International Journal on Advanced Science Engineering Information Technology

Vol.8 (2018) No. 6
ISSN: 2088-5334

Classification of Polymorphic Virus Based on Integrated Features

Isredza Rahmi A Hamid[#], Sharmila Subramaniam[#], Edi Sutoyo[*], Zubaile Abdullah[#]

[#]*Information Security Interest Group (ISIG), Faculty Computer Science and Information Technology,*
Universiti Tun Hussein Onn Malaysia, 86400 Johor, Malaysia
E-mail: rahmi@uthm.edu.my, sharmilasubramaniam71@gmail.com, zubaile@uthm.edu.my

[*]*School of Industrial Engineering, Telkom University, 40257 Bandung, West Java, Indonesia*

Abstract—**Standard virus classification relies on the use of virus function, which is a small number of bytes written in assembly language. The addressable problem with current malware intrusion detection and prevention system is having difficulties in detecting unknown and multipath polymorphic computer virus solely based on either static or dynamic features. Thus, this paper presents a classification of polymorphic virus based on integrated features. The integrated feature is selected based on Information Gain rank value between static and dynamic features. Then, all datasets are tested on Naïve Bayes and Random Forest classifiers. We extracted 49 features from 700 polymorphic computer virus samples from Netherland Net Lab and VXHeaven, which includes benign and polymorphic virus function. We spilled dataset based on 60% for training and 40% for testing. The performance metric of accuracy value, receiver operating characteristic and mean absolute error are compared between two algorithms in the experiment of static, dynamic and integrated features. Our proposed integrated features manage to achieve 98.5% of accuracy value using the highest rank feature selection.**

Keywords— **classification; polymorphic virus; integrated features; naïve bayes classifiers; random forest classifiers.**

I. Introduction

The Internet has become an essential thing in life, and it is expected to overgrow. However, this left all connected computers susceptible to misuse and abuse. Recently, there has been a remarkable growth in a number of malware as well. Once the computer is online, anyone can have access to the network. So, intruders can attack the networks easily. Hence, malicious software attacks have become a common problem in recent years. Malware is a program or file that can damage a computer and collect computer user's information without permission that includes computer viruses, worms, Trojan horses and spyware. Malware Computer virus is a malware that replicates by reproducing itself or infecting other programs by modifying them during execution. Infecting computer programs can include data files or the boot sector of the hard drive.

A computer virus can be categorized into an encrypted, metamorphic and polymorphic virus. Each computer virus has to be classified accordingly for better intrusion detection and prevention system. Thus, classification of computer virus has highly significance in the anti-virus industry. Computer virus classification has drawn many considerations from many researchers. Several good classification techniques such as signature-based [1]–[3] and behavior based [4]–[7] have been developed to address the polymorphic computer virus problems. However, some new malware samples keep on increasing despite escalating growth of anti-malware services and technologies [8], [9], [17].

Generally, established virus detection utilizes virus function, in which it is a sequence of bytes in the machine code. This research is conducted focusing on polymorphic virus signature based on static, dynamic and integrated features. The behavior of the signature is analyzed using machine learning approaches, which are Random Forest and Naïve Bayes algorithms. To analyze the polymorphic virus signature, 49 of distinct static and dynamic features of raw data collected and experimented. Features of polymorphic computer virus are ranked into Highest_Rank, Lowest_Rank, Medium_Rank, and Random_Rank by using Information Gain (IG) value. The performance metric of accuracy value, Receiver Operating Characteristic (ROC) and mean absolute error are compared between two algorithms in the experiment of static, dynamic and integrated features. This study aims to propose an integrated feature for polymorphic virus classification to improve the accuracy rate, reduce error rate with good ROC value.

A. Related Works

A polymorphic virus can mutate its code and to make a new signature generation. Most of the computer users are not aware of the malware attacks. This is because the computer virus never gives a hint when it will attack. Therefore, it is essential to predict computer viruses attacks.

A polymorphic computer virus can be grouped into either static or dynamic features. It is difficult to analyze the polymorphic computer virus by using either static or dynamic features because the multipath polymorphic computer virus has both features. Therefore, the classification of polymorphic computer virus based on integrated features will have a significant contribution to the anti-virus industry.

B. Malware Classification

Malware classification can be grouped into two techniques that are anomaly-based and signature-based. Anomaly-based classification technique uses standard Android behavior information during instruction inspection to make a decision either the application is malicious or not. The specification-based is a particular type of anomaly-based classification. Specification-based techniques decide the maliciousness of a program through intrusion inspection that depends on rule set of valid behavior.

Any programs that breach the specification are supposed to be anomalous and malicious. The effective signature-based approach relies on the signature of malicious behavior.

