | 875.96 | Benign | 98.12 | 97.14 | 97.63 | 96.12 |
| | | | | | | Malware | 87.26 | 91.33 | 89.25 | |
| | | 100 | 16.21 | 170.61 | 996.06 | Benign | 98.07 | 97.64 | 97.86 | 96.59 |
| | | | | | | Malware | 90.86 | 92.44 | 91.64 | |
| | Tahn | 64 | 8.4 | 168.49 | 884.83 | Benign | 96.48 | 97.48 | 96.98 | 95.18 |
| | | | | | | Malware | 89.88 | 86.29 | 88.05 | |
| | | 100 | 17.24 | 172.95 | 993.33 | Benign | 96.57 | 97.27 | 96.92 | 94.94 |
| | | | | | | Malware | 87.33 | 84.52 | 85.90 | |

Table 15: Classification Results using Bag of System Calls with size 500 for Accounted CSV with Support Vector Machine.

| Algorithm | Time(sec) | RAM(MB) | CPU(%) | Kernel | C | Sample | PRE | REC | F1 | ACC |
|---|---|---|---|---|---|---|---|---|---|---|
| SVM | 2.18 | 124.07 | 169.22 | Linear | 0.7 | Benign | 96.83 | 98.25 | 97.53 | 96.00 |
| | | | | | | Malware | 92.31 | 86.75 | 89.44 | |
| | 2.39 | 127.95 | 163.37 | RBF | 27 | Benign | 95.03 | 99.71 | 97.31 | 95.53 |
| | | | | | | Malware | 98.41 | 77.50 | 86.71 | |
| | 2.36 | 123.32 | 163.77 | Poly | 10 | Benign | 88.13 | 99.56 | 93.50 | 88.82 |
| | | | | | | Malware | 96.00 | 43.90 | 60.25 | |
| | 2.20 | 121.39 | 169.61 | Sigmoid | 4 | Benign | 93.66 | 97.56 | 95.57 | 92.59 |
| | | | | | | Malware | 86.40 | 70.13 | 77.42 | |

19. Regarding the SVM we registered an improvement for Poly Kernel.

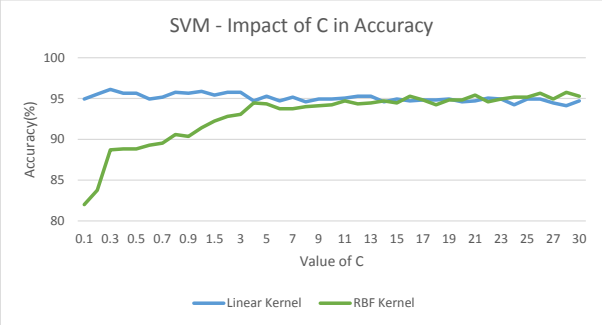


Figure 12: SVM classification results for Accounted CSV using Linear and RBF Kernels with a Bag of System Call of size 750.

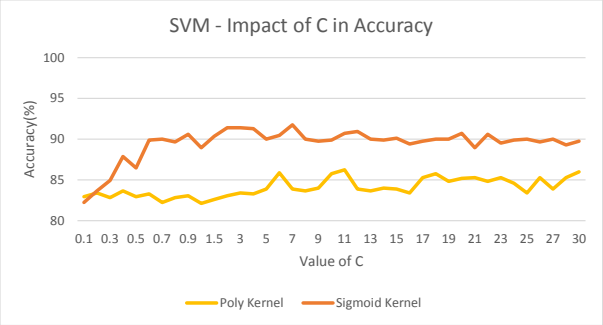


Figure 13: SVM classification results for Accounted CSV using Linear and RBF Kernels with a Bag of System Call of size 750.

Table 16: Classification results using Bag of System Calls with size of 500 for Normalized CSV with DT, GNB, KNN, LR, MNB, RF, and AB algorithms.

| Algorithm | Ex. Time (s) | RAM (MB) | CPU (%) | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|
| DT | 1.8 | 125.35 | 190.72 | Benign | 96.75 | 97.03 | 96.89 | 95.06 |
| | | | | Malware | 88.51 | 87.50 | 88.00 | |
| GNB | 1.77 | 118.6 | 192.77 | Benign | 91.75 | 96.11 | 93.88 | 89.76 |
| | | | | Malware | 78.05 | 61.54 | 68.82 | |
| KNN2 | 1.86 | 127.92 | 245.05 | Benign | 96.15 | 99.26 | 97.68 | 96.24 |
| | | | | Malware | 96.64 | 84.21 | 90.00 | |
| KNN3 | 1.99 | 134.86 | 240 | Benign | 97.67 | 98.10 | 97.88 | 96.59 |
| | | | | Malware | 92.02 | 90.36 | 91.19 | |
| KNN5 | 1.87 | 126.1 | 243.5 | Benign | 96.99 | 97.83 | 97.41 | 95.76 |
| | | | | Malware | 90.20 | 86.79 | 88.46 | |
| KNN10 | 2.1 | 118.32 | 239.27 | Benign | 93.23 | 98.83 | 95.95 | 93.29 |
| | | | | Malware | 93.65 | 70.66 | 80.55 | |
| LR | 1.91 | 125.7 | 285.55 | Benign | 96.20 | 98.70 | 97.43 | 95.76 |
| | | | | Malware | 93.57 | 82.91 | 87.92 | |
| MNB | 1.8 | 133.25 | 237.44 | Benign | 94.52 | 97.25 | 95.87 | 93.18 |
| | | | | Malware | 86.23 | 75.32 | 80.41 | |
| RF | 2.1 | 141.18 | 177.94 | Benign | 97.36 | 99.10 | 98.22 | 97.18 |
| | | | | Malware | 96.43 | 90.00 | 93.10 | |
| AB | 2.31 | 147.76 | 171.16 | Benign | 96.22 | 97.07 | 96.64 | 94.59 |
| | | | | Malware | 87.65 | 84.52 | 86.06 | |
| AB-DT | 2.33 | 141.95 | 170.5 | Benign | 95.51 | 97.84 | 96.66 | 94.47 |
| | | | | Malware | 89.05 | 79.22 | 83.85 | |
| AB-GNB | 5.52 | 161.49 | 129 | Benign | 92.60 | 97.50 | 94.99 | 91.76 |
| | | | | Malware | 87.31 | 68.82 | 76.97 | |
| AB-LR | 4.49 | 167.04 | 759.22 | Benign | 95.67 | 97.36 | 96.51 | 94.35 |
| | | | | Malware | 88.54 | 82.25 | 85.28 | |
| AB-MNB | 3.79 | 172.95 | 697.83 | Benign | 98.70 | 66.76 | 79.65 | 72.71 |
| | | | | Malware | 42.05 | 96.47 | 58.57 | |
| AB-RF | 19.85 | 227.09 | 107.58 | Benign | 97.53 | 99.26 | 98.39 | 97.41 |
| | | | | Malware | 96.89 | 90.17 | 93.41 | |

#### 4.4.4 Classification using Bag of System Calls with size of 750 for Normalized CSV

The results obtained in this Section are presented in tables 22, 23 and 24. We registered Accuracy improvements in seven of the fifteen algorithms. There were no improvements for Random Forest or for Adaboost-RF, however it is possible to observe that the best improvement was 7.53% for the Adaboost-GNB algorithm. For the MLP, there were no improvements as we can see in table 23. On the SVM side, we highlight the result of 97.06% with the RBF Kernel and the Poly Kernel improvements, shown in table 24 and in figures 14 and 15.

