Hindawi
Security and Communication Networks
Volume 2019, Article ID 2850932, 13 pages
https://doi.org/10.1155/2019/2850932

Research Article

Improved Malware Detection Model with Apriori Association Rule and Particle Swarm Optimization

Olawale Surajudeen Adebayo [1,2] **and Normaziah Abdul Aziz** [1]

[1] *Computer Science Department, International Islamic University Malaysia, Malaysia*
[2] *CSS Department, Federal University of Technology Minna, Nigeria*

Correspondence should be addressed to Olawale Surajudeen Adebayo; waleadebayo@futminna.edu.ng

Received 8 January 2019; Revised 3 April 2019; Accepted 2 July 2019; Published 8 August 2019

Guest Editor: Jose M. Gimenez-Guzman

The incessant destruction and harmful tendency of malware on mobile devices has made malware detection an indispensable continuous field of research. Different matching/mismatching approaches have been adopted in the detection of malware which includes anomaly detection technique, misuse detection, or hybrid detection technique. In order to improve the detection rate of malicious application on the Android platform, a novel knowledge-based database discovery model that improves apriori association rule mining of a priori algorithm with Particle Swarm Optimization (PSO) is proposed. Particle swarm optimization (PSO) is used to optimize the random generation of candidate detectors and parameters associated with apriori algorithm (AA) for features selection. In this method, the candidate detectors generated by particle swarm optimization form rules using apriori association rule. These rule models are used together with extraction algorithm to classify and detect malicious android application. Using a number of rule detectors, the true positive rate of detecting malicious code is maximized, while the false positive rate of wrongful detection is minimized. The results of the experiments show that the proposed a priori association rule with Particle Swarm Optimization model has remarkable improvement over the existing contemporary detection models.

1. Introduction

Malware classification and detection using data mining technique is a significant area in the detection of malicious applications. This technique of detection can be classified into supervised and unsupervised learning strategies and several techniques [1]. The strategy or technique to be used by malware detection expert depends on the nature of data and problem to be solved, that is whether the output of the data is categorical or numerical. There are several machine learning techniques for supervised data mining which includes support vector machine, neural network, rule based, decision tree, and Naïve Bayes, while unsupervised learning technique is based on clustering algorithm like k-nearest neighbour, association rules, and some other clustering algorithms. Supervised learning can be basically used for three purposes, namely, classification, prediction, and estimation depending on the output of data or whether to determine present or future circumstances.

Both supervised and unsupervised learning techniques were used in this research for experimentation. The supervised learners used include Bayesian classifier, Naïve Bayes, PART, decision tree, random forest, neural network, and Classification-Based Multiple Association Rule. The data mining unsupervised technique used is association rules mining of apriori algorithm which was improved and adopted for signature extraction and malware detection. Since the signature is being generated automatically after improving the rules, the model is capable of detecting known and unknown malware. Original a priori algorithm was proposed by Agrawal R. et al. [2] in order to address the problem of mining association rules. The need to improve the efficiency of mining of frequent itemsets, by reducing the times of scanning the database and reducing the number of candidate itemsets, prompted [3] to propose an improved a priori algorithm based on the classic apriori algorithm. The basic idea of apriori algorithm is to generate the frequent itemsets using iterative method in order to generate rules that

meet the minimum confidence to form rule sets and outputs [3].

This research purposely adopts particle swarm optimization in order to improve the generation of candidate detectors (flagbearers) which will otherwise improve the detection process by maximizing the true positive detection and minimizing the false positive detection. The research gathered several android applications both good and malicious for the purpose of classification and detection. The features were extracted from both samples after the thorough analysis of code samples. Three feature selection approaches were used to select high ranked features from the set of generated features. The association rules were generated based on the features, which were used for the detection of malicious android application.

The major contributions of this research are (1) improvement in the generation of candidate detectors using particle swarm optimization on apriori algorithm, (2) generation of rules using candidate detectors for the classification of android applications into malicious or benign, and (3) extraction of signatures for the detection of malicious applications. The rest of this paper is organized as follows: related works to this research are discussed in section two. Section three discusses the proposed model with its constituent frameworks including apriori algorithm, AAR-aPSO, and detection model using new rule AAR-aPSO. In section four, empirical study, results, and conclusion were given to the work. Section five is used to explain the experimental study and discussion. Section six is used to conclude the work with adequate recommendations.

2. Related Work

Computer codes with malicious inclinations are referred to as malware. Effort to detect, analyze, and remove malware from computing gadgets remains a huge demand across the globe. Various methods and algorithms have been worked out by many researchers to carry out this task, such as in [4–8], among others. The initial problem of mining association rules was addressed by [2] in apriori algorithm where the generated frequent itemsets were used to generate rules that meet the minimum confidence to form rule sets and output. Association rules mining has been used in the previous work [5] to generate frequent itemsets of program signatures (malware and benign) and extract features from the parsed files for subsequent supervised learning. This work designed a malware detection algorithm using association rules of apriori algorithm for feature extraction and signature generation of window executable files. This detector however has limitations due to the slow generation of candidate detectors by apriori algorithm. This limitation was removed in this research by improving association rules with PSO. This system is also designed effectively for android mobile devices. Other researches in [9–12] have attempted to provide solution to the association problem of detecting malware using apriori algorithm of association rule mining. In order to improve the generation of candidate detectors that form rule for signature extraction and feature selection, particle swarm optimization was used. This algorithm [13] was developed by Eberhart and Kennedy to provide solution to the problem of optimization. The problem was modeled towards the behavior of a group of birds searching for food and follows a particular bird which is nearest to the food. This algorithm, PSO, has been applied successfully for the generation of candidate detector in negative selection algorithm for spam detection [14, 15], virus detection [16], feature selection [17–19], anomaly detection [20, 21], and intrusion and misuse detection [11, 12]. PSO has also proved to be a successful optimizer in fuzzy system [12], multiobjective problems [22], and tracking system [23], fuzzy rule learning [24], and classifying malicious android applications [25].

Due to the inefficiency of apriori algorithm in generating large itemset, its different versions have been developed to improve the original algorithm such as in [3, 10]. The research in this present paper makes effort to improve apriori association rule for the detection of malicious programs. The research adopts particle swarm optimization in the candidate generation of detector so as to increase the detection of malicious application and reduce false alarm rate. Malicious application is an application program designed primarily by malware authors to hamper the normal operation of a mobile phone or stealing the vital information of a user. Obfuscation [26] is a very common technique adopted by malware writer; it is a set of codes which prevent detection by the detectors. Analysis of malicious application could be static, dynamic, or hybrid. Static analysis is the process of analyzing a program's code statistically without actually executing the code [27]. It scans the software for malicious patterns without installing it. In the static analysis on a smartphone, the sandbox decompresses installation files and disassembles corresponding executable. Dynamic analysis on the other hand executes the application in a fully isolated environment, i.e., sandbox [28], which intervenes and logs low-level interactions with the system for further analysis using Android emulator. Dynamic analysis was based on some heuristics such as the monitoring of modifications to the system registry and the hooks' insertion into system interface or library. Dynamic analysis however is disadvantaged with the lack of fundamental attributes of malware for analysis, which subject its results to high false positive and false negative rates. The behavioural malware detection on mobile handset to reduce malicious casualty in the mobile community is another detection technique by [26]. This approach is unique in the definition of application behaviour. It observes the programs' run-time behaviour at a higher level (i.e., system events or resource-access) than system calls of [29] and machine instructions of [6]. In addition, the approach used a runtime analysis, effectively bypassing the need to deal with code/data obfuscation [7]. The recent researches on the detection of malware on mobile platform include [7, 8, 28, 30–36].

