International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878, Volume-8 Issue-5, January 2020

EKMPRFG: Ensemble of KNN, Multilayer Perceptron and Random Forest using Grading for Android Malware Classification

Niranjan A., K. S. S. S. Abhishikth, P. Deepa Shenoy, Venugopal K. R.

Abstract: ***Android is the most popular Operating Systems with over 2.5 billion devices across the globe. The popularity of this OS has unfortunately made the devices and the services they enable, vulnerable to numerous security threats. As a result of this, a significant research is being done in the field of Android Malware Detection employing Machine Learning Algorithms. Our current work emphasizes on the possible use of Machine Learning techniques for the detection of malware on such android devices. The proposed EKMPRFG is applied for the classification of Android Malware after a preprocessing phase involving a hybrid Feature Selection model using proposed Standard Deviation of Standard Deviation of Ranks (SDSDR) and several other builtin Feature Selection algorithms such as Correlation based Feature Selection (CFS), Classifier SubsetEval, Consistency SubsetEval, and Filtered SubsetEval followed by Principal Component Analysis(PCA) for dimensionality reduction. The experimental results obtained on two data sets indicate that EKMPRFG outperforms the existing works in terms of Prediction Accuracy and Weighted F- Measure values.***

Keywords: Ensemble Learning, Hybrid Feature Selection, Malware Classification, Malware Analysis

I. INTRODUCTION

As per the recent statistics, market share of Android is over 85% of the total Mobile OS market. There are over 2.5 billion monthly active users of the Android OS. This popularity has made Android vulnerable to increased number of malware attacks [1]. Some of the Malware families that have grown exponentially over a very short period of time are, TimpDoor, DressCode, MilkyDoor, Guerrilla, and Rootnik. TimpDoor malware family the most recent and the deadliest of all, directly contacts potential victims via SMS and lures them to install the infected application thus avoiding the threat of being removed from the store.

Manuscript published on January 30, 2020.
* Correspondence Author
Niranjan A*., Research Scholar, Department of Computer Science and Engineering, University Visvesvaraya College of Engineering, Bangalore University, Bangalore, India
K. S. S. S. Abhishikth, Department of Computer Science, University of South Florida, USA.
P. Deepa Shenoy, Professor, Department of Computer Science and Engineering, University Visvesvaraya College of Engineering, Bangalore University, Bangalore, India.
Venugopal K. R., Vice Chancellor, Bangalore University, Bangalore, India.

Malware Detection on its very onset is the only effective solution to deal with this problem. It can happen only after an accurate classification of a sample as malware or benign. Machine Learning techniques prove to be effective for all classification problems.

Hybrid models involving Multi-layered classification have not been explored on this problem as much as they are used in other applications.

In this work, we propose and investigate a Hybrid model involving Multi-layered classification for effective prediction of Malware samples. We explore various Ensembling Schemes such as StackingC, Voting and Grading for implementing the Hybrid model and based on the results, the most efficient model is proposed. Just before classification, we explore various Rank Based Feature Selection algorithms to determine the ranks of all the features present in the data set. Based on the rank values given by various algorithms for a Feature, a Standard Deviation is determined for the Feature. This is repeatedly computed for every feature present in the data set. Finally, a Standard Deviation is computed for the Standard Deviation values of all features. All such features whose Standard Deviation values are lesser than the final standard Deviation value are discarded and only a Feature subset that is relevant and significant for Classification is chosen. We call this technique as SDSDR. To validate our proposed SDSDR Feature Selection algorithm, we further run various non-rank-based Feature Selection algorithms with different Search techniques and determine those features that are picked up at least five times by various non-rank-based Feature Selection algorithms. We call this technique as FSF (Feature Selection using Frequency). This step ensures that only most significant Feature subset is chosen for the further process and helps in reducing the dimensionality of the data set there by improving the speed. We perform a union operation on the subsets generated by SDSDR with FSF to actually ensure no significant feature is left out. To further reduce dimensionality of the final Feature subset, PCA (Principal Component Analysis) is performed as the final step of Pre-Processing Phase.

The current work emphasizes on significant feature subset selection and dimensionality reduction without degradation in Prediction accuracy. This is achieved by selecting classifier algorithms with better individual performances experimenting with various combinations of them. The top performing classifiers in terms of performance on MALGENOME and DREBIN data sets are determined and are then combined using various ensembling approaches such as StackingC, Grading and Voting.

It was observed through our experiments that a hybrid model involving kNN, Multilayer Perceptron, and Random Forest when ensembled through Grading offers better results in terms of the chosen metrics such as Precision, Recall of benign and malware samples, and Weighted F-Measure. We call this approach as EKMPRFG.