#### 4.4.5 Classification Using Bag of System Calls with Size of 1000 for Accounted CSV

Using a Bag of System Calls with size of 1000 brought us to the results presented in tables 25, 26 and 27 and figures 16 and 17. We registered an increase of the Accuracy of Random Forest by 0.23%, bringing it to 97.29%. Regarding the Adaboost-RF we registered the same Accuracy. In addition, we can see

Table 17: Classification results using Bag of System Calls with size of 500 for Normalized CSV with Multi-layer Perceptron.

| Algorithm | Activation | H.L. | Ex. Time (s) | RAM (MB) | CPU (%) | Sample | Pr(%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|---|---|
| MLP | Identity | 64 | 6.25 | 170.3 | 853.44 | Benign | 96.18 | 97.33 | 96.75 | 94.82 |
| | | | | | | Malware | 89.35 | 85.31 | 87.28 | |
| | | 100 | 10.67 | 177.76 | 957.77 | Benign | 96.03 | 97.03 | 96.53 | 94.47 |
| | | | | | | Malware | 88.24 | 84.75 | 86.46 | |
| | Logistic | 64 | 14.32 | 177.58 | 997.38 | Benign | 97.20 | 97.35 | 97.27 | 95.65 |
| | | | | | | Malware | 89.47 | 88.95 | 89.21 | |
| | | 100 | 27.79 | 182.33 | 1046.55 | Benign | 95.92 | 97.63 | 96.77 | 94.82 |
| | | | | | | Malware | 90.18 | 84.00 | 86.98 | |
| | Relu | 64 | 7.68 | 174.92 | 892.88 | Benign | 98.25 | 96.69 | 97.46 | 95.88 |
| | | | | | | Malware | 86.14 | 92.26 | 89.10 | |
| | | 100 | 14.74 | 179.19 | 998 | Benign | 97.93 | 97.50 | 97.72 | 96.35 |
| | | | | | | Malware | 90.17 | 91.76 | 90.96 | |
| | Tahn | 64 | 10.7 | 175.38 | 950.94 | Benign | 98.36 | 96.07 | 97.20 | 95.53 |
| | | | | | | Malware | 84.92 | 93.25 | 88.89 | |
| | | 100 | 22.62 | 182.53 | 1028.55 | Benign | 97.19 | 96.76 | 96.97 | 95.18 |
| | | | | | | Malware | 87.43 | 88.95 | 88.18 | |

Table 18: Classification results using Bag of System Calls with size of 500 for Normalized CSV with Support Vector Machine.

| Algorithm | Ex. Time(s) | RAM (MB) | CPU (%) | Kernel | C | Sample | Pr(%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|---|---|
| SVM | 2.26 | 128.28 | 165.48 | Linear | 11 | Benign | 97.36 | 97.79 | 97.58 | 96.12 |
| | | | | | | Malware | 91.02 | 89.41 | 90.21 | |
| | 2.48 | 126.79 | 159.32 | RBF | 25 | Benign | 97.60 | 99.00 | 98.29 | 97.18 |
| | | | | | | Malware | 95.10 | 88.89 | 91.89 | |
| | 2.30 | 125.42 | 163.92 | Poly | 30 | Benign | 97.02 | 98.13 | 97.57 | 96.00 |
| | | | | | | Malware | 91.03 | 86.27 | 88.59 | |
| | 2.26 | 120.26 | 166.0 | Sigmoid | 0.3 | Benign | 92.27 | 97.66 | 94.89 | 91.53 |
| | | | | | | Malware | 87.30 | 66.27 | 75.34 | |

Table 19: Classification results using Bag of System Calls with size of 750 for Accounted CSV with Multi-layer Perceptron.

| Algorithm | Activation | H.L. | Ex. Time (s) | RAM (MB) | CPU (%) | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|---|---|
| MLP | Identity | 64 | 8.03 | 198.63 | 847.6 | Benign | 97.69 | 96.86 | 97.27 | 95.53 |
| | | | | | | Malware | 85.90 | 89.33 | 87.58 | |
| | | 100 | 14.9 | 210.79 | 983.4 | Benign | 97.50 | 95.40 | 96.44 | 94.24 |
| | | | | | | Malware | 81.18 | 89.03 | 84.92 | |
| | Logistic | 64 | 16.56 | 214.34 | 991.85 | Benign | 96.83 | 97.53 | 97.18 | 95.41 |
| | | | | | | Malware | 89.10 | 86.34 | 87.70 | |
| | | 100 | 36.23 | 220.11 | 1049.9 | Benign | 97.21 | 96.64 | 96.92 | 95.06 |
| | | | | | | Malware | 86.47 | 88.55 | 87.50 | |
| | Relu | 64 | 9.56 | 206.44 | 909 | Benign | 98.25 | 97.25 | 97.75 | 96.35 |
| | | | | | | Malware | 88.55 | 92.45 | 90.46 | |
| | | 100 | 21.27 | 217.31 | 1012.3 | Benign | 96.78 | 98.36 | 97.56 | 96.12 |
| | | | | | | Malware | 93.41 | 87.64 | 90.43 | |
| | Tahn | 64 | 13.14 | 210.87 | 948.25 | Benign | 98.20 | 96.18 | 97.18 | 95.53 |
| | | | | | | Malware | 85.79 | 92.90 | 89.20 | |
| | | 100 | 25.55 | 219.48 | 1022.45 | Benign | 97.20 | 96.20 | 96.70 | 94.71 |
| | | | | | | Malware | 84.88 | 88.48 | 86.65 | |

Table 20: Classification results using Bag of System Calls with size of 750 for Accounted CSV with DT, GNB, KNN, LR, MNB, RF, and AB algorithms.