3. The Proposed Improved Model and Its Associated Frameworks

The application of improved systems in solving the real-world problems remains a huge success. This proposed improved system improved apriori association rule using particle

swarm optimization through data mining technique with modified function as fitness function for signature extraction and malware detection. Mining association rule is used in malware detection system to extract important signatures from the collected features and to discover useful association rules in the signature. This task (mining association) is basically to discover the large itemsets that is the sets of items that have support above a predetermined threshold, and use the large itemsets to generate the signature rules for the features that have confidence above a predetermined threshold [1]. The apriori association rule, the rule mining technique to be improved for this research, is an implication expression of the form $X_1 \longrightarrow X_2$, where X_1 and X_2 are disjoint itemsets such that $X_1 \cap X_2 = \emptyset$ [37]. The strength of the rule $X_1 \longrightarrow X_2$ can be measured by the support and confidence of the rule. Support determines the rate at which a rule is applicable to a dataset, while confidence determines how frequently the items in X_2 appear in transactions that contain X_1. Strong association rules have their confidence value c above given threshold. For example, for a certain rule $(x_1, x_2) \longrightarrow x_3$, then it is a precondition for both itemsets (x_1, x_2) and (x_1, x_2, x_3) to be frequent. The confidence (c) is then computed to determine the rule that is strong using $c = S(x_1, x_2, x_3)/s(x_1, x_2)$. Apriori association rule is a two-step process [37]. The first step generates candidate itemsets, while the second step uses the generated itemsets to create a set of association rules, which meet up with the required support and confidence.

Association rules were generated from the selected candidates using apriori association rules and combined apriori association rules with Particle Swarm Optimization. These rules were used to extract signatures from the features of android applications for the detection of malicious or benign applications. The components and collection of an improved model are discussed in the subsequent subsections. Association rule of frequent itemsets has been used in [38] for the detection of HTTP botnet, detection of executable [39], and detection of malicious executable in the wild [40]. Figure 1 is the flowchart of the association rule of apriori algorithm.

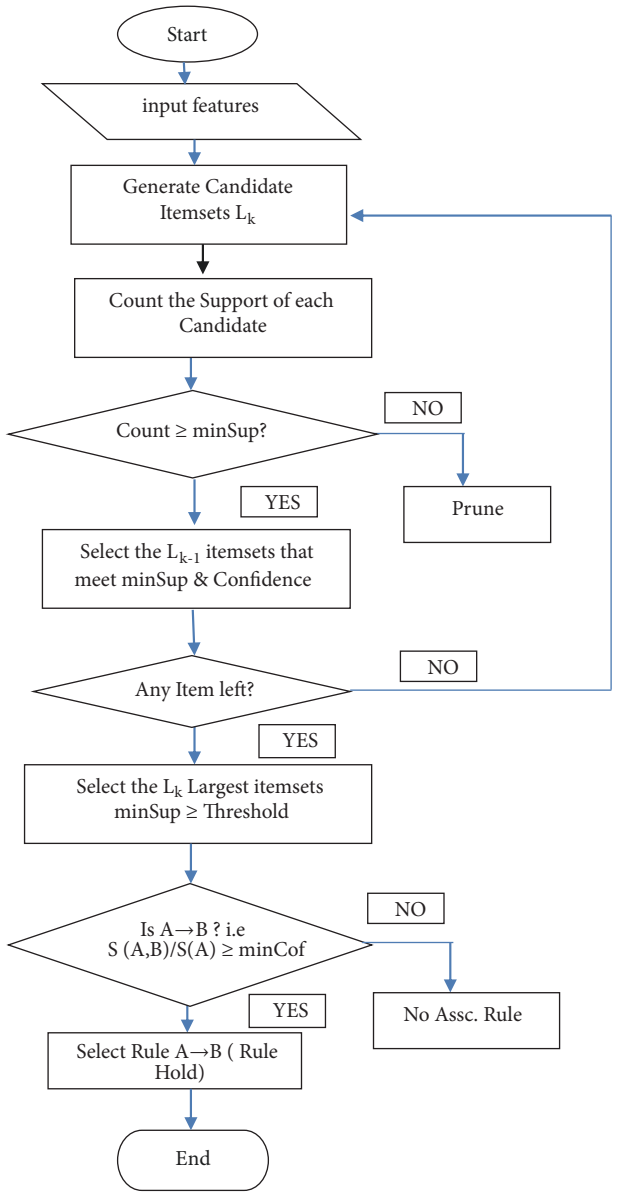

Figure 1: Association rule of apriori algorithm.

3.1. Optimization of Apriori Algorithm Candidate Generator with Particle Swarm Optimization (AA-PSO). The most important task in apriori algorithm is candidate generation of large k-itemsets with highest frequency and the association of rules. The problem is to generate large k-itemsets that meet the minima support and confidence in a short period of time with efficiency. This paper presents a technique to optimize the generation of large k-itemsets using adaptive Particle Swarm Optimization (aPSO) in order to increase the effectiveness of signature extraction and malware detection model. The particle's velocity and position in an updated standard PSO were given in (1) and (2), respectively, below:

$$\mathbf{V_{id}}(\mathbf{t}+\mathbf{1}) = \mathbf{w}\mathbf{V_{id}}(\mathbf{t}) + \mathbf{c_1}\mathbf{r_1}\left(\mathbf{P_{id}}\mathbf{best}(\mathbf{t}) - \mathbf{x_{id}}(\mathbf{t})\right) + \mathbf{c_2}\mathbf{r_2}\left(\mathbf{P_{gd}}\mathbf{best}(\mathbf{t}) - \mathbf{x_{id}}(\mathbf{t})\right) \quad (1)$$

$$\mathbf{X_{id}}(\mathbf{t}+\mathbf{1}) = \mathbf{x_{id}}(\mathbf{t}) + \mathbf{V_{id}}(\mathbf{t}+\mathbf{1}) \quad (2)$$

where $i = 1, 2, \ldots, n$, n represent the number of particles in the swarm, $d = 1, 2, \ldots, D$, D is the dimension of solution space. $w \epsilon [0, 1]$ is the inertia weight associated to the given particle velocity and position to ensure balance between the local and global search best. Also c_1 and c_2 represent the nonnegative learning factor, while r_1 and r_2 uniformly distributed random numbers in the interval $[0, 1]$. The velocity $V_{id} \epsilon [-V_m, V_m]$, where V_m is a maximum velocity predefined by the users in relation to the objective function. In this paper, fitness function in [17] was used with little modification.