For demonstrating the efficiency of the proposed EKMPRFG approach, extensive experiments were conducted on two data sets from publicly available malware samples collections namely MALGENOME-215 [2] and DREBIN-215 [3] consisting of 3799 and 15036 instances respectively. In the final phase of our model, the feature subset obtained after PCA is subjected to various Machine Learning Classifiers and different performance metrics such as Precision, Recall of benign and malware samples, and Weighted F-Measure are recorded. The top performing classifiers in terms of Precision, Recall, and Weighted F-Measure values on both the data sets is determined thereafter. The top classifiers are then combined using various ensembling approaches such as StackingC, Grading and Voting. The experimental results obtained on two data sets indicate that EKMPRFG outperforms the existing DROIDFUSION [4] in terms of Prediction Accuracy and Weighted F-Measure.

kNN is a classifier that learns on the individual instances. As the computation is deferred until actual classification, kNN is also termed as a Lazy learning algorithm. The kNN algorithm presumes that similar things exist in proximity and hence the predictions of all the closely placed neighbors are considered for making the final prediction of a test set sample.

A perceptron classifies an input with a straight line by separating the two categories. A Product of feature vector f and weights wt that is added to a bias bi act as the input y = wt * f + bi. A Multilayer Perceptron (MLP) is a deep, artificial neural network. An MLP is composed of multiple number of hidden layers in between an input layer that receives various inputs, and an output layer for making a prediction about the input.

As Multilayer perceptrons train on a set of input-output pairs and learn to model the correlation between them, they are suitable for supervised learning problems. For minimizing the errors, it is required to adjust the parameters of a feature vector or the weights and biases of the model during the Training phase.

An MLP is typically like a game of guesses and responses: with every guess we would be testing what we know, and every response is like a feedback of how wrong we are. It typically involves two phases namely: 1. forward pass & ii. backward pass. In the forward Pass, similar to a Supervised Learner, the prediction made by the output obtained from the output layer is measured with respect to the ground truth labels but in the backward pass, partial derivatives if any of the error function regarding weights and biases are backpropagated through the MLP. This enables the input layer to determine the gradient of error and the parameters that can be adjusted to keep the errors at lowest possible values or at the convergence state.

The Random Forest is a classifier composed of several independent decision trees. Every individual tree is built by through the use of a popular Ensemble technique called Bagging and Feature Randomness so that an uncorrelated forest of trees is created, and the final prediction is made by a committee based on the maximum votes garnered for a particular prediction and such a prediction is much more accurate than that of any individual tree.

Some of the commonly used Ensemble techniques are Bagging, Boosting, Voting, Stacking, StackingC, and Grading. Voting involves the creation of a number of sub-models and involving each of them in the voting process of choosing on the outcome of a prediction. Stacking involves the individual and independent training of heterogeneous learning algorithms on the data and considering the outcomes of each of them as additional inputs to the combiner algorithm for the final training. StackingC an improvised version of Stacking, makes use of Linear Regression as the Meta Classifier. Linear Regression is a process of merging a set of numeric values (x) into a predicted output value (y). Grading is one of the meta classification techniques that involves the process of identifying and correcting incorrect predictions if any. Unlike Stacking that uses the predictions of the base classifiers as metalevel attributes, Grading makes use of graded predictions (correct or incorrect) as meta-level classes.

The contributions of this article can be summarized as below:

- A novel general-purpose classifier framework involving Hybrid approach (EKMPRFG) has been presented along with its evaluation on two data sets.
- We also propose two Feature Selection algorithms SDSDR and FSF for the efficient selection of the Features without compromising on the Performance of Classification.
- The results of the extensive experiments that are conducted on individual classifiers and ensemble classifiers such as StackingC, Voting and Grading are presented to demonstrate the effectiveness of our proposed approach.
- We also present results of a performance comparison of EKMPRFG with Droid Fusion. The remainder of the article is organized as follows. Related work with respect to this field is discussed in Section II while Section III presents the proposed **EKMPRFG** framework. Section IV elaborates the investigation methodology, while section V presents results, with analyses and discussions. Conclusion forms the final Section of this paper.

II. RELATED WORK

DROIDFUSION [4] uses classifier fusion approach based on a multilevel architecture incorporating a combination of various machine learning algorithms. Their model typically involves, training of base classifiers for the estimation of their relative predictive accuracies by making use of a stratified N-fold cross validation technique. In the next level, four different ranking-based algorithms make use of the outcomes obtained from the previous level and select combination of a subset of the applicable base classifiers. The outcomes of these ranking algorithms are combined in pairs and only the strongest pair is chosen to build their model. Their approach outperforms the Stacked Generalization model. The authors of [5], emphasize the importance of meta data available on the descriptive page of the APK file available in market and to include some of this information as additional features into the data set.

After preparing the data set with all the included meta data as additional features, they classify the samples using the Linear SVM technique. They achieve detection accuracy of about 94%. The methodology presented in [6] illustrates a dynamic behavior inspection and analysis framework for malicious behavior detection in Android apps. The authors of the work customize the Android System to record run-time features such as API calls, Uses of Permission etc., To collect the run-time behavior data for classification training they develop an automated testing platform and load apps of about 13825 including both malicious and benign apps. They then analyze a total of 3917 apps available on the play store including 952 malware apps from the Anruan market. Through their experiments they further prove that SVM classifier achieves 99.0% Detection Accuracy with a 1.0% FPR and a 2.3% FNR.