| Algorithm | Ex. Time(s) | RAM (MB) | CPU(%) | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|
| DT | 2.09 | 139.67 | 176.95 | Benign | 97.30 | 96.72 | 97.01 | 95.29 |
| | | | | Malware | 88.04 | 90.00 | 89.01 | |
| GNB | 2.09 | 139.34 | 178.05 | Benign | 95.89 | 51.85 | 67.31 | 60.00 |
| | | | | Malware | 32.99 | 91.43 | 48.48 | |
| KNN2 | 2.15 | 139.21 | 227.1 | Benign | 96.23 | 98.96 | 97.58 | 96.12 |
| | | | | Malware | 95.63 | 85.47 | 90.27 | |
| KNN3 | 2.16 | 127.6 | 225.55 | Benign | 96.40 | 98.38 | 97.38 | 95.76 |
| | | | | Malware | 92.90 | 85.21 | 88.89 | |
| KNN5 | 2.15 | 131.6 | 226.45 | Benign | 95.95 | 98.08 | 97.01 | 95.18 |
| | | | | Malware | 91.77 | 83.82 | 87.61 | |
| KNN10 | 2.16 | 124.4 | 225.6 | Benign | 92.69 | 99.56 | 96.01 | 93.29 |
| | | | | Malware | 97.30 | 66.67 | 79.12 | |
| LR | 2.37 | 113.25 | 305.15 | Benign | 96.58 | 99.12 | 97.84 | 96.47 |
| | | | | Malware | 95.95 | 85.54 | 90.45 | |
| MNB | 2.11 | 118.75 | 216.45 | Benign | 96.32 | 95.61 | 95.96 | 93.53 |
| | | | | Malware | 82.46 | 84.94 | 83.68 | |
| RF | 2.48 | 150.11 | 164.95 | Benign | 97.87 | 99.28 | 98.57 | 97.65 |
| | | | | Malware | 96.60 | 90.45 | 93.42 | |
| AB | 2.86 | 141.32 | 156 | Benign | 96.59 | 97.42 | 97.00 | 95.06 |
| | | | | Malware | 87.67 | 84.21 | 85.91 | |
| AB-DT | 2.86 | 169.3 | 156.2 | Benign | 95.44 | 98.09 | 96.75 | 94.71 |
| | | | | Malware | 91.28 | 80.95 | 85.80 | |
| AB-GNB | 8.19 | 195.74 | 119.6 | Benign | 94.85 | 95.27 | 95.06 | 92.12 |
| | | | | Malware | 81.29 | 79.89 | 80.58 | |
| AB-LR | 7.66 | 196.89 | 864.45 | Benign | 94.47 | 99.13 | 96.74 | 94.59 |
| | | | | Malware | 95.28 | 75.16 | 84.03 | |
| AB-MNB | 6.47 | 196.5 | 821.6 | Benign | 85.54 | 93.78 | 89.47 | 82.47 |
| | | | | Malware | 61.82 | 38.86 | 47.72 | |
| AB-RF | 23.16 | 247.8 | 106.4 | Benign | 97.52 | 99.70 | 98.60 | 97.76 |
| | | | | Malware | 98.78 | 90.50 | 94.46 | |

Table 21: Classification Results using Bag of System Calls with size 750 for Accounted CSV with Support Vector Machine.

| Algorithm | Ex. Time (s) | RAM (MB) | CPU (%) | Kernel | C | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|---|---|
| SVM | 2.88 | 148.70 | 151.41 | Linear | 0.3 | Benign | 96.58 | 98.69 | 97.62 | 96.12 |
| | | | | | | Malware | 93.96 | 85.37 | 89.46 | |
| | 3.19 | 152.05 | 146.83 | RBF | 28 | Benign | 96.43 | 98.40 | 97.41 | 95.76 |
| | | | | | | Malware | 92.62 | 84.66 | 88.46 | |
| | 3.25 | 150.40 | 145.25 | Poly | 11 | Benign | 85.71 | 99.86 | 92.25 | 86.24 |
| | | | | | | Malware | 97.37 | 24.18 | 38.74 | |
| | 2.87 | 148.16 | 151.61 | Sigmoid | 7 | Benign | 93.58 | 96.55 | 95.04 | 91.76 |
| | | | | | | Malware | 81.95 | 70.32 | 75.69 | |

Table 22: Classification results using Bag of System Calls with size of 750 for Normalized CSV with DT, GNB, KNN, LR, MNB, RF, and AB algorithms.

| Algorithm | Ex. Time (s) | RAM (MB) | CPU (%) | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|
| DT | 2.09 | 120.16 | 175.6 | Benign | 96.64 | 96.36 | 96.50 | 94.35 |
| | | | | Malware | 84.94 | 85.98 | 85.45 | |
| GNB | 2.08 | 122.33 | 175.3 | Benign | 95.96 | 45.91 | 62.11 | 55.65 |
| | | | | Malware | 31.06 | 92.66 | 46.52 | |
| KNN2 | 2.15 | 129.91 | 226.1 | Benign | 96.40 | 98.96 | 97.66 | 96.24 |
| | | | | Malware | 95.51 | 85.63 | 90.30 | |
| KNN3 | 2.14 | 127.48 | 225.35 | Benign | 97.67 | 98.53 | 98.10 | 96.94 |
| | | | | Malware | 93.90 | 90.59 | 92.22 | |
| KNN5 | 2.15 | 135.63 | 227.75 | Benign | 96.12 | 98.09 | 97.10 | 95.29 |
| | | | | Malware | 91.56 | 83.93 | 87.58 | |
| KNN10 | 2.16 | 128.47 | 224.5 | Benign | 93.68 | 98.67 | 96.11 | 93.65 |
| | | | | Malware | 93.48 | 74.14 | 82.69 | |
| LR | 2.28 | 126.63 | 281.4 | Benign | 97.31 | 97.46 | 97.38 | 95.88 |
| | | | | Malware | 90.61 | 90.11 | 90.36 | |
| MNB | 2.09 | 125.35 | 214.05 | Benign | 93.37 | 98.37 | 95.80 | 93.18 |
| | | | | Malware | 92.20 | 73.45 | 81.76 | |
| RF | 2.46 | 146.36 | 163.4 | Benign | 97.55 | 98.98 | 98.26 | 97.18 |
| | | | | Malware | 95.54 | 89.82 | 92.59 | |
| AB | 2.83 | 157.13 | 155.8 | Benign | 95.46 | 98.10 | 96.76 | 94.71 |
| | | | | Malware | 91.03 | 80.49 | 85.44 | |
| AB-DT | 2.82 | 144.6 | 156.75 | Benign | 95.24 | 98.41 | 96.80 | 94.71 |
| | | | | Malware | 91.85 | 78.48 | 84.64 | |
| AB-GNB | 8.17 | 203.29 | 118.9 | Benign | 95.04 | 96.16 | 95.59 | 92.94 |
| | | | | Malware | 84.24 | 80.35 | 82.25 | |
| AB-LR | 7.42 | 205.63 | 851.5 | Benign | 95.16 | 98.67 | 96.88 | 94.94 |
| | | | | Malware | 93.92 | 80.35 | 86.60 | |
| AB-MNB | 6.4 | 200.42 | 824.05 | Benign | 99.07 | 63.06 | 77.06 | 70.24 |
| | | | | Malware | 40.86 | 97.73 | 57.62 | |
| AB-RF | 23.01 | 249.13 | 106.45 | Benign | 98.45 | 98.87 | 98.66 | 97.76 |
| | | | | Malware | 94.33 | 92.36 | 93.33 | |

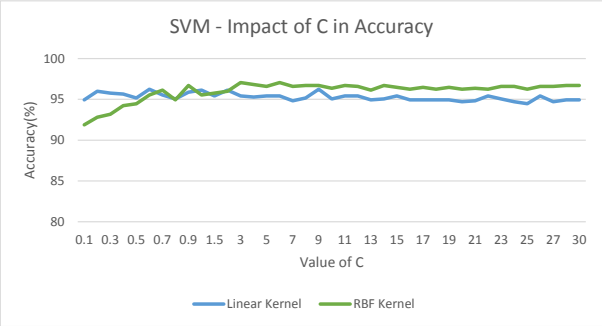


Figure 14: SVM classification results for Normalized CSV using Linear and RBF Kernels with a Bag of System Call of size of 750.