(i) Particle Swarm Optimization. Particle swarm optimization (PSO) was developed by Eberhart and Kennedy in 1995 [13] purposely to solve the problems of optimization. Since

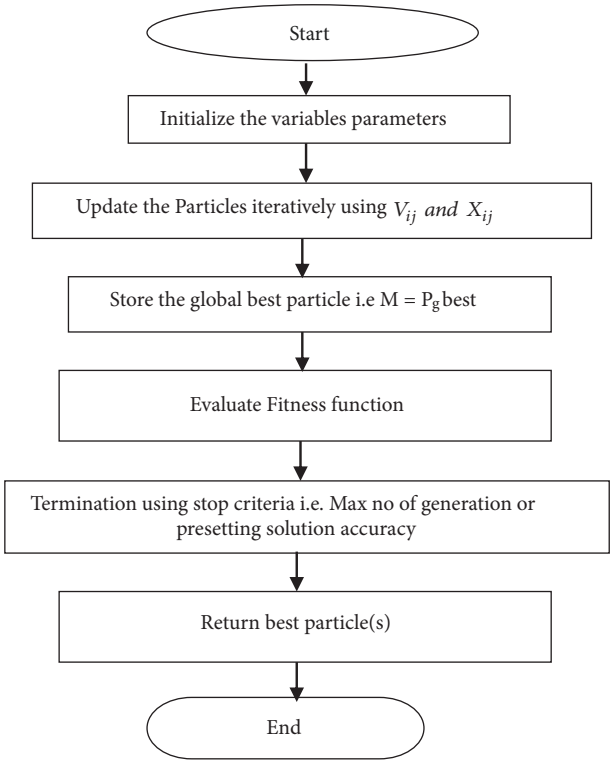

Figure 2: Particle Swarm Optimization Algorithm.

then, variants of PSO have emerged in a bid to improve the performance of original PSO. The variants of PSO include [41], which introduces inertia weight into PSO; others are [42, 43], CPSO [44], etc. The particles which have potential solutions [19] move around the solution space and following the current optimum particles. All other particles in the space do therefore follow the particle with potential solutions that have particle's best position. PSO was used to provide robustness to the generation of large candidate itemsets. The flowchart of the representation of PSO is given in Figure 2.

In this research, an adaptive Particle Swarm Optimization (aPSO), which knows and can transmit a bit more local information and a global one (swarm size), was used.

A permission-based feature f of an android application represents a particle in an N-dimensional binary space.

The ith particle (feature) is represented as

$$X_{ij} = \left(X_{i1,X_{i2},\ldots X_{iN}}\right) \tag{3}$$

The best previous position that gives the best fitness values (P_{best}) of a particle in the application is given as

$$P_i = \left(P_{i1,P_{i2},\ldots P_{iN}}\right) \tag{4}$$

The best particle among all particles in the application (*global best*) = G_{best} is given as

$$V_{ij} = \left(V_{i1,V_{i2},\ldots V_{iN}}\right) \tag{5}$$

The rate of change of a position particle for a particle I called velocity is represented by equation (6):

$$v_{ij}(k+1) = w * v_{ij}(k) + c_1 * r_1\left(p_{ij} - x_{ij}\right) + c_2 * r_2 * \left(Gbest - x_{ij}\right) \tag{6}$$

$$x_{ij}(k+1) = x_{ij}(k) + v_{ij}(k+1) \tag{7}$$

The velocity is updated using (6), while (7) is used to update particles position by particle swarm optimization, where $x_{ij}(k)$ *and* $v_{ij}(k)$ are the initial position and velocity, respectively, and j = 1, 2, ... represent the social and cognitive factors associated with particle movement. w is an inertia weight, a positive function that balances the particle movement between global and local particle exploration. r_1 *and* r_2 (0, 1) are random functions between 0 and 1 that keep the particle position changing in a random distribution. Velocity V_{ij} has both maximum and minimum velocity, v_{max} and v_{min}, respectively. The maximum velocity limits the movement of particle within the search space, so that the particle does not fly above solution space, while the minimum velocity determines the lowest level a particle can fly to. The maximum velocity must not be too high so as to ensure the particle does not fly past good solutions and must not be too low so as not to confine particle search to a local minima.

4. Proposed Model Framework

This research shows that apriori association rule can be improved as a selection technique using particle swarm optimization. The existing detection algorithm uses apriori association analysis for its signature extraction which was characterized with shortcomings. This research used apriori association rule that has been improved with particle swarm optimization (AAR-aPSO) in order to improve the effectiveness and efficiency of the signature extraction and model performance in the detection of malicious android application. This proposed system consists of detection algorithm together with an improved apriori association rule with particle swarm optimization model. The rule model consists of malicious and benign android application signatures generated from the selected candidates of apriori algorithm with particle swarm optimization for better detection of new benign or malicious signatures.

Figures 3 and 4, respectively, show an improved apriori association rule with particle swarm optimization and detection algorithm with an improved model (AAR-aPSO)

4.1. Malware Detection Model Using AAR-aPSO. A new model for the detection of malware was built with a combined apriori and adaptive particle swarm optimization. The model simply substituted apriori association rule with apriori association rule-adaptive PSO (AAR-aPSO) in the detection algorithm. Feature subset length and classification qualification quality were used to calculate fitness function that measure the quality of candidate to be generated.

4.2. Fitness Function. The fitness function in [17] was adopted in this research with little modification. The fitness function