The authors in their work [7], present a Deep Learning Model that is based on the DBN (Deep Belief Network) algorithm for characterization and identification of Android Malware. As a part of the Pre-Processing phase, they include two additional features namely Permissions and API calls into the Feature set by checking the manifest (.xml) and source (.java) files of various apps respectively. Finally, the data set with the above two features included is provided as input to the DBN algorithm for the classification of the test samples. Their results demonstrate that their approach performs better than various versions of SVM in terms of Precision, Recall, and F1-score.

The work presented in [8] illustrates the possible use of calls of dataflow-related APIs as the additional feature in the data set for efficient classification of android malware samples using an improved version of distance calculation for the neighbors in kNN. They argue that through their approach it is possible to reduce computational overheads to a considerable extent.

III. EKMPRFG: A FRAMEWORK FOR EFFICIENT ANDROID MALWARE DETECTION

Majority of the applications that involve Machine Learning techniques would revolve around Preprocessing and Classification phases. Selection of significant features from the data set becomes an important step of preprocessing, as all features in the data set are not relevant during the final prediction. We propose SDSDR and FSF algorithms for the selection of Features by exploiting the numerous advantages of existing Weight Based Ranking Algorithms and Non-Weighted Feature Selection Algorithms. The Preprocessing phase also involves the application of PCA (Principal Component Analysis) along with SDSDR and FSF for dimensionality reduction. The classification phase involves, subjecting the resultant Feature subset to the proposed **EKMPRFG** framework by means of ten-fold cross validation involving kNN, Multilayer Perceptron, and Random Forest that are ensembled through Grading and various performance metrics such as Precision, Recall of benign and malware samples, and Weighted F-Measure are recorded. The **EKMPRFG** framework is depicted in Fig.1. The two data sets that are used for the experimentation purpose are MALGENOME and DREBIN. The MALGENOME-215 has 215 Features having a total of 3799 samples with 2,539 benign and 1,260 malware samples and DREBIN-215 also has 215 Features and a total of 15036 instances with 9476 benign and the remaining 5,560 malware samples.

The proposed SDSDR (Standard Deviation of Standard Deviation values of Ranks) Feature Selection algorithm as listed in Algorithm 1 employs several Rank Based Feature Selection Algorithms for determining the ranks of all the features that are present in the data set. Based on the rank values given by various Algorithms for a Feature, a Standard Deviation is determined for the Feature. This is repeatedly computed for every feature present in the data set. Finally, a Standard Deviation is computed for the Standard Deviation values of all features. All such features whose Standard Deviation values are lesser than the final standard Deviation value are discarded and only a Feature subset that is relevant and significant for Classification is chosen. Information Gain, Gain Ratio Attribute Eval, Correlation Attribute Eval, Symmetrical Uncert AttributeEval, Probabilistic Significance AE, and ClassPART were employed to compute the Ranks R_i in the proposed SDSDR. The SDSDR that we obtained for the DREBIN Data set was 0.043991045 and that of MALGENOME was 0.047264133. Features having lesser Weights than the SDSDR values were ignored. The proposed SDSDR returned a subset of 16 features in case of DREBIN and 20 features in case of MALGENOME out of a total 215 features.

So as to determine the most common features of various Non-Weight Based Feature Selection algorithms and to further ensure that only significant Feature subset is selected for the classification phase, we propose another simple but useful algorithm called FSF (Feature Selection using Frequency). The FSF algorithm as listed in Algorithm 2 involves the determination of most significant features through the use of various Non-Weight Based Feature Selection algorithms with different Search Algorithms and to determine those features that appear in at least 5 different Non-Weight Based Feature Selection and various Search Algorithm combinations. The feature subset that is chosen in this manner by the FSF is compared with the feature subset that is chosen by the SDSDR and to ensure no significant feature is left out from the final feature subset that is chosen for the classification phase, we perform an Union operation of both the subsets. The various Non-Weight Based Feature Selection algorithms that were employed for experimentation are CFS Subset Eval, Consistency Subset Eval, Filtered Subset Eval, and Classifier_Subset_Eval. We employed Genetic, PSO, Rank, Reranking, and BFS search techniques for all of the above Non-Weight Based Feature Selection algorithms except for the Classifier Subset Eval in whose case we used only Genetic, and PSO search techniques. The FSF algorithm on MALGENOME returned a subset of 80 features while the DREBIN returned a subset of 101 features with a minimum frequency of 5. It was observed that the 101 features chosen by the FSF on DREBIN after a Union Operation with feature subset obtained from SDSDR on DREBIN (16 features) do not yield better classification results and hence we considered only the 80 features chosen by the FSF on MALGENOME and after the union operation with the 20 features subset obtained from SDSDR on MALGENOME, we got a feature set having 81 features and a class label. This amounts to a total reduction of features by 62.33% without any noticeable reduction in the performance.

As a final step of the Preprocessing Phase, we propose to perform PCA for the further reduction in dimensionality. It reduces the number of features to 59 and a class label.