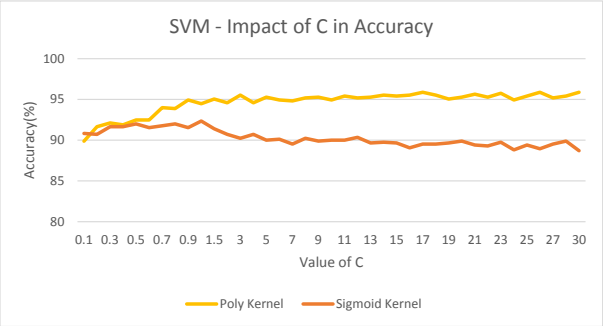


Figure 15: SVM classification results for Normalized CSV using Poly and Sigmoid Kernels with a Bag of System Call of size of 750.

Table 23: Classification results using Bag of System Calls with size of 750 for Normalized CSV with Multi-layer Perceptron.

| Algorithm | Activation | H.L. | Ex. Time (s) | RAM (MB) | CPU (%) | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|---|---|
| MLP | Identity | 64 | 8.03 | 198.63 | 847.6 | Benign | 97.69 | 96.86 | 97.27 | 95.53 |
| | | | | | | Malware | 85.90 | 89.33 | 87.58 | |
| | | 100 | 14.9 | 210.79 | 983.4 | Benign | 97.50 | 95.40 | 96.44 | 94.24 |
| | | | | | | Malware | 81.18 | 89.03 | 84.92 | |
| | Logistic | 64 | 16.56 | 214.34 | 991.85 | Benign | 96.83 | 97.53 | 97.18 | 95.41 |
| | | | | | | Malware | 89.10 | 86.34 | 87.70 | |
| | | 100 | 36.23 | 220.11 | 1049.9 | Benign | 97.21 | 96.64 | 96.92 | 95.06 |
| | | | | | | Malware | 86.47 | 88.55 | 87.50 | |
| | Relu | 64 | 9.56 | 206.44 | 909 | Benign | 98.25 | 97.25 | 97.75 | 96.35 |
| | | | | | | Malware | 88.55 | 92.45 | 90.46 | |
| | | 100 | 21.27 | 217.31 | 1012.3 | Benign | 96.78 | 98.36 | 97.56 | 96.12 |
| | | | | | | Malware | 93.41 | 87.64 | 90.43 | |
| | Tahn | 64 | 13.14 | 210.87 | 948.25 | Benign | 98.20 | 96.18 | 97.18 | 95.53 |
| | | | | | | Malware | 85.79 | 92.90 | 89.20 | |
| | | 100 | 25.55 | 219.48 | 1022.45 | Benign | 97.20 | 96.20 | 96.70 | 94.71 |
| | | | | | | Malware | 84.88 | 88.48 | 86.65 | |

Table 24: Classification results using Bag of System Calls with size of 750 for Normalized CSV with Support Vector Machine.

| Algorithm | Ex. Time (s) | RAM (MB) | CPU (%) | Kernel | C | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|---|---|
| SVM | 2.76 | 137.74 | 153.92 | Linear | 0.6 | Benign | 96.80 | 98.52 | 97.65 | 96.24 |
| | | | | | | Malware | 93.83 | 87.36 | 90.48 | |
| | | | | | 9 | Benign | 96.84 | 98.54 | 97.68 | 96.24 |
| | | | | | | Malware | 93.46 | 86.67 | 89.94 | |
| | 3.15 | 155.22 | 146.57 | RBF | 3 | Benign | 97.32 | 99.14 | 98.22 | 97.06 |
| | | | | | | Malware | 95.77 | 87.74 | 91.58 | |
| | | | | | 6 | Benign | 97.51 | 99.02 | 98.26 | 97.06 |
| | | | | | | Malware | 94.53 | 87.05 | 90.64 | |
| | 2.85 | 142.19 | 151.21 | Poly | 17 | Benign | 96.10 | 98.81 | 97.44 | 95.88 |
| | | | | | | Malware | 94.90 | 84.66 | 89.49 | |
| | | | | | 26 | Benign | 97.36 | 97.51 | 97.44 | 95.88 |
| | | | | | | Malware | 89.82 | 89.29 | 89.55 | |
| | | | | | 30 | Benign | 96.26 | 98.67 | 97.45 | 95.88 |
| | | | | | | Malware | 94.16 | 84.80 | 89.23 | |
| | 2.74 | 142.70 | 154.05 | Sigmoid | 1 | Benign | 91.55 | 97.85 | 94.60 | 90.82 |
| | | | | | | Malware | 85.58 | 58.55 | 69.53 | |

seven improvements in table 25. In table 26 devoted to MLP, we also have two improvements using Relu and Tahn activation function both with Hidden Layer size 64. For the SVM, as shown in table 27 and in figures 16 and 17, no improvements were obtained in terms of Accuracy.

#### 4.4.6 Classification Using Bag of System Calls with Size of 1000 for Normalized CSV

The results obtained for BoSC with Size of 1000 for Normalized CSV are available in tables 28, 29, 30 and in figures 18 and 19. Considering the table 28 registered five improvements comparing with the results from Section 4.3. We highlight the Accuracy reached by Adaboost-RF of 98.12%, only 0.12% below the result obtained in Section 4.3. For the MLPs we obtained five improvements and we also highlight the Accuracy of 97.18% obtained with Relu Activation Function with an Hidden Layer with size of 64. This Accuracy matches the second best Accuracy from Random Forest. For SVM, as can be

Table 25: Classification results using Bag of System Calls with size of 1000 for Accounted CSV with DT, GNB, KNN, LR, MNB, RF, and AB algorithms.

| Algorithm | Ex. Time (s) | RAM (MB) | CPU (%) | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|
| DT | 2.5 | 131.55 | 166.6 | Benign | 97.04 | 96.61 | 96.83 | 94.94 |
| | | | | Malware | 86.78 | 88.30 | 87.54 | |
| GNB | 2.5 | 166.79 | 167.3 | Benign | 97.18 | 46.62 | 63.01 | 57.18 |
| | | | | Malware | 33.15 | 95.14 | 49.16 | |
| KNN2 | 2.51 | 146.74 | 212.1 | Benign | 96.59 | 99.27 | 97.91 | 96.59 |
| | | | | Malware | 96.58 | 85.45 | 90.68 | |
| KNN3 | 2.53 | 130.68 | 208.5 | Benign | 97.16 | 97.45 | 97.30 | 95.76 |
| | | | | Malware | 90.66 | 89.67 | 90.16 | |
| KNN5 | 2.53 | 139.68 | 212.8 | Benign | 95.56 | 97.94 | 96.74 | 94.71 |
| | | | | Malware | 90.79 | 81.66 | 85.98 | |
| KNN10 | 2.51 | 133.2 | 212 | Benign | 92.58 | 99.42 | 95.88 | 93.06 |
| | | | | Malware | 96.33 | 65.62 | 78.07 | |
| LR | 2.83 | 156.04 | 301.5 | Benign | 97.23 | 97.23 | 97.23 | 95.53 |
| | | | | Malware | 88.34 | 88.34 | 88.34 | |
| MNB | 2.46 | 136.14 | 200.44 | Benign | 95.79 | 95.79 | 95.79 | 93.18 |
| | | | | Malware | 81.99 | 81.99 | 81.99 | |
| RF | 2.91 | 164.93 | 154.55 | Benign | 97.97 | 98.69 | 98.33 | 97.29 |
| | | | | Malware | 94.34 | 91.46 | 92.88 | |
| AB | 3.47 | 170.67 | 148.44 | Benign | 95.33 | 97.82 | 96.56 | 94.35 |
| | | | | Malware | 89.51 | 79.50 | 84.21 | |
| AB-DT | 3.5 | 201.53 | 147.77 | Benign | 95.72 | 97.96 | 96.83 | 94.82 |
| | | | | Malware | 90.60 | 81.82 | 85.99 | |
| AB-GNB | 13.01 | 238.55 | 112.11 | Benign | 95.38 | 95.66 | 95.52 | 92.71 |
| | | | | Malware | 80.89 | 79.87 | 80.38 | |
| AB-LR | 14.53 | 251.8 | 922.33 | Benign | 95.01 | 98.81 | 96.88 | 94.94 |
| | | | | Malware | 94.59 | 80.00 | 86.69 | |
| AB-MNB | 13.12 | 244.99 | 906.22 | Benign | 87.01 | 92.43 | 89.64 | 83.06 |
| | | | | Malware | 61.94 | 47.16 | 53.55 | |
| AB-RF | 27.5 | 274.42 | 105.77 | Benign | 98.01 | 99.28 | 98.64 | 97.76 |
| | | | | Malware | 96.60 | 91.03 | 93.73 | |