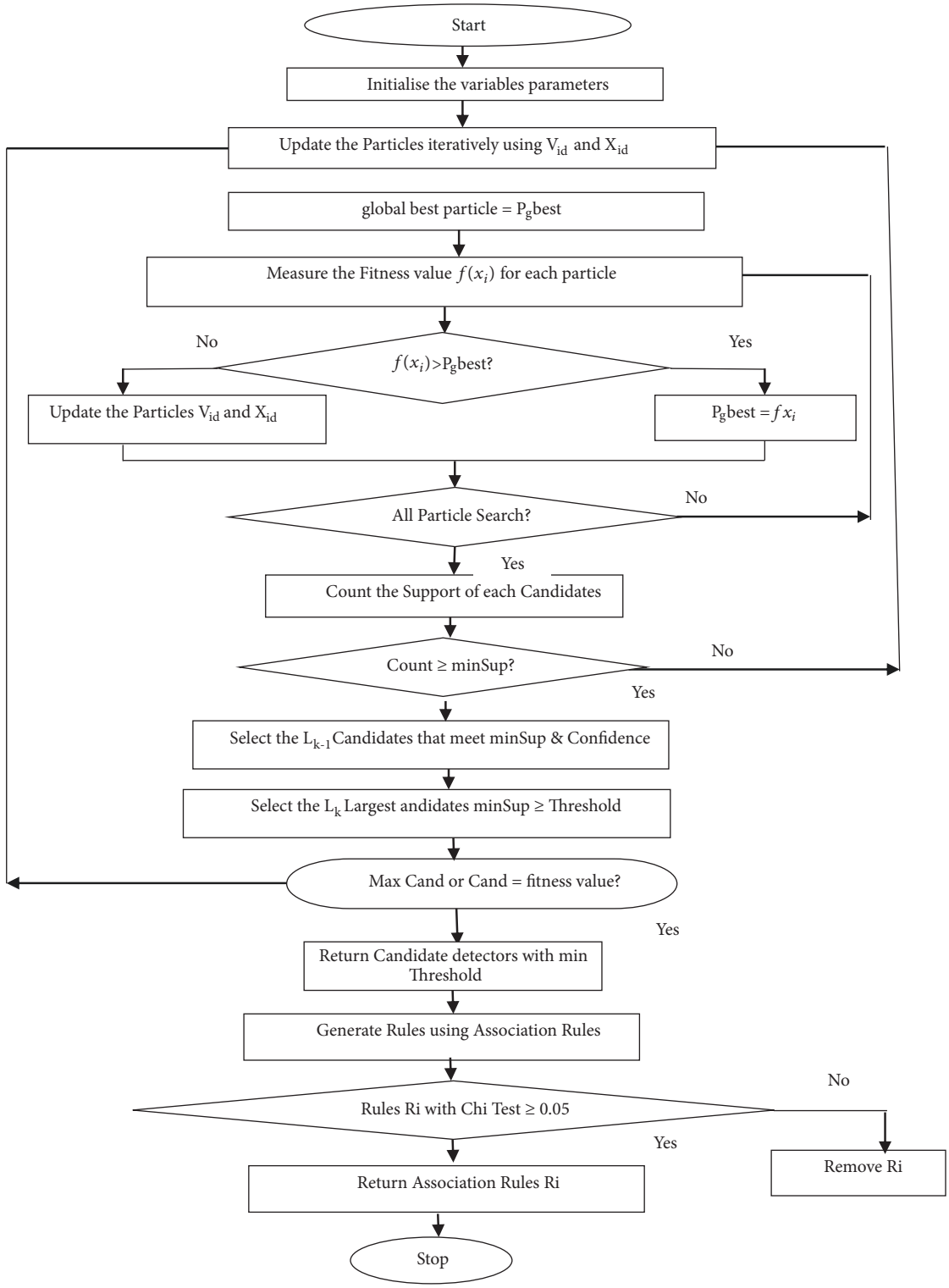

Figure 3: Proposed Improved AAR-Particle Swarm Optimization candidate generation model. Figure 3 is reproduced from O. S. Adebayo & N. A. Abdul Aziz (2014) (under theCreative Commons Attribution License/public domain). Source link: https://ieeexplore .ieee.org/abstract/document/7077314.

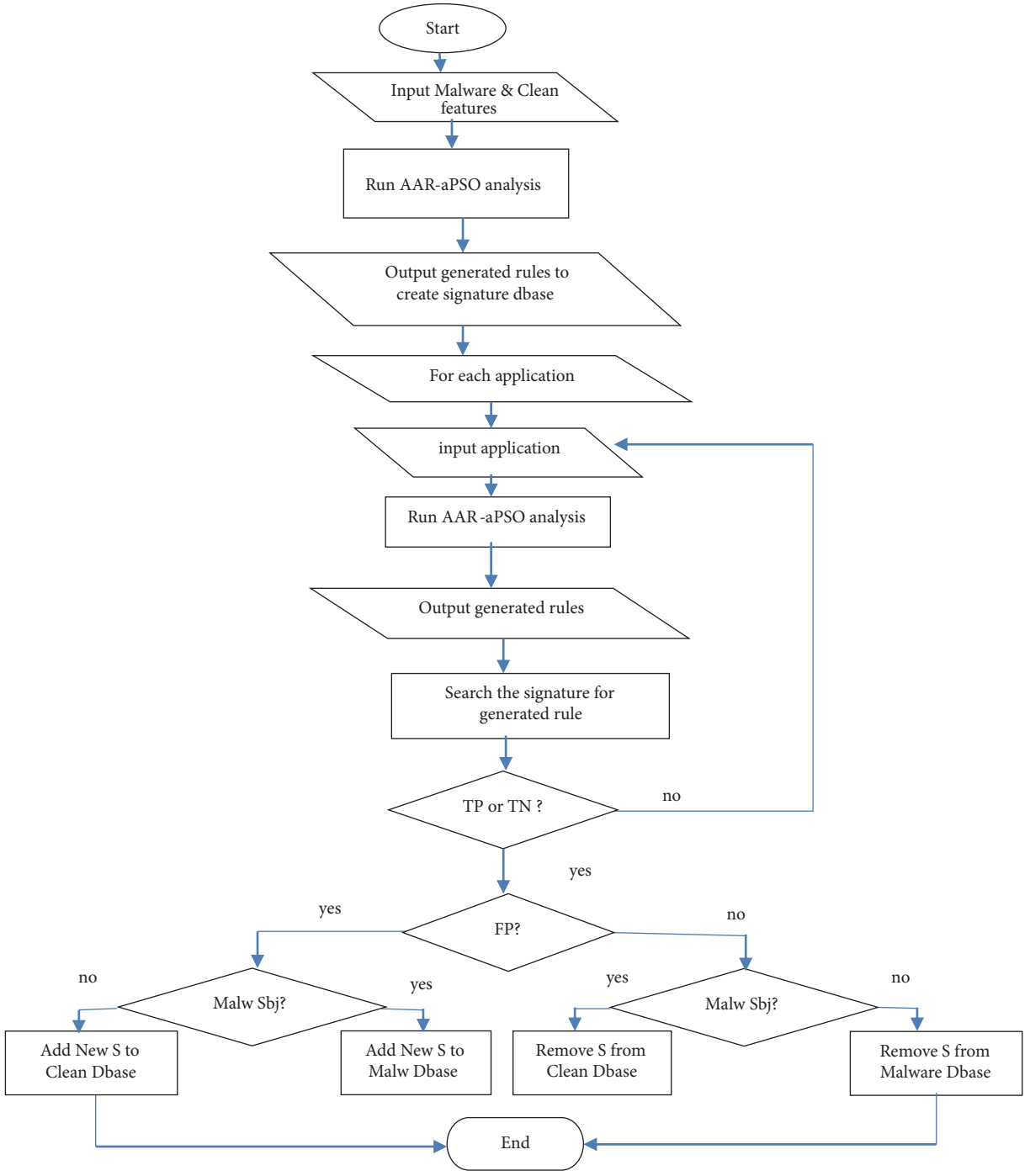

Figure 4: Proposed AAR-aPSO Malware detection model.

$f(x)$ used feature length, feature population, and feature classification quality and is defined as

$$f(x) = \varphi \Upsilon_{\rho} + \lambda \frac{\left|\partial_{fl}\right| - \left|\rho_{fl}\right|}{\partial_{fl}} \tag{8}$$

where φ is the classification qualityλ is the feature subset length, ∂_{fl} is the total number of features in the dataset, ρ_{fl} is the length of a selected particle, Υ_{ρ}, the classification quality of selected particle, is the ratio of selected particle feature length ρ to the total number of features ∂_{fl}.