Algorithm 1: SDSDR Feature Selection (Standard Deviation of Standard Deviation of Ranks)

Input: Data set *D* having *n* number of Features
Output: $F \subseteq D$ with Significant Features

1. for every feature $f_i \in D$ do
   Ranks R_i are determined through different Weight Based Feature Selection Techniques
   next
2. for every feature $f_i \in D$ do
   Sum and Mean of Ranks ($\sum R_i$) and (mR_i) are computed.
   $\sum R_i = R_1 + R_2 + \ldots + R_n$ and $mR_i = \sum R_i / n$
   next
3. for every feature $f_i \in D$ do
   Determine Standard Deviation σ_r for the feature
   next
4. Determine Standard Deviation of Standard deviation of Ranks $\sigma_f (\sigma_{r1}, \sigma_{r2}, \ldots \sigma_{rn})$
5. Discard all $f_i \in D < \sigma_f$
6. **return** *F*

Algorithm 2: FSF (Feature Selection using Frequency)

Input: *D* data set having *n* number of Features
Output: $F \subseteq D$ with Most Significant Features

1. Perform the intersection ($\cap$) of the features selected by the various Non-Weight Based Feature Selection Algorithms and their Search technique combinations to determine the Common Features.
2. Determine the Frequency of the Common Features.
3. Select only those Features whose Frequency is >=5
4. Perform the Union ($\cup$) of the features obtained in Step 3 and Features obtained from SDSDR.
5. **return *F***

Algorithm 3: Ensemble of kNN, Multilayer Perceptron, and Random Forest using Voting **(EKMPRFV)**

Input: $F \subseteq D$ obtained after applying **PCA**

Output: Performance Metrics

1. Provide input to the proposed Ensemble of kNN, Multilayer Perceptron, and Random Forest using Voting.
2. Apply ten-fold cross validation and the performance metrics are recorded.

The resulting feature subsets are finally subjected to the proposed EKMPRFV framework. The proposed EKMPRFV algorithm for efficient Android Malware Classification is presented in Algorithm 3. Table 1 lists the selected feature subsets of the SDSDR on DREBIN-215 and MALGENOME-215 respectively. The proposed SDSDR and FSF for Feature Selection are presented as Algorithm1 and Algorithm2 . Table 2 lists the 81(80 selected by the FSF on MALGENOME-215 plus the feature added after performing Union operation with SDSDR) features. It was observed that the 101 features chosen by the FSF on DREBIN-215 after a Union Operation with feature subset obtained from SDSDR on DREBIN (16 features) do not yield better classification results and hence we considered only the 80 features chosen by the FSF on MALGENOME-215 and after the union operation with the 20 features subset obtained from SDSDR on MALGENOME-215, we got a feature set having 81 features and a class label.

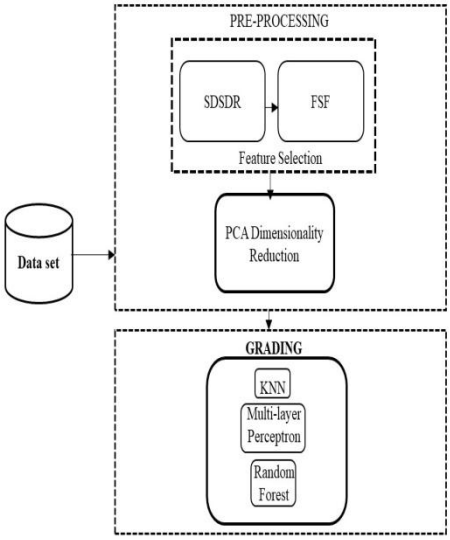

FIG. 1. Proposed framework for Android Malware Detection

The experimental results as indicated in Table 5 suggest that the hybrid ensemble using Grading Framework involving kNN, Multilayer Perceptron and Random Forest classifiers offer best Precision, Recall, and F-1 measure values of various combinations.

Table 1: Features Selected by SDSDR

DREBIN-215	MALGENOME-215
transact	transact
onServiceConnected	android.telephony .gsm.SmsManager
bindService	RESTART_PACKAGES
attachInterface	abortBroadcast
ServiceConnection	onServiceConnected
android.os.Binder	READ_SMS
SEND_SMS	TelephonyManager. getSubscriberId
Ljava.net.URLDecoder	SEND_SMS
android.telephony .SmsManager	chmod
ClassLoader	WRITE_APN_SETTINGS
Landroid.content. Context.registerReceiver	TelephonyManager .getLine1Number
RECEIVE_SMS	ServiceConnection
READ_SMS	createSubprocess
android.content.pm .PackageInfo	android.intent action.BOOT_ COMPLETED
TelephonyManager .getLine1Number	attachInterface
WRITE_SMS	bindService