TABLE I
THE ADVANTAGES AND DISADVANTAGES OF STATIC AND DYNAMIC APPROACH [10]

	Advantage	**Disadvantage**
Specification-based (Static analysis)	• Fast and safe • Low level of false positives.	Difficulty in analyzing unknown malware.
Signature-based (Dynamic analysis)	Good in detecting unknown malware	The difficulty is analyzing multipath malware.

Each of the classification techniques can employ one of three different approaches, which are static, dynamic, or integrated. Table 1 shows the advantages and disadvantages of dynamic and static analysis. To overcome the limitation of an existing method, the integrated approach is proposed in this research.

C. Previous Work on Classification Techniques

This section examines the present static feature work and the dynamic feature research. There are other classify malware based on a machine learning approach [22], [23]. Work by Leder et al. [12] proposed a classification of metamorphic variants based on static features. Usually, the virus scanner overlooks Metamorphic malware because of its ability to modify the code and structure consistently throughout infection. This work uses a small number of malware samples based on a static analysis technique.

Ye et al. [13] presented a classifier based on an analysis of API execution calls. They test a large collection dataset tat consists of 35,000 and 15,000 malware and cleanware samples respectively. They executed the Android malware analysis by various data mining techniques and managed to achieve 88% accuracy.

Work by [14] developed an integrated system called ISMCS that made of feature exactor, malware categorizer using weighted subspace clustering method and malware signature generator modules. ISMCS categorized 2029 malware from 407 families. Their method uses IDA Pro Disassembler to extract function call. The experimental results show that the ISMCS system outperforms K-Means and hierarchical clustering algorithms which are closed to 79% accuracy.

Islam, Tian, Batten and Versteeg [11], [18], [19], proposed a tool that detects mobile self-encrypting and polymorphic viruses. Traditional pattern matching techniques cannot detect these types of viruses as they are designed to circumvent pattern matching techniques. This tool is allowed to decrypt itself in an emulator of the Operating System. Their framework combines the static features of function length and printable string information extracted from malware samples into a single test to classify the malware. They input 1400 unpacked malware sample along with 151 clean files using k-fold cross-validation. Their work achieved an accuracy of over 98%.

Generally, there are current works which use a behavioral analysis approach to classify the malware. In [20], the authors propose a behavior-based automated classification method based on behavioral analysis profile. The report contains the status change caused by executable and event transferred from Win32API calls and other parameters. The feature vectors consist of 13000 malware sample which is input to the SVM. They can classify with an average accuracy of 83.3%.

Work by [21] develops a behavioral model based on attribute-grammars, including semantic attributes and rules which has two layers. The first layer interprets the collected instructions, API calls, and arguments and classifies these operations as well as the involved objects according to their purpose in the malware lifecycle. The second detection layer remains generic and is interoperable between the different abstraction components. They tested about 200 samples of Portable Executable (PE) malware, 200 samples of Visual Basic Script malware and 50 clean files. This technique only achieved 51% accuracy of PE malware. However, it performed much better on Visual Basic scripts with 89% accuracy rate.

Our work differs than previous works in such a way that, we proposed an integrated approach which considered Function Length Pattern (FLP), Printable String Information (PSI) and Application Programming Interfaces (API) features. Then, we improved the feature matrix using Information Gain (IG) algorithm to assess the feature value in classifying malware. The dataset was then tested using Naïve Bayes and Random Forest based on Accuracy Rate, Error rate, Receiver Operating Characteristic. These algorithms are the simplest and most commonly used classification methods. Random Forest can deal with unbalanced and missing data while Naive Bayes performs well when we have multiple classes and working with text classification.

II. Material and Method

In this section, we discuss the proposed classification of polymorphic virus based on integrated features. We will first introduce the classification model, feature selection and extraction process

A. Virus Classification Model

Figure 1 shows the polymorphic virus classification model. We use 700 raw data collected from Netherland Net Lab [3] and VXHeaven [16]. We consider three types of features: 26 Function Length Pattern (FLP), 11 Printable String Information (PSI) and 12 Application Programming Interfaces (API). These features had been ranked into Highest_Rank, Lowest_Rank, Medium_Rank, and Random_Rank by using Information Gain (IG) value. The IG algorithm is used to make a decision which attributes in a given set of training feature vectors is most important for discriminating between the classes to be learned. Consider the dataset represented by a feature vector, F in the form of $(x, y) = (x_1, x_2, x_3, \dots, x_n, y)$ where $x_a \in val(a)$ is the value of a^{th} attribute of feature vector x and y is the corresponding class label.

We rank the features into various types to evaluate the performance of the selected features based on the level of importance. Then, we split the data into 60:40 ratios for training and testing data respectively. Training data is used for implemented to build up a model. Testing data is used to validating the model built. Next, each ranked features are classified using Naïve Bayes and Random Forest classifiers via WEKA tools.