The classification quality φ and feature subset length λ are two numeric constant values within the range $[0, 1]$, which is based on the significance of each value. The classification quality value is believed to be of more importance and was assigned 0.95, while feature subset length value was assigned 0.05. The value ∂_{fl} is the total number of features contained in the entire dataset, while ρ_{fl} is the length or number of

features in the selected particle or particle with P_{best}. Υ_ρ is the classification quality of selected particle, which represents the ratio of selected particle feature length ρ to the total number of feature ∂_{fl}.

$$\Upsilon_\rho = \frac{\rho_{fl}}{\partial_{fl}} \tag{9}$$

And weight of a velocity V_{ij} is defined as

$$w = w\Upsilon_\rho * \frac{maxIter - Iter}{maxIter} \tag{10}$$

where $maxIter$ is the highest given iteration to terminate the optimization process and $maxIter$ is the current value of iteration. The value of weight depends on the classification quality of selected particle.

5. Empirical Study, Results, and Conclusion

In order to test the validity of the research's hypotheses in Section 5.2, malicious and benign Android applications were gathered from contagiominidump [45] and Google Play [46], respectively. Supervised and unsupervised learning experiments were carried out to compare the best results. Stratified sampling technique was used to create training and test dataset for better representation for supervised learning algorithms. Apriori association rules (AAR), AAR with aPSO model (AAR-aPSO), classification multiple rule, and FP-Growth were used to generate rules for unsupervised learning experiment. Holdout evaluation technique was used where dataset was partitioned into 70% training and 30% test data. Both training and test set were set of features extracted from.apk files. The training data was used to train the model, while the test set was used to test the performance of the model. Seven classification algorithms were used for supervised learning with the extracted features of android applications. The entire empirical process was discussed in the following subsections.

5.1. Dataset Description and Analysis. The steps in the empirical process include data collection, Android application analysis, disassembling, features extraction, feature selection, independent test on the dataset, and classification model building. Set of android.apk files were collected for both clean and malicious programs. The programs were made up of 1000 malicious applications from contagiominidump and 500 benign applications from official android market Google Play representing 66.7% and 33.3%, respectively. In order to analyze the dataset, static analysis in [47, 48] was adopted using combination of tools. After this initial experiment, researchers were able to access the source code of the program and useful features were collected.

File analysis was carried out using stratified sampling technique on the entire programs to balance the number of extracted features from malware and benign applications. After the partitioning of the data, each file is parsed and a vector equivalent to each file was defined as its representative feature.

In order to extract best features from the disassembled parsed files, frequent feature structures were search globally in the entire data collection using the combined apriori and PSO algorithm. The combined model was used to generate association rules from set of best features. The mining was done using between 20% to 50% support and 80% confidence on the partitions which yields separate rules for malicious and benign dataset. The sum of sixty-seven rules ($M \cup B$) were generated in all comprising twelve (12) rules for benign applications ($B \setminus M \cap B$), fifty-seven (57) malicious rules ($M \setminus M \cap B$), and two (2) rules common to both malicious and benign applications ($M \cap B$). Rules common to both malicious and benign were treated as malicious applications in order to enhance true positive rate. The combined rules generated from both malicious and clean Android applications are 67 rules.

These rules were used together with the signature extraction algorithm to classify applications into malware or benign. In order to select the best rules from the entire set of rules, a signature rule found only in a single class was defined and removed in order not to reduce the detector into signature based detection. Due to the large number of features extracted, which might become redundant to the system, unnecessary features were removed leaving us with moderate feature and were selected using apriori algorithm, and new model AA-PSO algorithm selection technique. Final dataset was represented using a vector space model where each android application was a vector in an n-dimensional space with n a number of selected features. A binary variable was defined to represent a malicious, benign application and target variable (malicious or benign application).

Statistical test using p-value was carried out on the features to examine the existence of relationship or otherwise on the feature and final class value. Those features that did not show any significant relationship with the target variable were removed from the dataset. The results of classification using supervised techniques with different classification algorithms are presented in Table 4. In unsupervised learning experiment, number of runs of detection algorithm using apriori association rule, classification multiple rule, FP-Growth rule, and new model apriori association rule with particle swarm optimization (AAR-aPSO) results are presented in Table 5. Table 1 displays the quantity of android application acquired for the extraction of features, while Table 2 presents the percentages of the.apk files found in both benign and malicious android applications (.apk files).

5.2. Feature Description. The features extracted are permission-based android features. These features are classified into three categories, viz., normal, dangerous, and signature. The samples of selected permission features are shown in Table 3, while Table 4 displays the statistics of selected features using model AA-PSO. Table 7 describes the permission structure of numeric data that is the formulation of independent and class variables.

5.3. Criteria for Performance Evaluation. The criteria for measuring the performance of the proposed method were based on a basic research questions upon which research

Table 1: Malware and Clean Applications.

File type	Source	Quantity	Percentage	Usage
Benign Apps	Googleplay	500	33.3	Analysis
Malicious Apps	Contagiominidump, contagiodump	1000	66.7	Analysis
Total		1500	100	

Table 2: Percentages of .apks files.

Android applications (.apk files)	Quantity	Percentage (%)
.apks used (Benign & Malicious)	1500	100
Valid .apk with features	1420	94.66667
Invalid .apks without features	80	5.333333
Malicious .apk files	1000	66.66667
Benign .apks	500	33.33333
Valid Malicious .apk	950	66.90141
Valid Benign	470	33.09859
Invalid Malicious	50	62.5
Invalid Benign	30	37.5

Table 3: Sample of Selected Permission features.

Permission Features	Assigned value
android.permission.ACCESS_FINE_LOCATION	2
android.permission.INTERNET	3
android.permission.WRITE_CONTACTS	4
android.permission.READ_PHONE_STATE	5
android.permission.USE_CREDENTIALS	6
android.permission.DISABLE_KEYGUARD	7
android.permission.WAKE_LOCK	8
android.permission.CAMERA	9
android.permission.CHANGE_CONFIGURATION	10

Table 4: Selected Features Statistics.

Features	Quantity
Single feature remove from Malicious	37
Single feature remove from Benign	81
Single feature remove from both Malicious & Benign	0
Malicious (M)	128
Benign (B)	119
Malicious Alone ($M \setminus M \cap B$)	54
Benign Alone ($B \setminus M \cap B$)	45
M & B ($M \cap B$)	74
M OR B ($M \cup B$)	173
Class (C)	1
Feature used for Vector analysis	174

hypotheses were defined and were done through the use of statistical quality measures usually used in machine learning.

5.3.1. Research Questions. The research questions on which the proposed model was evaluated are as follows:

(1) Can the detection rate of malicious applications be improved using an improved apriori association rule with Particle Swarm Optimization?

(2) Is new AAR-aPSO unsupervised model better than other supervised and rule models?

5.3.2. Research Hypotheses. The research hypotheses based on the research questions are as follows:

(H1) The detection rate of malicious applications is not significantly improved using an improved AAR.

The detection rate of malicious application is significantly improved using an improved AAR.

(H2) An improved AAR-aPSO unsupervised model is not better than supervised and other rule models.

An improved AAR-aPSO unsupervised model is better than supervised and other rule models.