	WRITE_SMS
	android.os.Binder
	RECEIVE_SMS
	INSTALL_PACKAGES

Table 2: Features Selected by FSF on MALGENOME-215

1	transact
2	onServiceConnected
3	ServiceConnection
4	android.os.Binder
5	READ_SMS
6	attachInterface
7	WRITE_SMS
8	TelephonyManager.getSubscriberId
9	Ljava.lang.Class.getCanonicalName
10	Ljava.lang.Class.getField
11	READ_PHONE_STATE
12	GET_ACCOUNTS
13	getBinder
14	chmod
15	Ljava.net.URLDecoder
16	WRITE_APN_SETTINGS
17	TelephonyManager.getDeviceId
18	Ljava.lang.Class.getDeclaredField
19	abortBroadcast
20	TelephonyManager.getLine1Number
21	USE_CREDENTIALS
22	MANAGE_ACCOUNTS
23	android.telephony.gsm.SmsManager
24	.system.bin
25	RESTART_PACKAGES
26	INSTALL_PACKAGES
27	Ljava.lang.Class.forName
28	CAMERA
29	onBind
30	READ_SYNC_SETTINGS
31	INTERNET
32	android.telephony.SmsManager
33	WRITE_CONTACTS
34	.system.app
35	Ljava.lang.Class.getResource
36	android.intent.action.SEND_MULTIPLE
37	ACCESS_WIFI_STATE
38	URLClassLoader
39	WAKE_LOCK
40	READ_SYNC_STATS
41	BROADCAST_STICKY
42	android.intent.action.PACKAGE_REMOVED
43	WRITE_SECURE_SETTINGS
44	SUBSCRIBED_FEEDS_WRITE
45	PROCESS_OUTGOING_CALLS
46	MASTER_CLEAR
47	android.intent.action.BATTERY_LOW
48	WRITE_CALL_LOG
49	BIND_INPUT_METHOD
50	android.intent.action.SCREEN_ON
51	android.intent.action.SCREEN_OFF
52	INTERNAL_SYSTEM_WINDOW
53	INJECT_EVENTS
54	SET_PROCESS_LIMIT
55	INSTALL_LOCATION_PROVIDER
56	bindService
57	Ljava.lang.Class.getMethods
58	android.intent.action.BOOT_COMPLETED
59	Landroid.content.Context.registerReceiver
60	Ljava.lang.Class.cast
61	createSubprocess
62	ClassLoader
63	Ljava.lang.Class.getMethod
64	PackageInstaller
65	remount
66	TelephonyManager.getSimCountryIso
67	READ_HISTORY_BOOKMARKS
68	WRITE_HISTORY_BOOKMARKS
69	RECEIVE_WAP_PUSH
70	WRITE_CALENDAR
71	WRITE_USER_DICTIONARY
72	android.content.pm.PackageInfo
73	Landroid.content.Context.unregisterReceiver
74	SEND_SMS
75	android.intent.action.TIME_SET
76	ACCESS_MOCK_LOCATION
77	android.intent.action.SEND
78	android.intent.action.PACKAGE_ADDED
79	WRITE_GSERVICES
80	HttpGet.init
81	RECEIVE_SMS

IV. INVESTIGATION METHODOLOGY

DREBIN 215 and MALGENOME 215 were chosen to act as the benchmark data sets for all the experimentation. Both of them consist of a total of 215 features. The DREBIN 215 has 15036 instances while the MALGENOME-215 comprises of 3799 instances. The classifiers were subjected to a rigorous ten-fold cross validation before arriving at the performance metrics which requires dividing the data set into ten sections and the model would be trained with the nine sections of data and the excluded section would be as the test set and the process would be repeated for ten rounds and each unused test set would be used during each round. After applying the proposed SDSDR and FSF Feature Selection Algorithms on both the data sets we decided to use 81 features subset as listed in Table 2 for the further Dimensionality Reduction using PCA. The Principal Component Analysis (PCA) is suitable when the data set has a number of features that are correlated with one another. By retaining only, the variation present in the data set and ignoring all other features, PCA ensures dimensionality reduction. All the existing features of the data set are transformed to a new set of orthogonal features, called the Principal Components (PCs). The correlation between any pair of PCs is always a 0. Such a resultant set is finally subjected to a number of built-in Classifier Algorithms and the performance metrics are recorded after a rigorous ten-fold cross validation. Precision, Recall, and Weighted F-Measure values are used as the yard stick for the efficiency evaluation of the classifiers.

Different Ensemble approaches on the top performing classifiers are also tried to enhance the Performance of Classification.

Table 3 lists various classifier results on MALGENOME-215 while Table 4 lists the results on DREBIN-215. A classifier with higher true positive rates and lower false positive rates is considered to be efficient.

We define 8 Performance metrics of a classical classification methodology. N_{ben} is the total number of benign (normal) samples while N_{mal} is the number of malicious samples in the malware data set. True Positive (TP) is the number of benign samples classified accurately as benign and is denoted as $N_{ben\to ben}$

True Negative (TN) is the accurate number of malicious samples classified as malicious. It is denoted as $N_{mal\to mal}$. False Positive (FP) is a measure of benign samples misclassified as malicious. It is denoted as $N_{ben\to mal}$ and False Negative (FN) is a measure of malicious instances misclassified as benign. It is represented as $N_{mal\to ben}$.