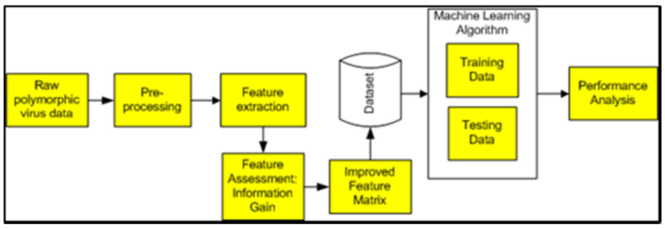

Fig. 1: Polymorphic Virus Classification Model

B. Feature Selection and Extraction

In the feature selection process, the selected features are extracted from a polymorphic virus that has been extorted in .xml file format. The next step in the process is to generate components of a feature vector by analyzing the database. Table 2 shows the description of 26 Function Length Pattern (FLP), Table 3 portrays 11 Printable String Information (PSI) and Table 4 depicts 12 Application Programming Interfaces (API) that used in the experiment. Then, the integrated feature is selected based on the rank value calculated using Information Gain (IG) algorithm.

TABLE II
FUNCTION LENGTH PATTERN (FLP) FEATURES

Num	String	Description
1	LoadLibraryW	Loads the specified module into the address space of the calling process
2	HeapAlloc	A handle to the heap from which the memory allocated. This handle is returned by the HeapCreate or GetProcessHeap function.
3	HeapFree	Frees a memory block
4	NtOpenKey	Opens the specified registry key
5	GetProcAddress	Retrieves the address of an exported function or variable from the specified dynamic-link library
6	RegOpenKeyExW	Opens the specified registry key
7	ISBadReadPtr	Verifies that the calling process has read access to the specified range of memory.
8	CId	Security Identifiers. A security identifier (SID) is a unique value of variable length used to identify a trustee.
9	LocalAlloc	Allocates fixed memory. The return value is a pointer to the memory object.
10	GetCurrentProcess	Retrieves a pseudo handle for the current process
11	GetCurrentThread	Retrieves the thread identifier of the calling thread.
12	NtQueryValueKey	If the lpValueName registry value does not exist, RegQueryValueEx returns ERROR_FILE_NOT_FOUND
13	ISBadWritePtr	This function is obsolete and should not be used. that the memory pointed to is safe to use
14	RegDeleteValueA	Removes a named value from the specified registry key
15	RegOpenKeyA	Does not create the specified key if the key does not exist in the
16	GetModuleHandleA	Returns a handle to a mapped module without incrementing
17	RegQueryValueExA	When calling the RegQueryValueEx function with hKey set to the HKEY_PERFORMANCE_DATA
18	Socket	Make a socket that is bound to a specific transport service provider.
19	ntohS	Change a u_short from TCP/IP network byte order to host byte order (which is little-endian on Intel processors).
20	Sleep	Suspends the execution of the current thread until the time-out interval elapses. To enter an alert able wait state, use the SleepEx function.
21	FindFirStFileA	Opens a search handle and returns information about the first
22	FindCloSe	Closes a file search handle opened by the FindFirstFile, FindFirstFileEx, FindFirstFileNameW, FindFirstFileNameTransactedW
23	GetModuleFileName A	Does not retrieve the path for modules that were loaded
24	GetFileAttributeSA	GetFileAttributesA(ANSI) See also DeviceIoControl File Attribute Constants File Management Functions
25	CreateFileA	Produce a handle to a communications resource.
26	ReadFile	Parameters hFile [in] A handle to the device (for example, a file, file stream, physical disk, volume, console buffer, tape drive, socket, communications resource

TABLE III
PRINTABLE STRING INFORMATION (PSI) FEATURES

Num	String	Description
1	RegSetValueExA	If the lpValueName registry value does not exist, RegQueryValueEx returns ERROR_FILE_NOT_FOUND
2	RegSetValueExW	Sets the data and type of a specified value under a registry
3	GetDriveTypeA	Determines whether a disk drive is a removable
4	RegOpenKeyExA	This handle is returned by the RegCreateKeyEx or RegOpenKeyEx function
5	ShellExecuteA	Operates on a specified file.
6	cloSeK	Closes a handle to the specified registry key.
7	GetFileSize	Will get the file type
8	WinExec	The function returns when the started process calls the GetMessage function, or a time-out limit is reached.
9	GetProceSSVerSion	Retrieves timing information for the specified process.
10	RegDeleteKeyA	Deletes a subkey and its values from the specified platform-specific view of the registry.
11	RegCreateKeyExA	Creates all missing keys in the specified path.