Statistical Test. The statistical tests used to evaluate the performance of supervised learners and unsupervised learning using apriori association rules, CMR, FP-Growth, and apriori association rules with Particle Swarm Optimization (AAR-aPSO) in the detection of malicious android application, include Accuracy (ACC), True positive rate (which measure sensitivity), True negative rate, False positive rate (specificity), and root mean square error.

(i) The Accuracy Measure. In order to measure the accuracy, a confusion matrix table upon which the accuracy definition was based was formulated.

	True Accept (P)	*False Reject (N)*
True (T)	TP	TN
False (F)	FP	FN

TP (True positive) was defined as the android malicious application that was actually classified as malware; i.e., *TPR* is the proportion of positive instances classified correctly.

Table 5: Average Accuracy, detection, and error rates of supervised learning experiment.

Models	Avg. Detection Rate (TPR) (%)	Avg. False Alarm Rate (FPR) (%)	Average Accuracy	F-Measure (%)	MAE	RMSE
FP GROWTH	97.87	0.1404	94.44	96.17	0.0556	0.2357
CMR	96.34	0.0108	97.71	97.53	0.0058	0.0764
AAR	96.08	0.0490	95.80	97.03	0.0420	0.2050
AAR-aPSO	*98.25*	*0.0192*	*98.17*	*98.25*	-	*0.0355*

Table 6: Models Best Performance of Unsupervised learners (Appropriateness) in terms of Error and Accuracy.

Classification Algorithms	True Detection Rate (TPR)	False Alarm Rate (FPR)	Accuracy	F-Measure	MAE	RMSE
Bayesian Classifier	0.9489	0.2705	0.8411	0.8598	0.1695	0.3668
Naïve Bayes	0.9270	0.2771	0.8280	0.8500	0.1703	0.3748
PART	0.8883	0.1332	0.8791	0.9030	0.1297	0.3210
J48	0.8815	0.1053	0.8802	0.9075	0.1477	0.3233
Random Forest	0.9200	0.1409	0.8960	0.9164	0.1548	0.2662
Neural Network	0.9501	0.1528	0.9114	0.9332	-	0.3983
Classification-based Multiple Association Rule	0.7884	0.8862	0.6670	0.7974	-	0.5747

Table 7: Permission Structure of numeric data.

	android.permission. ACCESS_FINE_LOCATION	android.permission.INTERNET	android.permission. WRITE_CONTACTS	...	Class
Application1	2	3	4	...	0
Application2		3		...	1
Application3	2		4	...	1

TN: Benign android application that was classified as Benign; i.e., *TNR* is the proportion of negative instances classified correctly.

FP: Nonmalicious android application that was classified as malware; i.e., *FPR* is the proportion of negative instances classified wrongly as positive (malware).

FN: Malicious android application that was classified as Benign; i.e., *FNR* is the proportion of positive instances wrongly classified as negative (nonmalicious android application)

Therefore,

$$\begin{aligned} TPR &= \frac{TP}{TP + FN} \\ TNR &= \frac{TN}{TN + FP} \\ FPR &= \frac{FP}{FP + TN} \\ FNR &= \frac{TN}{TN + FN} \end{aligned} \tag{11}$$

The accuracy actually measures the proportion of correctly classified instances (features):

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{12}$$

5.4. Experimental Settings and Implementation. The basis of this research experiment was based on research questions defined in Section 5.2, upon which statistical tests were carried out. The essence of the experiment is to examine the effectiveness of using an apriori association rule with Particle Swarm Optimization over other contemporary models. To this end, features were extracted and selected using apriori algorithm and other selection techniques and used to train classification algorithms for supervised learning experiment. In an unsupervised learning, apriori association rule, CMR, and FP-Growth are used to generate association rules for signature detection. A detection algorithm was built and used in conjunction with an existing and improved models for malware detection.

Accuracy (ACC), false positive rate (FPR), true positive rate, and error rate were measured and used to determine the effectiveness of the models. Table 5 shows the results of supervised learning experiment for seven different classification algorithms on the features of android applications. Table 6 shows the performance of unsupervised learners using new improved association rule model with particle swarm optimization and contemporary rule models. The permission structure of numeric data is shown in Table 7.

The time and memory complexity are another important attributes to measure the quality of algorithm. Table 8 shows the efficiency of new method AAR-aPSO with lower time. Table 9 shows the stratified sampling used for data model.

6. Experimental Results and Discussion

In order to compare the effectiveness of an improved AAR-aPSO with other existing models, rules were tested with

Table 8: Selection Algorithms' Memory and Execution time efficiency with 20% support and 80% confidence.

	Features Quantity											
	100	200	300	400	700	1000	100	200	300	400	700	1000
Selection Algorithms	Time taken (seconds)						Memory (MB)					
AA	8.2	15.5	20.8	25.1	37.0	54.0	8.4	9.0	9.8	10.2	49.2	52.5
PSO	7.1	14.2	18.5	22.2	28.5	45.0	11.0	13.2	15.8	20.0	40.9	41.2
FP GROWTH	6.9	13.7	17.1	21.2	25.8	35.3	8.7	9.9	13.5	17.3	27.1	40.9
AA-aPSO	6.5	12.4	16.5	20.8	24.5	31.0	6.6	8.8	12.4	15.4	22.0	40.2

Table 9: Stratified Sampling for Data Model (Population).

	Total	Training	Testing	Training & Test%
All	1420	70%	30%	
Benign	470	329	141	33.09859
Malicious	950	665	285	66.90141
Aggregate		994	426	100
segment Selection		94	95	

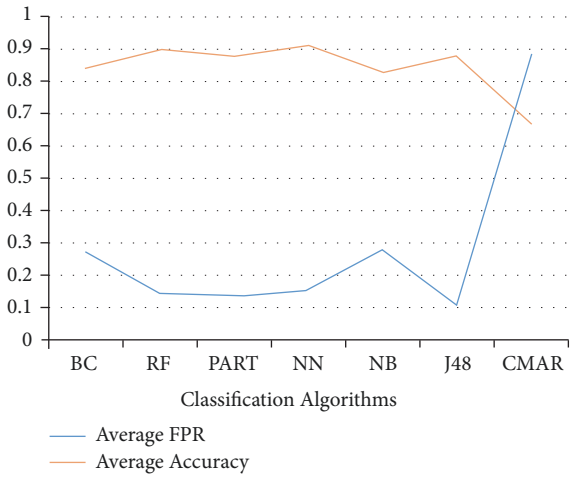

Figure 5: Average accuracy, detection, and error rates of supervised learners.

detection algorithm in Figure 4 and mean value of obtained results is computed to examine the distribution of the quality of the experimented models in terms of accuracy and detection rate. Table 5 shows the average accuracy, detection, and error rates of supervised learning experiment, while Table 6 and Figure 5 present the average statistics of AAR, CMR, FP-Growth, and AAR-aPSO models. Classification accuracy, detection rates, and error rates are also used to measure the effectiveness of supervised algorithms. The sum features reduced after selection from 1420 to 155 with 115 malicious and 40 benign.