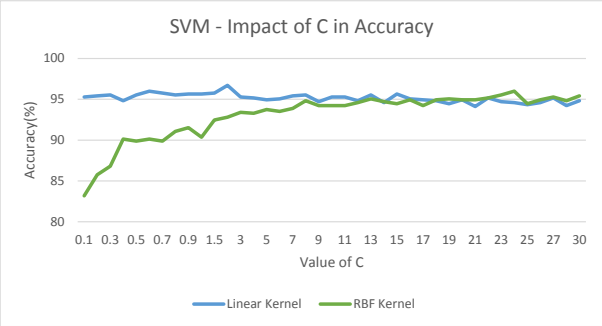


Figure 16: SVM classification results for Accounted CSV using Linear and RBF Kernels with a Bag of System Call of size of 1000.

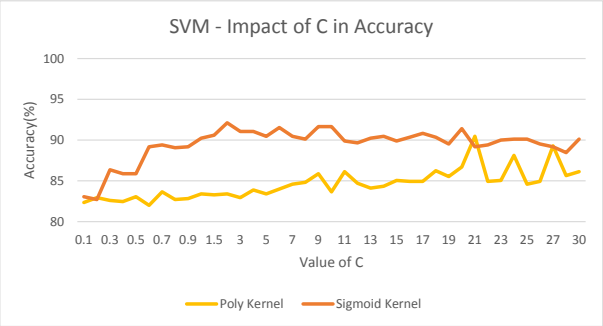


Figure 17: SVM Classification results for Accounted CSV using Poly and Sigmoid Kernels with a Bag of System Call of size 1000.

Table 26: Classification results using Bag of System Calls with size of 1000 for Accounted CSV with Multi-layer Perceptron.

| Algorithm | Activation | H.L. | Ex. Time (s) | RAM (MB) | CPU (%) | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|---|---|
| MLP | Identity | 64 | 12.66 | 243.03 | 923 | Benign | 95.91 | 97.05 | 96.48 | 94.35 |
| | | | | | | Malware | 87.88 | 83.82 | 85.80 | |
| | | 100 | 23.83 | 262.39 | 1011.77 | Benign | 96.42 | 96.57 | 96.50 | 94.47 |
| | | | | | | Malware | 87.15 | 86.67 | 86.91 | |
| | Logistic | 64 | 25.39 | 252.91 | 1013.55 | Benign | 97.11 | 97.53 | 97.32 | 95.65 |
| | | | | | | Malware | 89.24 | 87.58 | 88.40 | |
| | | 100 | 55.19 | 264.49 | 1064.77 | Benign | 96.19 | 97.48 | 96.83 | 94.94 |
| | | | | | | Malware | 89.82 | 85.23 | 87.46 | |
| | Relu | 64 | 17.82 | 264.49 | 972.1 | Benign | 96.12 | 98.96 | 97.52 | 96.00 |
| | | | | | | Malware | 95.45 | 84.48 | 89.63 | |
| | | 100 | 38.68 | 258.44 | 1044.2 | Benign | 96.66 | 98.08 | 97.36 | 95.76 |
| | | | | | | Malware | 91.98 | 86.63 | 89.22 | |
| | Tahn | 64 | 13.58 | 246.89 | 936.88 | Benign | 97.59 | 97.01 | 97.30 | 95.76 |
| | | | | | | Malware | 89.25 | 91.21 | 90.22 | |
| | | 100 | 34.25 | 261.26 | 1043.55 | Benign | 97.36 | 96.51 | 96.93 | 95.06 |
| | | | | | | Malware | 85.80 | 88.96 | 87.35 | |

Table 27: Classification results using Bag of System Calls with size of 1000 for Accounted CSV with Support Vector Machine.

| Algorithm | Ex. Time (s) | RAM (MB) | CPU (%) | Kernel | C | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|---|---|
| SVM | 3.76 | 158.73 | 136.22 | Linear | 0.7 | Benign | 96.93 | 97.79 | 97.36 | 95.76 |
| | | | | | | Malware | 90.96 | 87.79 | 89.35 | |
| | 4.38 | 168.02 | 130.38 | RBF | 27 | Benign | 96.32 | 97.98 | 97.14 | 95.29 |
| | | | | | | Malware | 90.28 | 83.33 | 86.67 | |
| | 4.43 | 178.67 | 130.13 | Poly | 10 | Benign | 83.21 | 100 | 90.84 | 83.65 |
| | | | | | | Malware | 100 | 13.66 | 24.04 | |
| | 3.78 | 154.318 | 136.38 | Sigmoid | 4 | Benign | 92.05 | 97.39 | 94.65 | 91.06 |
| | | | | | | Malware | 85.00 | 63.75 | 72.86 | |

seen in table 30 and figure 19, there is one improvement, for the Poly Kernel.

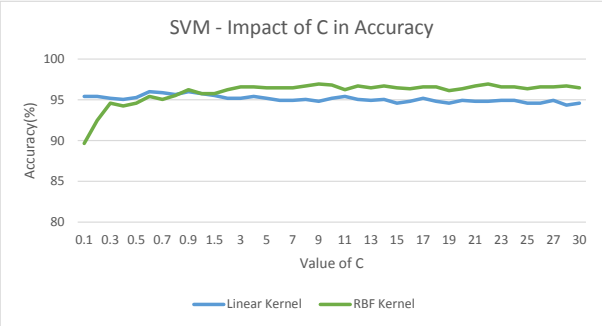


Figure 18: SVM classification results for Normalized CSV using Linear and RBF Kernels with a Bag of System Call of size 1000.

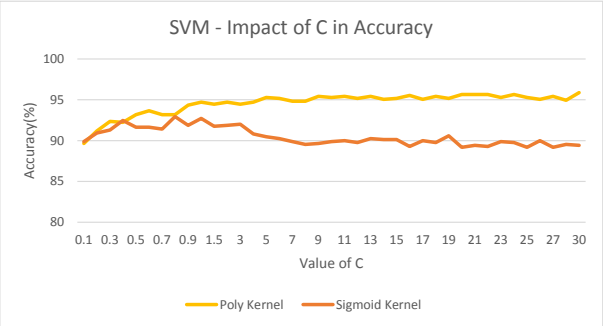


Figure 19: SVM classification results for Normalized CSV using Linear and RBF Kernels with a Bag of System Call of size 1000.

