Proceeding of the IEEE 28th
Canadian Conference on Electrical and Computer Engineering
Halifax, Canada, May 3-6, 2015

Hybrid Concentration Based Feature Extraction Approach for Malware Detection

Pengtao Zhang and Ying Tan, *Senior Member, IEEE*

***Abstract*— In this paper, a hybrid concentration based feature extraction (HCFE) approach is proposed. The HCFE approach extracts the hybrid concentration (HC) of a sample in both the global resolution and the local resolution. The HC of a sample characterizes the sample more precisely and completely by taking the global information and local information into account at the same time. With the help of the co-operation of the global and local information, the HC discards the bias of the global concentration (GC) to the global information and the local concentration (LC) to the local information, respectively. In order to incorporate the HCFE approach into the procedure of malware detection, a HC-based malware detection (HCMD) method is proposed. Eight groups of experiments on three public malware datasets are exploited to evaluate the effectiveness of the HCMD method using cross validation. Comprehensive experimental results suggest that the HC of a sample extracted by the HCFE approach characterizes the sample more precisely and completely than the GC and LC. The proposed HCMD method outperforms the GC-based and the LC-based malware detection methods in all the experiments for about 1.05% and 0.28% on average, respectively.**

I. Introduction

Malware is a general term for all the malicious code that is a program designed to harm or secretly access a computer system without the owners' informed consent [1]. According to the malware's method of operation, the malware can be roughly broken down into several categories, such as computer virus, Trojan horse and worm. Some adware is also regarded as malware. The malware costs hundreds of millions of dollars every year all over the world. It has been one of the most terrible threats to the security of the computers worldwide [2].

To address the problem of malware detection, a variety of malware detection methods have been proposed, while various commercial anti-malware products are available in the market. These anti-malware solutions can be classified into two categories: static methods and dynamic methods. The static methods attempt to detect malware without actually running any code. They are mainly based on machine learning and data mining methods, and heuristic theories (such as artificial immune theory [3][4]). The static methods usually work on the binary string or application programming interface (API) calls of a program, so they are portable and can be deployed on personal computers. The dynamic methods keep watch over the execution of every program during run-time, observe its behavior, and stop it once it tries to harm the system, such as behavior blockers, virtual machines. The dynamic methods bring too much extra load. Hence they are usually used to analyze malware in the computer security firms instead of to detect malware in personal computers.

Inspired by human immune system, the immune concentration has been proposed as an effective feature [5]. There are two concentration based features so far : the global concentration (GC) and the local concentration (LC). The GC was proposed firstly for spam detection [5][6] and later applied to detect malware [7]. Although the GC-based methods perform very well in the two problems, the GC merely contains the global information of a sample extracted in the global resolution. This design results in its bias to the global information, ignoring the local information, and a high diluent risk. To overcome the diluent risk of the GC, the LC was proposed [8][9]. The LC zooms out the concentration information and stores the position-correlated information implicitly by defining a local area. However, the LC ignores the global information and merely characterizes a sample from the perspective of a local resolution, resulting in its bias to the local information. Furthermore, the stability of the position-correlated information should be under suspicion. How to design and extract a discriminating immune concentration based feature, discarding the bias of the GC and LC to the global information and local information, respectively, becomes a worthwhile work.

In this paper, a hybrid concentration based feature extraction approach is proposed by taking inspiration from the GC and LC. The HCFE approach extracts the hybrid concentration (HC) of a sample in both the global resolution and the local resolution. The HC of a sample characterizes the sample more precisely and completely by taking the global and local information into account at the same time. It discards the bias of the GC and LC, respectively, to the global information and local information. In order to incorporate the HCFE approach into the procedure of malware detection, a HC-based malware detection (HCMD) method is proposed.

Extensive experimental results demonstrate that the proposed HCMD method is effective to detect unseen malware. It outperforms the GC-based and LC-based malware detection methods in the eight groups of experiments on the three malware datasets for about 1.08% and 0.28% on average, respectively.

The rest of the paper is organized as follows. In Section II, we introduce the related work. In Section III, we give the definition of the HC and describe the HCFE approach in de-

Y. Tan is the correspondent author with the Department of Machine Intelligence, School of Electronics Engineering and Computer Science, Peking University, Beijing, 100871, China. E-mail: ytan@pku.edu.cn.

P.T. Zhang is a PhD candidate with the Department of Machine Intelligence, School of Electronics Engineering and Computer Science, Peking University, Beijing, 100871, China. E-mail: pengtaozhang@gmail.com.