The results of supervised experiment show that neural network has the best results in terms of accuracy of 91.14% with the detection rate of 95.01%. Unfortunately, neural network cannot explain what it learned coupled with the high false alarm rate and root mean square error. The best model could be random forest algorithm with 89.6% average accuracy, 92% average detection rate, and least 0.2662 root mean square error. Bayesian classifier also has better accuracy of 94.89%, but with high false alarm and error rates.

In unsupervised learning experiment, new model AAR-aPSO has the best average accuracy of 98.17%, average detection rate of 98.25%, false alarm rate of 0.0192, and 0.0355 root mean square error. This new model result is also better in terms of accuracy, detection, and error rates than classification algorithms' results. The memory and time complexity of new model also show better efficiency for computation. These statistical results show that an improved AAR-aPSO unsupervised model is better in terms of accuracy, false alarm, time and memory efficiency than supervised and other rule models and thus the null hypotheses in Section 5.3 I and II are rejected and the alternate hypotheses are accepted. Figure 6 shows the permission structure of binary vector feature used to train particle swarm optimization and an improved model, while Table 7 is a table of numeric data structure used to train apriori algorithm and other rule models. In Figure 6, the binary data 1 represents the presence of a feature in an android application, while 0 represents the absent of a particular feature in an application. In Table 7, the presence of a numerical number identifies a particular feature of android application, while the empty space dictates that such application has no particular feature. The essence of supervised experiment was to examine the best classification algorithm for the classification process, while the unsupervised learner was to extract signatures of malicious applications for malware detection.

7. Conclusion and Recommendation

The major contributions of this research are (1) the generation of best candidate detectors using apriori algorithm with particle swarm optimization and (2) the detection of android malicious applications using apriori association rules. The research demonstrates that apriori algorithm and apriori association rule as selection and classification technique,

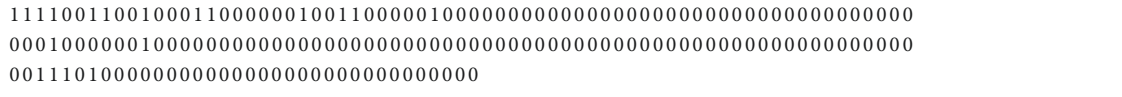

Figure 6: Permission structure of binary feature vector.

respectively, perform better in combination with particle swarm optimization. The research uses an improved system for the selection of best android features for the detection of android malicious applications. The research proposed an improved association rule based malware detection model using particle swarm optimization and features of mobile android application. The research conducted both supervised and unsupervised experiment using classification algorithms with android application features. The supervised learners were trained using 70% training data and 30% test data. The classification results measure the accuracy and detection rates of the models. In unsupervised experiment, association rules were generated using new model and contemporary models and were used with detection algorithm to test the accuracy and detection rates of the models. The final results show that new model, an improved apriori association rule with particle swarm optimization, has better performance in terms of accuracy, true detection, and error rates than other models both supervised and unsupervised. The performance of a model depends on McNemar's Test conducted on the models. The tests show that using Degree of freedom (DF) = (2-1) (2-1) = 1 with $\propto = 0.05, Chi^2 = 3.841$, for all McNemar's Test and previous calculated values are greater than distribution $\chi 2$ values; therefore, there is a significant difference between the compared models and, therefore, AAR-PSO outperforms other models. The novelty in this work is the inclusion of improved detection algorithm with PSO using association rule for signature extraction, compared to the existing one in [25], which was based only on classification exercise using an improved apriori algorithm with particle swarm optimization for selection and data mining algorithms for classification.

Data Availability

The data is available upon request from the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors wish to acknowledge the support of Federal Government of Nigeria under TETFUND Scholarship and International Islamic University Malaysia under the Endowment Grant research number EDW B14-123-1008. Authors also wish to acknowledge this work is a continuation of research in [25] with hyperlink https://ieeexplore.ieee.org/document/7077314.

References

[1] R. J. Roiger and M. W. Geatz, *Data Mining: A Tutorial-Based Primer*, Pearson Education Inc., 2003.

[2] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proceedings of the 20th International Conference on Very Large Data Bases, VLDB*, vol. 1215, pp. 487–499, September 1994.

[3] J. Gu, B. Wang, F. Zhang, W. Wang, and M. Gao, "An improved apriori algorithm," in *Applied Informatics and Communication*, pp. 127–133, Springer, Heidelberg, Berlin, 2011.

[4] O. S Adebayo, M. A. Mabayoje, A. Mishra, and O. Osho, "Malware detection, supportive software agents and its classification schemes," *International Journal of Network Security & Its Applications (IJNSA)*, vol. 4, no. 6, pp. 33–49, 2012.

[5] M. A. Siddiqui, *Data Mining Methods for Malware Detection [phd thesis]*, College of Sciences at the University of Central Florida, Orlando, Fla, USA, 2008.

[6] T. Wei, C. Mao, A. B. Jeng, H. Lee, H. Wang, and D. Wu, "Android malware detection via a latent network behavior analysis," in *Proceedings of the 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pp. 1251–1258, Liverpool, UK, June 2012.

[7] M. Christodorescu, S. Jha, S. A. Seshia, D. Song, and R. E. Bryant, "Semantics-aware malware detection," in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 32–46, May 2005.

[8] A. Shabtai, U. Kanonov, Y. Elovici, C. Glezer, and Y. Weiss, ""Andromaly": a behavioral malware detection framework for android devices," *Journal of Intelligent Information Systems*, vol. 38, no. 1, pp. 161–190, 2012.

[9] Y. Ye, D. Wang, T. Li, D. Ye, and Q. Jiang, "An intelligent PE-malware detection system based on association mining," *Journal of Computer Virology and Hacking Techniques*, vol. 4, no. 4, pp. 323–334, 2008.

[10] M. Ohrui, H. Kikuchi, M. Terada, and N. R. Rosyid, "Apriori-prefixspan hybrid approach for automated detection of botnet coordinated attacks, network-based information systems (NBiS)," in *Proceedings of the 14th International Conference on Network-Based Information Systems*, IEEE, 2011.

[11] Z. Yi and Z. Li-Jun, "A rule generation model using s-pso for misuse intrusion detection," in *Proceedings of the International Conference on Computer Application and System Modeling (ICCASM 2010)*, pp. V3-418–V3-423, Taiyuan, China, October 2010.

[12] M. Sheikhan and M. Sharifi Rad, "Gravitational search algorithm–optimized neural misuse detector with selected features by fuzzy grids–based association rules mining," *Neural Computing and Applications*, vol. 23, no. 7-8, pp. 2451–2463, 2013.

[13] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proceedings of the 6th International Symposium on Micro Machine and Human Science (MHS '95)*, vol. 1, pp. 39–43, Nagoya, Japan, October 1995.

[14] I. Idris and A. Selamat, "Improved email spam detection model with negative selection algorithm and particle swarm optimization," *Applied Soft Computing*, vol. 22, pp. 11–27, 2014.

[15] Y. Tan, "Particle swarm optimization algorithms inspired by immunity-clonal mechanism and their applications to spam detection," *International Journal of Swarm Intelligence Research*, vol. 1, no. 1, pp. 64–86, 2010.