Table 28: Classification results using Bag of System Calls with size 1000 for Normalized CSV with DT, GNB, KNN, LR, MNB, RF, and AB algorithms.

| Algorithm | Ex. Time (s) | RAM (MB) | CPU (%) | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|
| DT | 2.39 | 147.77 | 166.47 | Benign | 97.86 | 95.37 | 96.60 | 94.71 |
| | | | | Malware | 84.26 | 92.22 | 88.06 | |
| GNB | 2.45 | 140.19 | 166.64 | Benign | 97.30 | 48.07 | 64.35 | 57.76 |
| | | | | Malware | 32.30 | 94.89 | 48.20 | |
| KNN2 | 2.52 | 151.43 | 209.87 | Benign | 96.71 | 98.11 | 97.41 | 95.76 |
| | | | | Malware | 91.39 | 85.71 | 88.46 | |
| KNN3 | 2.53 | 153.78 | 207.18 | Benign | 97.63 | 97.05 | 97.34 | 95.76 |
| | | | | Malware | 88.57 | 90.64 | 89.60 | |
| KNN5 | 2.49 | 155.81 | 210.18 | Benign | 97.38 | 97.10 | 97.24 | 95.53 |
| | | | | Malware | 87.73 | 88.82 | 88.27 | |
| KNN10 | 2.5 | 138.43 | 211.56 | Benign | 95.04 | 97.24 | 96.13 | 93.65 |
| | | | | Malware | 86.90 | 78.26 | 82.35 | |
| LR | 2.78 | 170.08 | 289.5 | Benign | 97.42 | 97.42 | 97.42 | 95.76 |
| | | | | Malware | 88.24 | 88.24 | 88.24 | |
| MNB | 2.47 | 133.58 | 199.86 | Benign | 94.15 | 96.92 | 95.51 | 92.71 |
| | | | | Malware | 85.91 | 75.74 | 80.50 | |
| RF | 2.96 | 165.9 | 152.86 | Benign | 96.97 | 99.56 | 98.25 | 97.18 |
| | | | | Malware | 98.08 | 87.93 | 92.73 | |
| AB | 3.52 | 167.96 | 145.13 | Benign | 95.14 | 98.52 | 96.80 | 94.82 |
| | | | | Malware | 93.38 | 80.57 | 86.50 | |
| AB-DT | 3.52 | 216.54 | 144.6 | Benign | 96.25 | 98.23 | 97.23 | 95.53 |
| | | | | Malware | 92.36 | 84.80 | 88.41 | |
| AB-GNB | 13.26 | 255.09 | 111.53 | Benign | 95.21 | 96.22 | 95.71 | 93.29 |
| | | | | Malware | 86.26 | 83.07 | 84.64 | |
| AB-LR | 14.77 | 259.75 | 930.93 | Benign | 95.78 | 97.63 | 96.70 | 94.71 |
| | | | | Malware | 90.12 | 83.43 | 86.65 | |
| AB-MNB | 13.32 | 256.82 | 917.86 | Benign | 99.52 | 61.16 | 75.76 | 69.06 |
| | | | | Malware | 40.27 | 98.88 | 57.24 | |
| AB-RF | 28.16 | 285.77 | 105.26 | Benign | 98.25 | 99.41 | 98.83 | 98.12 |
| | | | | Malware | 97.56 | 93.02 | 95.24 | |

## 4.5 Results Overview and Discussion

We presented in the previous subsection the classification results and the resource consumption results for the four defined scenarios. We now present the best classification results obtained for each of the scenarios. Table 31 shows the best classification results obtained for the four scenarios as well as the performance in terms of computational resources, being all obtained with the AB-RF algorithm. The best classification result is an Accuracy of 98.24% obtained by AB-RF in B scenario (Normalized CSV without BoSC algorithm). Also noteworthy is the second best result obtained by AB-RF with a BoSC of size 1000 for the Normalized file, where this algorithm reached an Accuracy of 98.12%.

Regarding the computational resources used by the algorithms, we can observe that there is a direct relationship with the technique in use. We verified the larger RAM consumption occur in scenarios A and B due to the use of all features, and consequently the execution times are larger.

The best results of the Random Forest algorithm for each scenario are also shown in table 32. The best Accuracy of 97.65% is obtained for C scenario (Accounted CSV with BoSC algorithm with size of

Table 29: Classification results using Bag of System Calls with size of 1000 for Normalized CSV with Multi-layer Perceptron.

| Algorithm | Activation | H.L. | Ex. Time (s) | RAM (MB) | CPU (%) | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|---|---|
| MLP | Identity | 64 | 11.71 | 260.85 | 918.43 | Benign | 97.16 | 96.43 | 96.79 | 94.94 |
| | | | | | | Malware | 86.81 | 89.27 | 88.02 | |
| | | 100 | 22.41 | 269.54 | 1006 | Benign | 96.44 | 96.44 | 96.44 | 94.35 |
| | | | | | | Malware | 86.29 | 86.29 | 86.29 | |
| | Logistic | 64 | 25.78 | 268.62 | 1019.06 | Benign | 98.12 | 96.87 | 97.49 | 95.88 |
| | | | | | | Malware | 85.99 | 91.22 | 88.52 | |
| | | 100 | 69.73 | 274.83 | 1076 | Benign | 98.37 | 95.53 | 96.93 | 95.06 |
| | | | | | | Malware | 82.39 | 92.95 | 87.35 | |
| | Relu | 64 | 15.8 | 262.63 | 960.31 | Benign | 97.78 | 98.65 | 98.21 | 97.18 |
| | | | | | | Malware | 94.86 | 91.71 | 93.26 | |
| | | 100 | 37.85 | 269.23 | 1050.25 | Benign | 98.10 | 96.97 | 97.53 | 96.00 |
| | | | | | | Malware | 87.27 | 91.72 | 89.44 | |
| | Tahn | 64 | 18.21 | 269.01 | 955.93 | Benign | 97.39 | 96.97 | 97.18 | 95.41 |
| | | | | | | Malware | 86.88 | 88.54 | 87.70 | |
| | | 100 | 55.02 | 273.59 | 1069.4 | Benign | 97.93 | 96.64 | 97.28 | 95.65 |
| | | | | | | Malware | 86.78 | 91.52 | 89.09 | |

Table 30: Classification results using Bag of System Calls with size of 1000 for Normalized CSV with Support Vector Machine.

| Algorithm | Ex. Time (s) | RAM (MB) | CPU (%) | Kernel | C | Sample | Pr (%) | Re (%) | F1 (%) | Ac (%) |
|---|---|---|---|---|---|---|---|---|---|---|
| SVM | 3.69 | 169.26 | 136.63 | Linear | 0.6 | Benign | 96.75 | 98.42 | 97.57 | 96.00 |
| | | | | | | Malware | 92.31 | 85.16 | 88.59 | |
| | | | | | 0.9 | Benign | 96.70 | 98.40 | 97.54 | 96.00 |
| | | | | | | Malware | 92.76 | 85.98 | 89.24 | |
| | 4.25 | 175.13 | 132.31 | RBF | 9 | Benign | 97.74 | 98.33 | 98.03 | 96.94 |
| | | | | | | Malware | 94.12 | 92.15 | 93.12 | |
| | | | | | 22 | Benign | 97.66 | 98.53 | 98.09 | 96.94 |
| | | | | | | Malware | 93.94 | 90.64 | 92.26 | |
| | 3.79 | 162.68 | 136.22 | Poly | 30 | Benign | 97.07 | 97.79 | 97.43 | 95.88 |
| | | | | | | Malware | 91.02 | 88.37 | 89.68 | |
| | 3.64 | 167.60 | 138.15 | Sigmoid | 0.8 | Benign | 94.84 | 96.50 | 95.66 | 92.94 |
| | | | | | | Malware | 84.31 | 78.18 | 81.13 | |

750). Although Adaboost-RF presents the best results, this entails a weight in terms of computational resources, as it is possible to observe, in terms of the execution time in the best case, Adaboost-RF needs seven times more time and in the worst case 27 times more time to perform a classification. Regarding RAM, except for some exceptions, it uses approximately twice the amount. Regarding CPU, it uses less CPU than Random Forest.