TABLE IV
APPLICATION PROGRAMMING INTERFACE (API) FEATURES

Num	String	Description
1	FindNextFileA	Continues a file search from a previous call to
2	inet_addr	The inet_addr function converts a string containing an IPv4
3	connect	Enables you to provide feedback directly to the team's
4	SetFilePointer	This function stores the file pointer in two LONG values
5	WriteFile	may fail with an error
6	HeapCreate	Creates a private heap object
7	SetThreadPriority	Sets the priority value
8	CreateProceSSA	Fails if the total size of the environment
9	GetModuleHandleW	Returns a handle to a mapped module
10	GlobalAlloc	Parameters uFlags
11	recv	to read incoming data
12	SetFileAttributeSA	Sets the attributes for a file or directory

III. RESULT AND DISCUSSION

In this section, Classification of Polymorphic virus based on integrated features selected from static and dynamic features has been done.

A. *Experimental Setup*

This research constructed three types of experiments which the features had been ranked into Highest_Rank, Lowest_Rank, Medium_Rank, and Random_Rank by using Information Gain (IG) value. Next, each ranked features are tested using Naïve Bayes [15] and Random Forest classifiers. Then, the performance metric for both classification algorithm are compared regarding accuracy rate, receiver operating characteristic, and mean absolute error.

In Experiment 1 (static features), 37 static features of the polymorphic computer virus have been ranked from various features group as Function Length Pattern (FLP) and Printable String Information (PSI) as shown in Table 5 and 6. Table 5 shows the IG value for 26 static features specifically the Function Length Pattern (FLP) features. The “LoadLibraryW” feature has the highest IG value that is 0.49620. Meanwhile, the “GetFileAttributes,” “CreateFileA” and “ReadFile” have the lowest IG value of 0.00000. Highest value shows the features are the most relevant features to be applied in the experiment with the lowest redundant and high robustness characteristics. Table 6 shows that “RegSetValueExA” feature has the highest IG value as compared to 10 other features selected.

In Experiment 2 (dynamic features), 12 dynamic features from Application Programming Interface (API) group features have been ranked based on IG value as shown in Table 7. Feature “FindNextFileA” is the most significant features with IG value of 0.32489. The least IG value is “recv” and “SetFileAttributeSA” which is 0.0000.

In Experiment 3 (integrated features), the three best features are selected from each type of feature: Function Length Pattern (static), Printable String Information (static) and Application Programming Interfaces (dynamic). Table 8 shows the combination of features for each type of feature rank: Highest_Rank, Medium_Rank Random_Rank, and Lowest_Rank. The performance metric for each experiment was compared between both classifiers. In order to ensure that the accuracy and effectiveness of the experiments, all types of feature rank are using the same split ratio size for training and testing which is 60:40 respectively.

TABLE V
INFORMATION GAIN (IG) RANKING FILTER FOR FLP (STATIC FEATURES)

Ranking	Information Gain Value	Features
1	0.49620	LoadLibraryW
2	0.46478	HeapAlloc
3	0.32489	HeapFree
4	0.28996	NtOpenKey
5	0.23310	GetProcAddress
6	0.18640	RegOpenKeyExW
7	0.18430	ISBadReadPtr
8	0.17200	Cid
9	0.16556	LocalAlloc
10	0.15520	GetCurrentProcess
11	0.09080	GetCurrentThreadId
12	0.08766	NtQueryValueKey
13	0.06984	ISBadWritePtr
14	0.04630	RegDeleteValueA
15	0.04281	RegOpenKeyExA
16	0.02750	GetModuleHandleA
17	0.02490	RegQueryValueExA
18	0.02287	socket
19	0.01750	ntohs
20	0.01437	Sleep
21	0.00613	FindFirstFileA
22	0.00552	FindClose
23	0.00401	GetModuleFileNameW
24	0.00000	GetFileAttributesA
25	0.00000	CreateFileA
26	0.00000	ReadFile

TABLE VI
INFORMATION GAIN (IG) RANKING FILTER FOR PSI (STATIC FEATURES)

Ranking	Information Gain Value	Features
1	0.72780	RegSetValueExA
2	0.51842	RegSetValueExW
3	0.49902	GetDriveTypeA
4	0.15956	RegOpenKeyExA
5	0.12433	ShellExecuteA
6	0.08710	cloSeK
7	0.03078	GetFileSize
8	0.01619	WinExec
9	0.00625	GetProceSSVerSion
10	0.00000	RegDeleteKeyA
11	0.00000	RegCreateKeyExA

TABLE VII
INFORMATION GAIN (IG) RANKING FILTER FOR API (DYNAMIC FEATURES)

Ranking	Information Gain Value	Features
1	0.32489	FindNextFileA
2	0.28996	inet_addr
3	0.23310	connect
4	0.18640	SetFilePointer
5	0.18430	WriteFile
6	0.17200	HeapCreate
7	0.15520	SetThreadPriority
8	0.12433	CreateProceSSA
9	0.09080	GetModuleHandleW
10	0.08766	GlobalAlloc
11	0.00000	recv
12	0.00000	SetFileAttributeSA