978-1-4799-5829-0/15/$31.00 ©2015 IEEE

tail. Section IV introduces the proposed HCMD method. The experimental setup, selection of parameters and experimental results are presented in Section V. Finally, we conclude the paper with a detailed discussion.

II. Related work

Inspired by human immune system, a global concentration based feature construction (CFC) approach was proposed for spam detection [5][6]. In the CFC approach, the GC is defined as a two-element concentration vector, consisting of 'self' concentration and 'non-self' concentration, one concentration for one class. The two elements in the GC are constructed through the 'self' gene library and 'non-self' gene library, respectively. The experimental results suggested that the CFC approach performed very well on the corpora PU1 and Linq. The GC was latter applied to detect malware [7] and achieved good results. However, the GC merely contains the global information of a sample extracted in the global resolution. This design results in its bias to the global information of a sample, ignoring the local information of a sample. Furthermore, the GC involves a great dilute risk due to its formula where the number of the distinct genes in a sample is taken as the denominator.

On the basis of the GC, a feature named local concentration was proposed which brought down the dilute risk of the GC to a certain extent [8][9]. Different from the CFC approach which works on the whole sample to collect the global information of the sample, the LC based feature extraction (LCFE) approach works on the local areas in a sample to collect the detailed local information of the sample. It extracts a series of concentration vectors in a series of local areas in a sample. All the concentration vectors are connected orderly to form the LC. In this way, the position-correlated information is considered to be extracted and stored in the LC. The dilute risk of the LC is brought down by using local areas in a sample. In order to incorporate the LCFE approach into the whole process of spam filtering and malware detection, respectively, two LC-based models were designed. The experimental results showed that the two models had promising performance. However, the LC ignores the global information and merely characterizes a sample from the perspective of a local resolution, resulting in its bias to the local information.

III. Hybrid Concentration based feature extraction approach

A. Hybrid Concentration

Inspired by human immune system, the immune concentration, as an effective feature, has been applied to spam filtering and malware detection successfully. Based on the immune concentration, importing the concept of multi-resolution, the HC is proposed in this paper.

Definition A hybrid concentration is constructed by the immune concentration vectors which are extracted in more than one resolution, e.g. both the global resolution and the local resolution.

In this paper, the HC is written as $< IC_1, ..., IC_m >$, where $IC_i (i = 1, 2, ..., m)$ denotes the concentration vector extracted in the *i-th* resolution, and m is the number of the resolutions in the HC. We make use of the GC extracted in the global resolution and the LC extracted in the local resolution to construct the HC. It is a two-resolution concentration, written as

$$HC =< GC, LC >,$$
$$GC =< GC_1, GC_2, ..., GC_M >,$$
$$LC =< LC_1, LC_2, ..., LC_N >,$$
$$LC_i =< LC_{i1}, LC_{i2}, ..., LC_{iM} >$$

where M is the number of the classes in a classification problem, and N is the number of the local areas defined in the LC. $GC_j (j = 1, 2, ..., M)$ is the global concentration value of class j in the whole sample. $LC_i (i = 1, 2, ..., N)$ is the local concentration vector in local area i, and the LC_{ij} is the local concentration value of class j in local area i.

It is easy to see that the HC consists of the GC and LC, which are extracted in the global and local resolutions, respectively. So the HC contains both the global and local information of a sample. Through the co-operation of the global and local information, the HC overcomes the disadvantages of the GC and LC which only characterizes a sample in a single resolution. In this way, the HC characterizes a sample more precisely and completely than the GC and LC alone.

The dimension of the HC is $(1 + N) * M$. To a specific classification problem, M is a constant. Hence the dimension of the HC is determined by the number of the local areas N defined in the LC. Furthermore, we could extract the LC in different local resolutions by different N, to obtain more coarse or detailed local information.

B. Flow chart of the HCFE Approach

There are two main stages in the HCFE approach : (1) generation of gene libraries; (2) feature extraction. The flow chart of the HCFE approach is shown in Figure 1, where $L_i (i = 1, 2, ..., M)$ denotes the gene library of class i, M is the number of the classes in a classification problem.

In the first stage, the HCFE approach generates the gene library for each class. The gene in this paper, which is borrowed from the biological genetics, is the basic element of a sample. The specific definition of the gene varies with the specific application field. In the field of text categorization, a gene is usually defined as a single word or a phrase. For malware detection, a gene is usually expressed as a binary string.