[16] W. Wang, P. Zhang, Y. Tan, and X. He, "An immune local concentration based virus detection approach," *Journal of Zhejiang University Science*, pp. 443–454, 2011.

[17] X. Wang, J. Yang, X. Teng, W. Xia, and R. Jensen, "Feature selection based on rough sets and particle swarm optimization," *Pattern Recognition Letters*, vol. 28, no. 4, pp. 459–471, 2007.

[18] S.-W. Lin, K.-C. Ying, S.-C. Chen, and Z.-J. Lee, "Particle swarm optimization for parameter determination and feature selection of support vector machines," *Expert Systems with Applications*, vol. 35, no. 4, pp. 1817–1824, 2008.

[19] C. Bae, W. Yeh, Y. Y. Chung, and S. Liu, "Feature selection with intelligent dynamic swarm and rough set," *Expert Systems with Applications*, vol. 37, no. 10, pp. 7026–7032, 2010.

[20] H. Wang, X. Z. Gao, X. Huang, and Z. Song, "PSO-optimized negative selection algorithm for anomaly detectio," in *Applications of Soft Computing*, pp. 13–21, Springer, Heidelberg, Berlin, 2009.

[21] H. Wang, X. Z. Gao, X. Huang, and Z. Song, "PSO-optimized negative selection algorithm for anomaly detection," in *Applications of Soft Computing*, pp. 13–21, Springer, Heidelberg, Berlin, 2009.

[22] X. Hu and R. Eberhart, "Multi-objective optimization using dynamic neighborhood particle swarm optimization," in *Proceedings of the World on Congress on Computational Intelligence*, vol. 2, pp. 1677–1681, IEEE, 2002.

[23] R. Siddiqui and S. Khatibi, "Visual tracking using particle swarm optimization," 2014, https://arxiv.org/abs/1401.4648.

[24] M. S. Abadeh, J. Habibi, and S. Aliari, "Using a particle swarm optimization approach for evolutionary fuzzy rule learning: a case study," in *Information Processing and Management of Uncertainty in Knowledge Based Systems (IPMU)*, pp. 2–7, 2006.

[25] O. S. Adebayo and N. AbdulAziz, "Android malware classification using static code analysis and Apriori algorithm improved with particle swarm optimization," in *Proceedings of the 2014 Fourth World Congress on Information and Communication Technologies (WICT)*, pp. 123–128, Melaka, Malaysia, December 2014.

[26] B. Abhijit, H. Xin, G. S. Kang, and P. Taejoon, "Behavioral detection of malware on mobile handsets," ACM, Breckenridge, Colo, USA, 2008.

[27] J. J. Drake, P. O. Fora, Z. Lanier, C. Mulliner, S. A. Ridley, and G. Wicherski, *Android Hacker's Handbook*, John Willey & Sons, Incorporation, Indianapolis, Ind, USA, 2014.

[28] T. Bläsing, L. Batyuk, A.-D. Schmidt, S. A. Camtepe, and S. Albayrak, "An android application sandbox system for suspicious software detection," in *Proceedings of the 5th International Conference on Malicious and Unwanted Software (Malware '10)*, pp. 55–62, Nancy, France, October 2010.

[29] C. Warrender, S. Forrest, and B. Pearlmutter, "Detecting intrusions using system calls: Alternative data models," in *Proceedings of the 1999 IEEE Symposium on Security and Privacy*, pp. 133–145, USA, May 1999.

[30] T. Eder, M. Rodler, D. Vymazal, and M. Zeilinger, "A framework for analyzing android applications," in *Proceedings of the Workshop on Emerging Cyber threats and Countermeasures ECTCM*, 2013.

[31] G. Dini, F. Martinelli, A. Saracino, and D. Sgandurra, "MADAM: a multi-level anomaly detector for android malware," in *Proceedings of the 6th international conference on Mathematical Methods, Models and Architectures for Computer Network Security: computer security (MMM-ACNS'12)*, pp. 240–253, 2012.

[32] A. Walenstein, L. Deshotels, and A. Lakhotia, "Program structure-based feature selection for android malware analysis," in *Proceedings of the 4th International Conference, MobiSec 2012*, vol. 107 of *LNICST*, pp. 51-52, Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, 2012.

[33] S. Holla and M. M. Katti, "Android based Mobile Application and Its Security," *International Journal of Computer Trends and Technology*, vol. 3, no. 3, pp. 486–490, 2013.

[34] I. Burguera, U. Zurutuza, and S. Nadjm-Tehrani, "Crowdroid: behavior-based malware detection system for android," in *Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices*, pp. 15–25, October 2011.

[35] K. J. L. Abela, D. K. E. Angeles, D. Alas, R. P. Jan, J. R. Tolentino, and M. A. N. Gomez, "An automated malware detection system for android using behavior-based analysis, AMDA," *International Journal of Cyber-Security and Digital Forensics*, vol. 2, no. 2, pp. 1–11, 2013.

[36] Y. Aafer, W. Du, and H. Yin, "DroidAPIMiner: mining API-level features for robust malware detection in android," in *Security and Privacy in Communication Networks*, pp. 86–103, Springer International Publishing, 2013.

[37] M. Kantardzic, *Data Mining: Concepts, Models, Methods, and Algorithms*, IEEE Press, USA, 2011.

[38] S. S. Garasia, D. P. Rana, and R. G. Mehta, "HTTP botnet detection using frequent patternset mining," *International Journal of Engineering Science & Advanced Technology*, vol. 2, pp. 619–624, 2012.

[39] M. G. Schultz, E. Eskin, E. Zadok, and S. J. Stolfo, "Data mining methods for detection of new malicious executables," in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 38–49, May 2001.

[40] J. Z. Kolter and M. A. Maloof, "Learning to detect and classify malicious executables in the wild," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 470–478, 2006.

[41] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.

[42] Y. Shi and R. C. Eberhart, "Fuzzy adaptive particle swarm optimization," in *Proceedings of the Congress on Evolutionary Computation*, vol. 1, pp. 101–106, IEEE, Seoul, Republic of Korea, May 2001.

[43] Y. H. Shi and R. C. Eberhart, "A modified particle swarm optimizer," in *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC '98)*, pp. 69–73, Anchorage, Alaska, USA, May 1998.

[44] Y. Tan and Z. M. Xiao, "Clonal particle swarm optimization and its applications," in *Proceedings of the 2007 IEEE Congress on Evolutionary Computation, CEC 2007*, pp. 2303–2309, Singapore, September 2007.

[45] Contagio Mobile, http://www.contagiominidump.com.

[46] Google play, 2013, https://play.google.com/store.

[47] O. S. Adebayo and N. A. Aziz, "Techniques for analysing android malware," in *Proceedings of the 5th International Conference on Information and Communication Technology for the Muslim World, ICT4M 2014*, Kuching, Malaysia, November 2014.

[48] V. J. Varghese and S. Walker, *Dissecting Andro Malware*, SAN Institute, School of Computer and Electronic Engineering, University of Essex, Colchester, UK, 2011.