Table 3: Classification Results of various Classifiers on MALGENOME-215

Classification Algorithm	PrecM	PrecB	RecM	RecB	WFM
kNN	0.987	0.995	0.99	0.993	0.992
LibSVM	0.991	0.99	0.979	0.996	0.99
Random Forest(RF)	0.998	0.985	0.969	0.999	0.989
LMT	0.98	0.991	0.983	0.99	0.988
FT	0.979	0.991	0.983	0.989	0.987
FURIA	0.984	0.985	0.97	0.992	0.985
MLP	0.97	0.992	0.984	0.985	0.985
SGD	0.972	0.989	0.977	0.986	0.983
SimpleLogistic	0.982	0.983	0.966	0.991	0.983
ForestPA	0.993	0.977	0.952	0.996	0.982
SMO	0.984	0.978	0.956	0.993	0.98
Spegasos	0.981	0.98	0.96	0.991	0.98
PART	0.969	0.984	0.968	0.985	0.979
J48	0.964	0.985	0.969	0.982	0.978
VotedPerceptron	0.962	0.985	0.969	0.981	0.977
SimpleCART	0.966	0.978	0.956	0.983	0.974
SPAARC	0.963	0.978	0.955	0.982	0.973
BFTree	0.962	0.977	0.954	0.981	0.972
J48Consolidated	0.948	0.984	0.967	0.974	0.972
REPTree	0.954	0.979	0.958	0.977	0.971

Table 4: Classification Results of various Classifiers on DREBIN-215

Classification Algorithm	PrecM	PrecB	RecM	RecB	WFM
RF	0.991	0.982	0.968	0.995	0.985
kNN	0.979	0.987	0.977	0.988	0.984
MLP	0.977	0.982	0.97	0.987	0.981
LMT	0.973	0.982	0.969	0.984	0.979
ForestPA	0.984	0.975	0.956	0.991	0.978
LibSVM	0.982	0.974	0.955	0.99	0.977
FURIA	0.981	0.973	0.952	0.989	0.976
FT	0.967	0.98	0.965	0.981	0.975
PART	0.969	0.977	0.961	0.982	0.974
J48	0.965	0.976	0.959	0.98	0.972
BFTree	0.967	0.973	0.954	0.981	0.971
SimpleCART	0.966	0.974	0.955	0.98	0.971
J48Consolidated	0.958	0.976	0.959	0.976	0.97
SPAARC	0.961	0.973	0.953	0.977	0.968
REPTree	0.955	0.97	0.948	0.974	0.965
Random Tree	0.955	0.969	0.948	0.974	0.964
SGD	0.95	0.965	0.94	0.971	0.96
VotedPerceptron	0.952	0.965	0.94	0.972	0.96
SimpleLogistic	0.953	0.963	0.937	0.973	0.959
SMO	0.956	0.955	0.922	0.975	0.955

The Detection Rate (DR) is the rate of malicious samples being classified accurately as malicious.

$$TP= \frac{N\ ben\to ben}{(Nben\to ben+Nanom\to ben)}\times 100 \quad (1)$$

False positive rate (FPR) is the rate of benign samples being classified inaccurately as malicious samples.

$$FPR =\frac{Nben\to mal}{(Nmal\to mal+Nben\to ben)}\times 100 \quad (2)$$

False Negative Rate (FNR) is the rate of malicious samples being classified incorrectly as benign samples.

$$FNR = \frac{Nanom\to ben}{(Nmal\to ben+Nmal\to mal)}\times 100 \quad (3)$$

True Negative Rate (TNR) is the rate of benign samples being classified accurately as benign out of the total available benign samples.

$$TNR = \frac{N\ ben\to ben}{(N\ ben\to ben+Nben\to mal)}\times 100 \quad (4)$$

Prediction Accuracy is the total number of malicious and benign samples that are identified accurately with respect to the total number of all available instances.

$$PA = \frac{(Nmal\to mal+N\ ben\to ben)}{(Nmal\to mal+N\ ben\to ben+Nben\to mal+Nmal\to ben)}\times 100 \quad (5)$$

Precision is the number of true positives divided by the total number of instances labeled as belonging to the positive class.

$$Precision = \frac{Nmal\to mal}{(Nmal\to mal+Nben\to mal)}\times 100 \quad (6)$$

Recall is the number of true positives divided by the total number of instances that really belong to the positive class.

$$Recall = \frac{Nmal\to mal}{(Nmal\to mal+Nmal\to ben)}\times 100 \quad (7)$$

Weighted F-Measure is the harmonic mean of Precision and Recall and is given by:

$$WFM = 2\frac{Precision*Recall}{Precision+Recall} \quad (8)$$

The experimental results as indicated in Table 3 and Table 4 suggest that kNN, Random forest, Multi-Layer Perceptron and SVM classifiers offer better Precision, Recall, and Weighted F-Measure values out of all classifier algorithms. Based on these findings, we decided to use an Ensemble approach involving top performing Classifiers on both Data sets. The Ensembling approaches that were experimented by us included Grading, Voting, and StackingC. Our findings are presented in Table 5 and are plotted in Fig. 2 and Fig. 3.