TABLE VIII
INTEGRATED RANKED FEATURES

Dataset	Feature	Type
Lowest_Rank	GetFileAttributesA	FLP
	CreateFileA	FLP
	ReadFile	FLP
	GetProceSSVerSion	PSI
	RegDeleteKeyA	PSI
	RegCreateKeyExA	PSI
	GlobalAlloc	API
	recv	API
	SetFileAttributeSA	API
Medium_Rank	RegOpenKeyExW	FLP
	ISBadReadPtr	FLP
	Cid	FLP
	RegOpenKeyExA	PSI
	ShellExecuteA	PSI
	cloSeK	PSI
	SetFilePointer	API
	WriteFile	API
	HeapCreate	API
Highest_Rank	LoadLibraryW	FLP
	HeapAlloc	FLP
	HeapFree	FLP
	RegSetValueExA	PSI
	RegSetValueExW	PSI
	GetDriveTypeA	PSI
	FindNextFileA	API
	inet_addr	API
	connect	API
Random_Rank	LocalAlloc	FLP
	GetCurrentProcess	FLP
	GetCurrentThreadId	FLP
	GetFileSize	PSI
	WinExec	PSI
	RegOpenKeyExA	PSI
	SetFilePointer	API
	WriteFile	API
	HeapCreate	API

B. *Performance Metric*

In order to measure the effectiveness of integrated features for the classification approach, we refer to three possible outcomes as Accuracy, Receiver Operation Curve (ROC) and Mean Absolute Error. The performance metric for accuracy is defined as,

$$Accuracy = \frac{Number\ of\ correct\ prediction}{Total\ number\ of\ prediction}.$$

Equivalently, the performance of the classification model can be expressed in term of its error rate, which given by,

$$Error\ rate = \frac{Number\ of\ wrong\ prediction}{Total\ number\ of\ prediction}.$$

Next, the ROC figure is a curve graphically displaying the transaction involving sensitivity and specificity for each cutoff value. It gives you an idea about weighted average values of two classes of malware. The ROC value is used to evaluate the effectiveness of malware classes in all experiments. The best class prediction will have larger ROC values.

C. *Result and Discussion*

In the experiment, the performance metric of Accuracy, Receiver Operating Characteristics (ROC) and Mean Absolute Error are discussed.

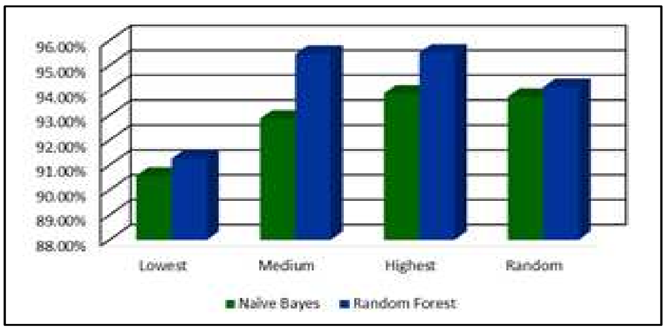

Fig. 2. Classification accuracy for Experiment 1 (static features)

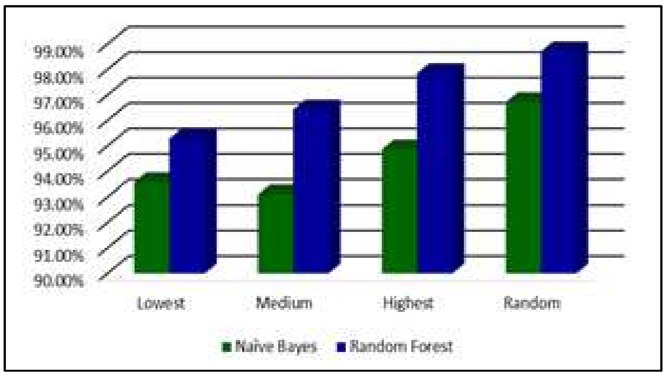

Fig. 3. Classification accuracy for Experiment 2 (dynamic features)

1) Accuracy value: The accuracy value for all set of feature rank based on static features in Experiment 1 setup is depicted in Figure 2. Highest_Rank feature selection achieved the highest accuracy which is 98.5% classified using Random Forest algorithm. Figure 3 shows the accurate result for dynamically based features on Experiment 2 setup. Both algorithms, Naïve Bayes and Random Forest, achieved the highest accuracy when tested on Random_Rank feature.

Figure 4 shows the accuracy value for all sets of feature rank based on integrated features in Experiment 3. The Random_Rank feature achieved the highest accuracy which is 99.7% which classified using Random Forest algorithm. Moreover, the Random Forest algorithm also managed to classify the Lowest_Rank feature up to 95.13% accuracy value. This proves that Random Forest presents reliable and clear enhancement in accuracy mainly for many class classifications. Moreover, Random Forest algorithm gives better result against overfitting issue.