In the procedure of the generation of gene libraries, the HCFE approach traverses the samples in the training set to count the document frequency of every gene. We take the IG as the gene selection criteria which helps to select the genes with the highest information content, and compute the IG of all the genes. Other criteria, such as document frequency, mutual information and χ^2 statistic [23] can also be used.

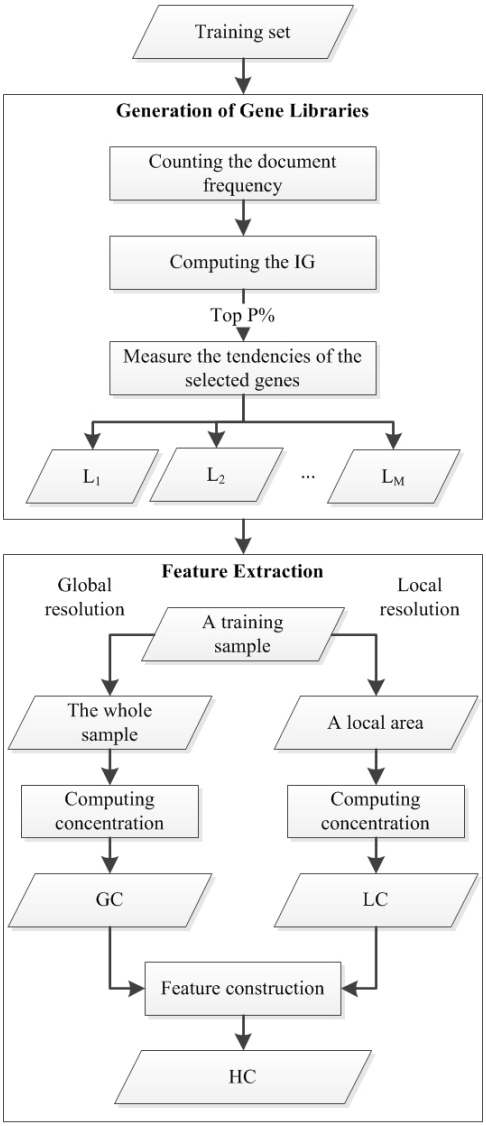

Fig. 1. The flow chart of the HCFE approach

Then the top P% genes with the highest IG value are selected to construct the gene library. However, the classes of the genes tending to appear in are unknown. With the help of Formula 1, we measure the class information of every gene and classify a gene into a specific class. We believe the gene g in the gene library L_i tends to appear in and represent class i. In this way, the gene libraries of all the classes are generated.

$$T(g, C_i) = P(g|C_i) - \sum_{j=1 \bigwedge j \neq i}^{M} P(g|C_j) \quad (1)$$

where g is a gene, C_i is a class and $P(g|C_i)$ denotes the proportion of samples in the C_i in which the gene g is presented, $i, j = 1, 2, ..., M$.

The $T(g, C_i)$ measures the tendency of the gene g to the class C_i. The larger of the $T(g, C_i)$ dedicates that the g tends to appear in the C_i. If $T(g, C_i) > \theta$, we believe that the g tends to represent the C_i, called a gene of class C_i in this paper.

After generating the gene libraries, the HCFE approach extracts the HC in both the global and local resolutions. The formula to compute the concentration value is shown as

$$IC(C_i) = N_i/W \quad (2)$$

where $IC(C_i)$ is the concentration value of the class C_i in the current feature extraction area. N_i is the number of the distinct genes which appear in both the gene library L_i and the current feature extraction area. W denotes the number of the distinct genes in the current feature extraction area.

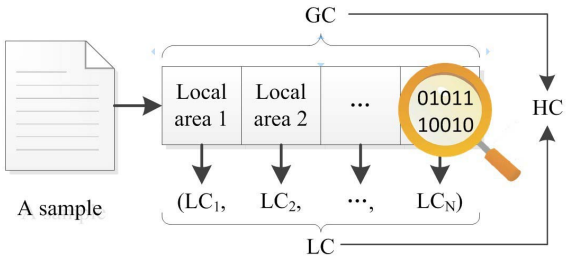

Fig. 2. The schematic diagram of the feature extraction

Figure 2 illustrates the procedure of the feature extraction. From Figure 2, a sample is treated as a data stream. We get the GC of the sample in the global resolution. In order to get the LC, we split the data stream into N parts, i.e., N local areas, and observe the data stream in the N local areas, respectively. It is just like we observe the sample with a magnifier. The local area helps us collect the local information of the sample. Then we compute the local concentration vector in each local area. The LC of the sample is constructed by connecting the local concentration vectors in all the local areas orderly. Through a feature construction process, the HC of the sample is generated from the GC and LC.