The points along the X-axis in both Fig. 2 and Fig. 3 indicate separate Precision, and Recall values recorded with respect to Malware and Benign Samples and Weighted F-Measure. It may be noticed from these figures that all ensemble schemes used by us perform better compared to Droid fusion[4]. Unfortunately, none of the Ensemble schemes behave uniformly on both the data sets. So, it was decided to use such Ensemble Scheme that behaves better on both the data sets. Based on the results, we chose the Ensemble Scheme involving kNN, MLP, and Random Forest using Grading technique as our Proposed Model.

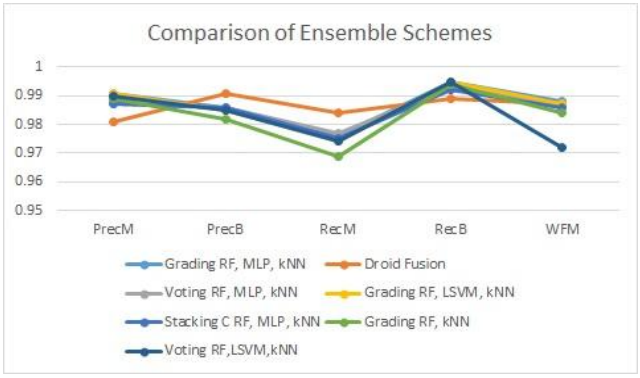

FIG. 2. Comparison of Ensemble Schemes on MALGENOME-215

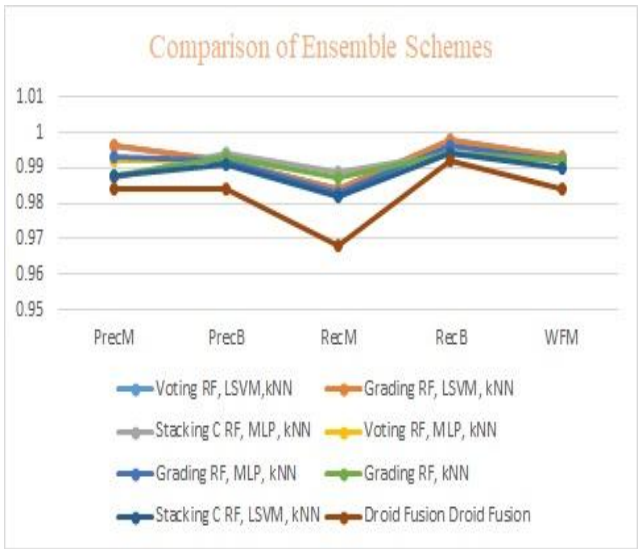

FIG. 3. Comparison of Ensemble Schemes on Drebin-215

Table 5:Comparison of various Ensemble Schemes

Data set	Ensemble Algorithm	Classification Algorithms used	PrecM	PrecB	RecM	RecB	WFM
DREBIN-215	Voting	RF, LSVM, kNN	0.996	0.992	0.984	0.998	0.993
	Grading	RF, LSVM, kNN	0.996	0.992	0.984	0.998	0.993
	Stacking C	RF, MLP, kNN	0.987	0.994	0.989	0.994	0.992
	Voting	RF, MLP, kNN	0.992	0.992	0.983	0.996	0.992
	Grading	RF, MLP, kNN	0.993	0.992	0.983	0.996	0.992
	Grading	RF, kNN	0.988	0.993	0.987	0.994	0.992
	StackingC	RF,LSVM, kNN	0.985	0.985	0.975	0.991	0.985
	Droid Fusion	Droid Fusion	0.984	0.984	0.968	0.992	0.984
MALGENOME-215	Voting	RF, LSVM, kNN	0.99	0.985	0.974	0.995	0.972
	Grading	RF, LSVM, kNN	0.991	0.985	0.974	0.995	0.987
	Stacking C	RF, MLP, kNN	0.987	0.986	0.975	0.992	0.986
	Voting	RF, MLP, kNN	0.988	0.986	0.977	0.993	0.987
	Grading	RF, MLP, kNN	0.991	0.986	0.976	0.995	0.988
	Grading	RF, kNN	0.989	0.982	0.969	0.994	0.984
	StackingC	RF, LSVM, kNN	0.988	0.991	0.982	0.994	0.99
	Droid Fusion	Droid Fusion	0.981	0.991	0.984	0.989	0.9872

V. CONCLUSION

Feature Selection using SDSDR and FSF are applied to select significant features from the data set as a part of the Pre-Processing phase[9,10]. The SDSDR and FSF Feature Selection algorithms are novel and greatly reduce the dimensionality of the data set equaling to 62.33%. Only 81 features are selected out of a total 215 features. Furthermore, PCA is applied to reduce dimensionality of the data set. 81 Features are reduced to 59 after this step. The proposed EKMPRFG outperforms DroidFusion in terms of PrecM, RecB and WFM rates on both the data sets but drops only in PrecB and RecM values in case of MALGENOME-215. The performance metrics are recorded after a tenfold cross validation.