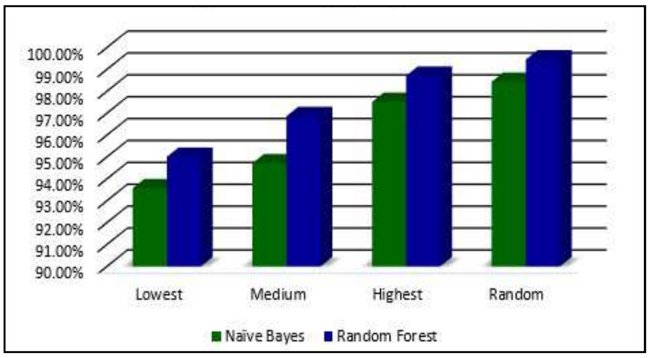

Fig. 4. Classification accuracy for Experiment 3 (integrated features)

2) Mean Absolute Error Value: Figure 5, 6 and 7 show the mean absolute error for experiment 1, 2 and three based on static, dynamic and integrated approach respectively. The Lowest_Rank feature has the highest error, which is 0.57 when tested using Random Forest classification algorithm and 0.63 using Naïve Bayes algorithm respectively as shown in Figure 5.

Figure 6 portrays the mean absolute error for dynamic classification method. Lowest_Rank feature has the highest mean absolute error which is 0.42 for Naïve Bayes classification algorithm and 0.401 for Random Forest classification algorithm.

Based on Figure 7, the Lowest_Rank feature has the highest error, which is 0.57 and 0.63 when tested using the Random Forest algorithm and Naïve Bayes algorithm respectively. This shows that the Lowest_Rank feature selection has the least significance features for virus classification used for all experiments.

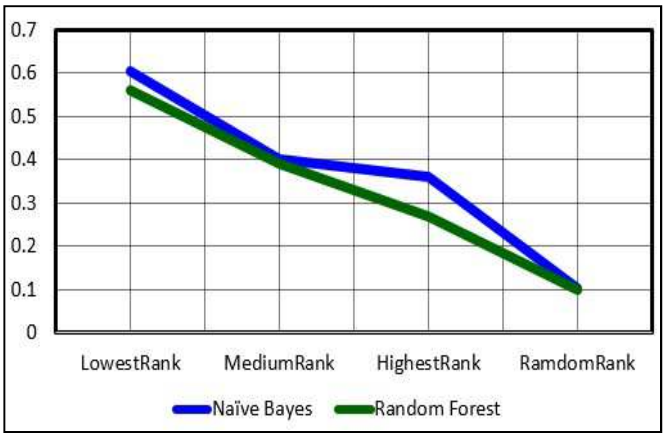

Fig. 5. Mean Absolute Error value for Experiment 1 (static features)

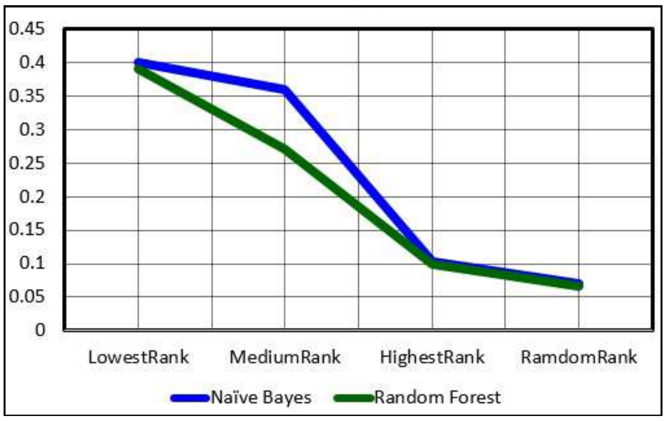

Fig. 6. Mean Absolute Error value for Experiment 2 (dynamic features)

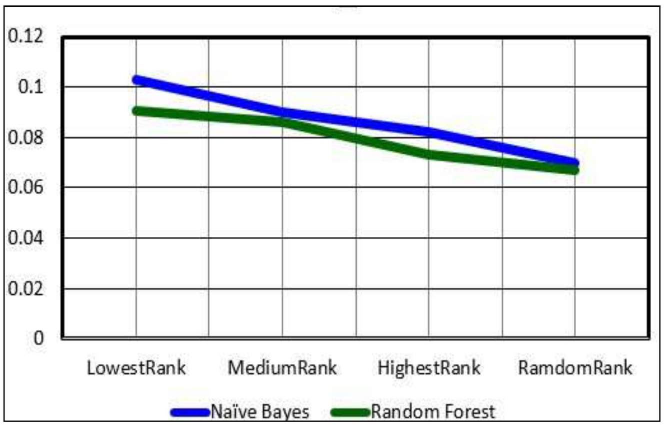

Fig. 7. Mean Absolute Error value for Experiment 3(integrated features)

3) Receiver Operating Characteristic (ROC) value: Figure 8, 9 and 10 shows the Receiver Operating Characteristic (ROC) for all experiment setup. The best ROC value is when it is near one. Figure 8 shows that the Random_Rank features have the highest value which is 0.998 and 0.849 when tested on the Random Forest algorithm and Naïve Bayes classification algorithm respectively. While Figure 9 shows the Random_Rank features also has the highest ROC value for both algorithms which is 0.9 for Naive Bayes and 0.995 for the Random Forest algorithm. Figure 10 shows the Random_Rank feature recorded the highest ROC value which was 0.936 and 0.964 for Naïve Bayes and Random Forest classification algorithm respectively.