We also compare our results with the ones reported by Taheri et al. [4] and Sangal and Verma [11], since these works presented a detailed description of the datasets and these are representative of current malware. Comparing our results with the ones reported in Taheri et al., the AB-RF algorithm reached a Precision higher than the precision of 95.3% obtained with the RF algorithm in the method reported by Taheri et al., for all scenarios, except D scenario using BoSC with size of 750. The Accuracy obtained with the AB-RF algorithm is also higher, for all scenarios herein considered, than the Accuracy of 96.05% reported by Sangal and Verma.

Unlike methods that predict the class labels of unknown examples using a single classifier induced from training data, such as Bayesian Classifiers, Artificial Neural Network, or SVM, ensemble methods construct a set of base classifiers from training data and perform classification by taking a vote on the

predictions made by each base classifier [21]. This explains why ensemble methods, such as Boosting and Random Forest, tend to perform better than any single classifier.

Main limitations of our work lie in the use of only Permissions and Intents as classification features.

Table 31: Best classification results obtained for each specified Scenario.

| Scenario | Dataset | Algorithm | Ac (%) | Pr (%) | Re (%) | F1 (%) | Ex. Time (s) | RAM (MB) | CPU (%) |
|---|---|---|---|---|---|---|---|---|---|
| A | Accounted CSV | AB-RF | 97.76 | 98.70 | 89.94 | 94.12 | 94.4852 | 852.02 | 100.98 |
| B | Normalized CSV | AB-RF | 98.24 | 99.31 | 91.14 | 95.05 | 95.547 | 843.32 | 101.28 |
| C | Accounted CSV | AB-RF | 97.29 | 97.95 | 87.73 | 92.56 | 19.75 | 218.56 | 107.57 |
| D | Normalized CSV | AB-RF | 97.41 | 96.89 | 90.17 | 93.41 | 19.85 | 227.09 | 107.58 |
| C | Accounted CSV | AB-RF | 97.76 | 98.78 | 90.50 | 94.46 | 23.16 | 247.8 | 106.4 |
| D | Normalized CSV | AB-RF | 97.76 | 94.33 | 92.36 | 93.33 | 23.01 | 249.13 | 106.45 |
| C | Accounted CSV | AB-RF | 97.76 | 96.60 | 91.03 | 93.73 | 27.50 | 274.42 | 105.77 |
| D | Normalized CSV | AB-RF | 98.12 | 97.56 | 93.02 | 95.24 | 28.16 | 285.77 | 105.26 |

Table 32: Best classification results of Random Forest for each scenario.

| Scenario | Dataset | Algorithm | Ac (%) | Pr (%) | Re (%) | F1 (%) | Ex. Time (s) | RAM (MB) | CPU (%) |
|---|---|---|---|---|---|---|---|---|---|
| A | Accounted CSV | RF | 97.06 | 96.67 | 87.88 | 92.06 | 11.9644 | 451.91 | 111.86 |
| B | Normalized CSV | RF | 97.41 | 96.03 | 90.06 | 92.95 | 11.6306 | 458.95 | 111.96 |
| C | Accounted CSV | RF | 97.18 | 95.27 | 89.24 | 92.16 | 2.09 | 130.04 | 175.68 |
| D | Normalized CSV | RF | 97.18 | 96.43 | 90.00 | 93.10 | 2.1 | 141.18 | 177.94 |
| C | Accounted CSV | RF | 97.65 | 96.60 | 90.45 | 93.42 | 2.48 | 150.11 | 164.95 |
| D | Normalized CSV | RF | 97.18 | 95.54 | 89.82 | 92.59 | 2.46 | 146.36 | 163.4 |
| C | Accounted CSV | RF | 97.29 | 94.34 | 91.46 | 92.88 | 2.91 | 164.93 | 154.55 |
| D | Normalized CSV | RF | 97.18 | 98.08 | 87.93 | 92.73 | 2.96 | 165.90 | 152.86 |

# 5 Conclusions

In this paper we investigated behavior-based malware detection in CICAndMal2017 dataset, which include recent and sophisticated Android malware samples. We used nine machine learning algorithms to classify the gathered features, Intents and Permissions, as benign or malware. We also explored the impact of the Bag of System Call algorithm in the classification results of those nine algorithms. The best classification results in the four scenarios were obtained by the Adaboost algorithm with base estimator Random Forest, leading to an Accuracy of 98.24%, a Precision of 99.31% (for malware) and a F1-Measure of 95.05% (for malware). The Random Forest algorithm reached an Accuracy of 97.41%, but with a smaller execution time (11.63 s against 95.547 s of Adaboost-RF), and nearly half of the RAM consumed (458.95 MB against 843.32 MB of Adaboost-RF), for a similar percentage of the use of CPU (111.96% against 101.28% of Adaboost-R). For future work we plan to optimize the size of the Bag of System Calls and to investigate the use of a sliding window for performance evaluation of the classifiers.

# 6 Acknowledgments

This work is funded by Portuguese FCT/MCTES through national funds and, when applicable, co-funded by EU funds under the project UIDB/50008/2020 and by FCT/COMPETE/FEDER under the project SECURIoTESIGN with reference number POCI-01-0145-FEDER-030657, and funded by operation Centro-01-0145-FEDER-000019 - C4 - Centro de Competências em Cloud Computing, co-funded by the European Regional Development Fund (ERDF/FEDER) through the Programa Operacional Regional do Centro (Centro 2020).

# References


[1] Cisco. Cisco Annual Internet Report (2018–2023) white paper, March 2020. https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html [Online; accessed on December 15, 2021].

[2] Cisco. Android statistics (2021), November 2021. https://www.businessofapps.com/data/android-statistics/ [Online; accessed on December 15, 2021].

[3] R. Casolare, C. D. Dominicis, G. Iadarola, F. Martinelli, F. Mercaldo, and A. Santone. Dynamic mobile malware detection through system call-based image representation. *Journal of Wireless Mobile Networks Ubiquitous Computing Dependable Applications*, 12(1):44–63, March 2021.

[4] L. Taheri, A. F. A. Kadir, and A. H. Lashkari. Extensible android malware detection and family classification using network-flows and api-calls. In *Proc. of the 53th International Carnahan Conference on Security Technology (ICCST'19), Chennai, India*, pages 1–8. IEEE, October 2019.