The proposed model is required to be tested on other data sets as well and the time required to carry out the entire process must be reduced so that online classification of the Android samples may be carried out.

REFERENCES

1. Niranjan A, Nitish A, P Deepa Shenoy and Venugopal K R, "Security in Data Mining-a Comprehensiv *Computer Science and Technology*, vol. 16, no. 5, 2017, pp. 52-73.
2. Y. Zhou and X. Jiang, "Dissecting Android Malware: Characterization and evolution", In proc.2012 IEEE Symposium on Security and Privacy (SP),San Francisco, CA, USA, 20-23 May, 2012 , pp. 95-109.
3. D. Arp, M. Spreitzenbarth, M. Hubner, H. Gascon, and K. Rieck, "Drebin Efficient and Explainable Detection of Android Malware in Your Pocket" In proc. 20th Annual Network & Distributed System Security Symposium
4. S. Y. Yerima and S. Sezer, "DroidFusion: A Novel Multilevel Classifier Fusion Approach for Android Malware Detection," in IEEE Transactions on Cybernetics, vol. 49, no. 2, pp. 453-466, Feb. 2019. doi: 10.1109/TCYB.2017.2777960.
5. `T. Ban, T. Takahashi and S. Guo "Integration of Multi-modal Features for Android Malware Detection Using Linear SVM" In proc. 11th Asia Joint Conference on Information Security, 2016.
6. Z. Ni, M. Yang, Z. Ling, J. N. Wu and J. Luo, "Real-Time Detection of Malicious Behavior in Android Apps," In proc. Int. Conf. on Advanced Cloud and Big Data (CBD), Chengdu, 2016, pp. 221-227.
7. Z. Wang, J. Chai, S. Chen and W. Li, "DroidDeepLearner: Identifying Android malware using deep learning" IEEE 37th Sarnoff Symposium, Newark, NJ, 2016, pp. 160-165.
8. S. Wu, P. Wang, X. Li, Y. Zhang. Effective detection of android malware based on the usage of data flow APIs and machine learning. Information and Software Technology, Vol.75, 2016, Pages 17-25, ISSN 0950-5849.
9. Asha S Manek, Samhitha M R, Shruthy S , Veena H Bhat, P Deepa Shenoy, M. Chandra Mohan, Venugopal K R, L M Patnaik "RePID-OK: Spam Detection using Repetitive Pre-processing", *IEEE CUBE 2013 Conference, ISBN : 978-1-4799-2234-5*, pp. 144-149, November 15-16, 2013.
10. Niranjan A, Akshobhya K M, P Deepa Shenoy, Venugopal K R," EKMC: Ensemble of kNN using MetaCost for Efficient Anomaly Detection", Volume 4, Issue 5, Pages 401-408, OCT 2019, ASTESJ, https://dx.doi.org/10.25046/aj040552

AUTHORS PROFILE

Niranjan A., is a full time Research Scholar, pursuing PhD in the Department of Computer Science and Engineering at University Visvesvaraya College of Engineering, Bangalore University, India. He is a Mtech Graduate from Jain University Bangalore. He is a Life member of ISTE, IEEE member and a member of CSTA and his current research interests include Feature Selection Models, Ensemble Models for User Protection against Online Attacks and Classifiication Techniques of Machine learning.

K. S. S. S. Abhishikth, is a B.Tech Graduate in Computer Science & Engineering from K L Deemed to be University, Guntur, India. He is currently Pursuing his Master's in Computer Science from the University of South Florida, USA. His current research interests include Feature Selection Models, Ensemble Models for User Protection against Online Attacks and Classifiication Techniques of Machine learning.

P. Deepa Shenoy, is currently working as Professor in the Department of Computer Science and Engineering, University Visvesvaraya College of Engineering, Bangalore University, Bangalore, India. She did her doctorate in the area of Data Mining from Bangalore University in the year 2005. Her areas of research include Data Mining, Soft Computing, Biometrics, Bioinformatics, Image Processing and Social Media Analysis. She has published more than 150 papers in refereed International Conferences and Journals.

Venugopal K. R., is currently working as Vice Chancellor, Bangalore University, Bangalore. He obtained his Bachelor of Engineering from University Visvesvaraya College of Engineering. He received his Master's degree in Computer Science and Automation from Indian Institute of Science Bangalore. He was awarded Ph.D in Economics from Bangalore University and Ph.D in Computer Science from Indian Institute of Technology, Madras. He has a distinguished academic career and has degrees in Electronics, Economics, Law, Business Finance, Public Relations, Communications, Industrial Relations, Computer Science and Journalism. He has authored and edited 64 books on Computer Science and Economics, which include Petrodollar and the World Economy, C Aptitude, Mastering C, Microprocessor Programming, Mastering C++ and Digital Circuits and Systems etc., He has filed 101 patents. During his three decades of service at UVCE he has over 700 research papers to his credit. His research interests include Computer Networks, Wireless Sensor Networks, Parallel and Distributed Systems, Digital Signal Processing and Data Mining. He is a Fellow of IEEE and ACM Distinguished Educator.