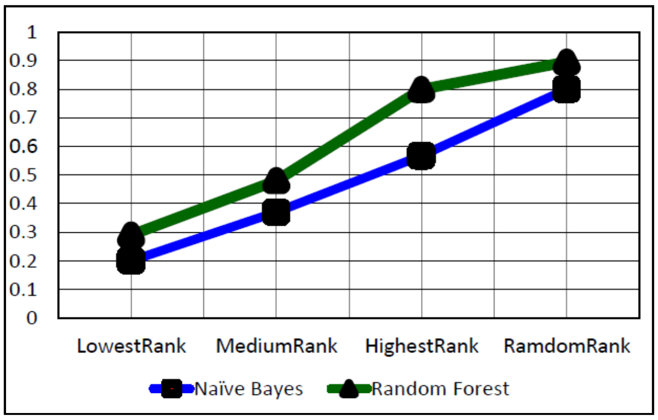

Fig. 8. Receiver Operation Curve (ROC) for static features

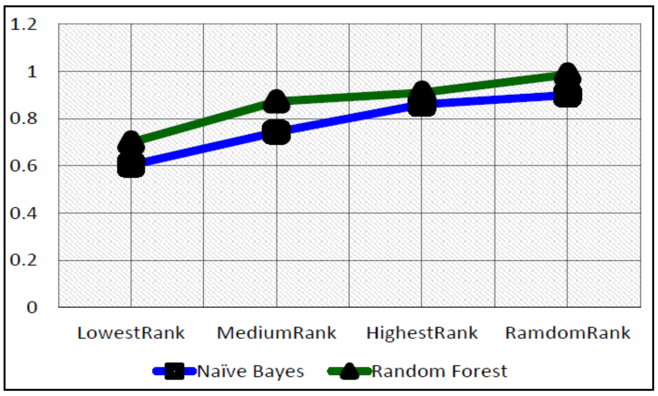

Fig. 9. Receiver Operation Curve (ROC) for dynamic features

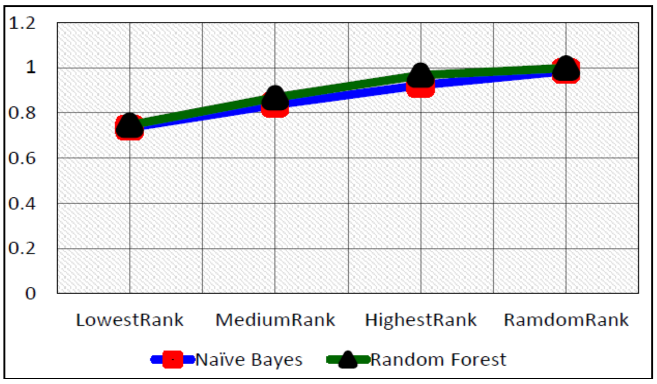

Fig.10. Receiver Operation Curve (ROC) for integrated features

IV. Conclusion

The experiment conducted between static features, dynamic features and integrated features on the various ranking group (Lowest_Rank, Medium_Rank, Highest_Rank, and Random_Rank). The Random Forest algorithm is the best classification algorithm because of it achieved high accuracy result, low means absolute error and high ROC value as compare to Naïve Bayes classifier for the most dataset. Our proposed integrated features show the promising result when tested on the various ranking group regarding accuracy; low means absolute error and high ROC value as compared to static and dynamic features. This shows that the selected integrated feature contribute well for virus classification and the given set of training feature vectors is most important for discriminating between the classes to be learned.

We plan to explore the integrated features to improve polymorphic computer virus classification. For instance, future work has to use an optimal number of features. This may reveal the information on processing time and also the effectiveness of the system since good features can affect the performance of the classifier. Thus, the integrated mechanism can update the classifier when notice new possible features in real time

Acknowledgment

The authors express appreciation to the Universiti Tun Hussein Onn Malaysia (UTHM). Postgraduate Research Grant vot number U610 supports this research, Short Term Grant vot number U653 and Gates IT Solution Sdn. Bhd. under its publication scheme.

References

[1] S. Chaumette, O. Ly, and R. Tabary, “Automated extraction of polymorphic virus signatures using abstract interpretation,” Proc. - 2011 5th Int. Conf. Netw. Syst. Secure. NSS 2011, pp. 41–48, 2011.