[5] S. Feldman, D. Stadther, and B. Wang. Manilyzer: Automated android malware detection through manifest analysis. In *Proc. of the 11th IEEE International Conference on Mobile Ad Hoc and Sensor Systems (MASS'12), Philadelphia, Pennsylvania, USA*, pages 767–772. IEEE, October 2014.

[6] F. Idrees and M. Rajarajan. Investigating the android intents and permissions for malware detection. In *Proc. of the 10th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob'14), Larnaca, Cyprus*, pages 354–358. IEEE, October 2014.

[7] J. Song, C. Han, K. Wang, J. Zhao, R. Ranjan, and L. Wang. An integrated static detection and analysis framework for android. *Pervasive and Mobile Computing*, 32:15–25, October 2016.

[8] F. Idrees, M. Rajarajan, M. Conti, T. M. Chen, and Y. Rahulamathavan. Pindroid: A novel android malware detection system using ensemble learning methods. *Computers & Security*, 68:36–46, July 2017.

[9] F. Idrees, M. Rajarajan, T. M. Chen, Y. Rahulamathavan, and A. Naureen. Andropin: Correlating android permissions and intents for malware detection. In *Proc. of the 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON'17), Vancouver, British Columbia, Canada*, pages 394–399. IEEE, October 2017.

[10] A. Feizollah, N. B. Anuar, R. Salleh, G. Suarez-Tangil, and S. Furnell. Androdialysis: Analysis of android intent effectiveness in malware detection. *Computers & Security*, 65:121–134, March 2017.

[11] A. Sangal and H. K. Verma. A static feature selection-based android malware detection using machine learning techniques. In *Proc. of the 1st International Conference on Smart Electronics and Communication (ICOSEC'20), Trichy, India*, pages 48–51. IEEE, September 2020.

[12] V. Kouliaridis, G. Kambourakis, and T. Peng. Feature importance in android malware detection. In *Proc. of the 19th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom'20), Guangzhou, China*, pages 1449–1454. IEEE, December 2020.

[13] K. Khariwal, J. Singh, and A. Arora. Ipdroid: Android malware detection using intents and permissions. In *Proc. of the 4th World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4'20), London, UK*, pages 197–202. IEEE, July 2020.

[14] Cisco. What is malware?, September 2021. https://www.cisco.com/c/en/us/products/security/advanced-malware-protection/what-is-malware.html#~what-is-malware [Online; accessed on December 15, 2021].

[15] Google. Android permissions usage-notes, May 2021. https://developer.android.com/training/permissions/usage-notes [Online; accessed on December 15, 2021].

[16] Google. Android intents, October 2021. https://developer.android.com/guide/components/intents-filters [Online; accessed on December 15, 2021].

[17] Apktool, September 2021. https://ibotpeaches.github.io/Apktool/ [Online; accessed on December 15, 2021].

[18] Google. Manifest permission, 2021 December. https://developer.android.com/reference/android/Manifest.permission [Online; accessed on December 15, 2021].

[19] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and Édouard Duch-

esnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(85):2825–2830, January 2011.

[20] Scikit-Learn. Machine learning in python, October 2021. `https://scikit-learn.org/stable/` [Online; accessed on December 15, 2021].

[21] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Pearson Education, July 2006.

[22] D.-K. Kang, D. Fuller, and V. Honavar. Learning classifiers for misuse and anomaly detection using a bag of system calls representation. In *Proc. of the 6th Annual IEEE SMC Information Assurance Workshop (IAW'05), West Point, New York, USA*, pages 118–125. IEEE, June 2005.

[23] N. Antunes and M. Vieira. On the metrics for benchmarking vulnerability detection tools. In *Proc. of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'15), Rio de Janeiro, Brazil*, pages 505–516. IEEE, June 2015.

---

## Author Biography

**André Prata Ferreira** received the bachelor's degree in computer science and engineering from the University of Beira Interior, in 2011 and the master's degree in computer science and engineering in 2013, and also finished a Post-Graduation in Information and Communication Tecnologies for the Telecommunications Sector in 2014. Currently he is persuing Ph.D. degree at the University of Beira Interior under the title Malware Detection in Android Platforms Using Behaviour Analysis. His main research and interest areas are Android, malware detection, cybersecurity, and automation.

No photo is available

**Chetna Gupta** received the Bachelor in Engineering, Master in Technology, and Ph.D. degrees in Computer Science & Engineering. She is currently a post-doctoral researcher in the University of Beira Interior, Covilhã, Portugal. She has authored or co-authored several journal, conference papers, book chapters, and books. Her current research interests include cloud computing, information management, and the application of machine learning techniques to address concerns in cloud systems.

**Pedro R. M. Inácio** is an associate professor of the Faculty of Engineering at the Universidade da Beira Interior (UBI), which he joined in 2010. He lectures subjects related with information assurance and security, programming of mobile devices and computer based simulation, to graduate and undergraduate courses, namely to the B.Sc., M.Sc. and Ph.D. programmes in Computer Science and Engineering. He is currently a Pro-Rector for the Digital University of UBI. He is an instructor of the UBI Cisco Academy. He holds a 5-year B.Sc. degree in Mathematics/Computer Science and a Ph.D. degree in Computer Science and Engineering, obtained from UBI, Portugal, in 2005 and 2009 respectively. The Ph.D. work was performed in the enterprise environment of Nokia Siemens Networks Portugal S.A., through a Ph.D. grant from the Portuguese Foundation for Science and Technology. He is an IEEE senior member, an ACM professional member and a researcher of the Instituto de Telecomunicações (IT). His main research topics are information assurance and security, computer based simulation, and network traffic monitoring, analysis and classification. He has 50+ publications in the form of book chapters and papers in international peer-reviewed books, conferences

and journals. He frequently reviews papers for IEEE, Springer, Wiley and Elsevier journals. He has been a member of the Technical Program Committee of national and international workshops and conferences, such as the ACM Symposium on Applied Computing - Track on Networking. Currently serves as an Associate Editor for IEEE Access.

**Mário M. Freire** received the five-year BS degree in Electrical Engineering and the two-year MS degree in Systems and Automation in 1992 and 1994, respectively, from the University of Coimbra, Portugal. He received the PhD degree in Electrical Engineering in 2000 and the Habilitation title in Computer Science in 2007 from the University of Beira Interior (UBI), Portugal. He is a full professor of Computer Science at UBI, which he joined in the Fall of 1994. In April 1993, he did one-month internship at the Research Centre of Alcatel-SEL (now Nokia Networks) in Stuttgart, Germany. His main research interests fall within the area of computer systems and networks, including network and systems virtualization, cloud and edge computing and security and privacy in computer systems and networks. He is the co-author of seven international patents, co-editor of eight books published in the Springer LNCS book series, and co-author of about 130 papers in international journals and conferences. He serves as a member of the editorial board of the ACM SIGAPP Applied Computing Review, serves as associate editor of the Wiley Security and Privacy journal and of the Wiley International Journal of Communication Systems, and served as editor of IEEE Communications Surveys and Tutorials in 2007–2011. He served as a technical program committee member for several IEEE international conferences and is co-chair of the track on Networking of ACM SAC 2021. Dr. Mário Freire is a chartered engineer by the Portuguese Order of Engineers and he is a member of the IEEE Computer Society and of the Association for Computing Machinery.