[2] A. A. E. Elhadi, “Malware Detection Based on Hybrid Signature Behaviour Application Programming Interface Call Graph,” Am. J. Appl. Sci., vol. 9, no. 3, pp. 283–288, 2012.

[3] H. Lim, Y. Yamaguchi, H. Shimada, and H. Takakura, “Malware classification method based on sequence of traffic flow BT - 1st International Conference on Information Systems Security and Privacy, ICISSP 2015, February 9, 2015 - February 11, 2015,” 2015, pp. 230–237.

[4] G. Nascimento and M. Correia, “Anomaly-based intrusion detection in software as a service,” Proc. Int. Conf. Dependable Syst. Networks, pp. 19–24, 2011.

[5] R. Islam, R. Tian, L. Batten, and S. Versteeg, “Classification of Malware Based on String and Function Feature Selection,” 2010 Second Cybercrime Trust. Comput. Work., pp. 9–17, 2010.

[6] A. Tang, S. Sethumadhavan, and S. Stolfo, “Unsupervised Anomaly-based Malware Detection using Hardware Features,” Proc. Int. Symp. Res. Attacks, Intrusion Detect., p. 1, 2014.

[7] R. Sekar, a Gupta, J. Frullo, T. Shanbhag, a Tiwari, H. Yang, and S. Zhou, “Specification-based anomaly detection: a new approach for detecting network intrusions,” CCS ’02 Proc. 9th ACM Conf. Comput. Commun. Secur., pp. 265–274, 2002.

[8] E. Al Daoud, I. Jebril, and B. Zaqaibeh, “Computer virus strategies and detection methods,” Int. J. Open Probl. Comput. Math., vol. 1, no. 2, pp. 122–129, 2008.

[9] A. Techniques, “MALWARE: Threats and Attacks Part 1-D: How to protect from Malware attacks, Antivirus Techniques Malware threats and attacks,” 2012.

[10] Idika, N. (2007). A Survey of Malware Detection Techniques.

[11] R. Islam, R. Tian, L. Batten, and S. Versteeg, “Classification of Malware Based on String and Function Feature Selection,” 2010.

[12] F. Leder, B. Steinbock, and P. Martini, "Classification and detection of metamorphic malware using value set analysis," *2009 4th International Conference on Malicious and Unwanted Software (MALWARE)*, Montreal, QC, 2009, pp. 39-46.

[13] Y. Ye, T. Li, Q. Jiang, and Y. Wang, "CIMDS: Adapting Postprocessing Techniques of Associative Classification for Malware Detection," in *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 3, pp. 298-307, May 2010.

[14] K. Huang, Y. Ye and Q. Jiang, "ISMCS: An intelligent instruction sequence based malware categorization system," *2009 3rd International Conference on Anti-counterfeiting, Security, and Identification in Communication*, Hong Kong, 2009, pp. 509-512.

[15] N. Bayes, “Naive Bayes classifier,” pp. 1–9, 2006.

[16] “VXHeaven_Dataset,” 2014 IEEE 11th Consumer Communications and Networking Conference (CCNC), 2014.

[17] A. R. Kakad, S. G. Kamble, S. S. Bhuvad, and V. N. Malavade, “Study and Comparison of Virus Detection Techniques,” Int. J. Adv. Res. Comput. Sci. Softw. Eng., vol. 4, no. 3, pp. 251–253, 2014.

[18] R. Tian, R. Islam, L. Batten, and S. Versteeg, “Differentiating Malware from Cleanware Using Behavioural Analysis,” pp. 23–30, 2010.

[19] R. Islam, R. Tian, L. M. Batten, and S. Versteeg, “Journal of Network and Computer Applications Classification of malware based on integrated static and dynamic features,” vol. 36, pp. 646–656, 2013.

[20] H. Zhao, M. Xu, N. Zheng, J. Yao and Q. Ho, "Malicious Executables Classification Based on Behavioral Factor Analysis," *2010 International Conference on e-Education, e-Business, e-Management and e-Learning*, Sanya, 2010, pp. 502-506.

[21] Grégoire Jacob, Hervé Debar, Eric Filiol, "Malware detection using attribute-automata to parse abstract behavioral descriptions," *CoRR abs/0902.0322*, 2009.

[22] I.R.A Hamid, N.S Khalid, N.A. Abdullah, N. H. Ab Rahman, C.C. Wen, “Android Malware Classification Using K-Means Clustering Algorithm,” 2017 IOP: Conference Series: Materials Science and Engineering, Melaka, 2017, vol. 226.

[23] A. Zulkifli, I.R.A Hamid, W.M Shah, and Z. Abdullah, “Android Malware Detection Based on Network Traffic Using Decision Tree Algorithm,” 2018 *International Conference on Soft Computing and Data Mining*, pp. 485-494.