The feature construction process can be formulated as

$$HC = f(GC, LC) \quad (3)$$

This paper defines $HC = f(GC, LC) =< GC, LC >$.

Up to this time, the HCFE approach extracts the HC of a sample successfully. The HC, which is a vector with lower data dimension, is the output of the HCFE approach and is taken as the input of a classifier.

C. Strategies for Definition of Local Areas

In this paper, a local area is defined as a gene string with variable-length. The length of a local area is determined by the length of a sample and the number of the local areas defined in the LC.

There are two strategies of defining local areas in [8] : local area with fixed-length and local area with variable-length. In the field of spam filtering, both the two strategies result in good performance without marked difference [8]. In the malware detection method [9], a local area is defined

as a local area with fixed-length which is set to 500 bytes. The number of the local areas is 40. This method performs very well using these parameters. However, this method only extracts the concentration information from the top 500 * 40 = 20,000 bytes $\approx$ 20 KB of a sample and ignores the remaining content. In the anti-malware field, we cannot make sure that the malicious codes in a malware appear in its top 20KB binary string, and we have to traverse a sample. As a result of the local area with fixed-length, it is easy for a malware to evade from the method in [9]. Actually, other fields, such as spam filtering, have the similar problem. Hence the local area in this paper is defined as the local area with variable-length, and the set of all the local areas covers the whole sample.

IV. HC-BASED MALWARE DETECTION METHOD

The HCMD method is proposed in this section. Its two main stages are shown in Figure 3.

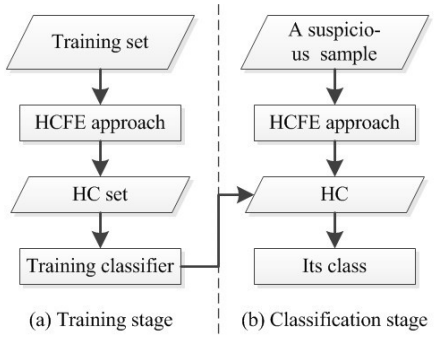

Fig. 3. The training and classification stages of the HCMD method

In the HCMD method, a gene is defined as a binary string of length 4 bytes. The gene in this length contains enough information to identify a meaningful operation, and is the same as the 4-Gram in [11][12]. Set $\theta = 0$.

In the training stage, a sliding window is used to traverse the whole training set to obtain the document frequency of every gene. The length of the sliding window is set to 4 bytes, so the content in a sliding window is a gene. The sliding window moves forward one byte at a time. There is an overlap of 3 bytes between two adjacent sliding windows. The overlap makes the gene be able to capture not only genes of length 4 bytes, but also longer genes implicitly. Then the HCFE approach outputs the HC set of all the training samples. The HC set are taken as the input of a classifier to train the classifier.

In the classification stage, the trained classifier makes classification to the HC of a suspicious sample.

V. EXPERIMENTS

A. Experimental Datasets

Comprehensive experiments are conducted on three public malware datasets: CILPKU08 dataset, Henchiri dataset and VXHeanvens dataset. The three datasets and their composition documents can be download from www.cil.pku.edu.cn/resources/. The benign program dataset used here consists of the files in portable executable format from Windows XP and a series of applications, which are the main punching bag of malware.

TABLE I
EXPERIMENTAL PLATFORM

CPU	Core 2 Duo 3.00 GHz
RAM	8 GB
Operating System	Win 7 64-bit
Programming Language	C# (.NET Framework 3.5), Matlab 2010a
Thread	Single Thread
Compiler	Visual Studio 2008

B. Experimental Setup

The support vector machine (SVM), realized by libSVM [24], is taken as the classifier of the proposed HCMD method. Other classifiers, such as k-nearest neighbor, naive bayes and decision tree, can also be used. The parameters of the SVM are set as follows: $g = 0.25, c = 4$. We do not take many works to optimize the parameters of the SVM as it is not the focus of the HCMD method. The detailed information of the experimental platform is listed in Table I.

The area under the receiver operating characteristic curve (AUC), which is widely used to evaluate the classification performance in the field of data mining, is utilized as the performance evaluation criteria in this paper.

In the experiments of Section V-D, all the experiments are taken using 5-fold cross validation to get a more precise and believable evaluation of the proposed HCMD method. In both the CILPKU08 dataset and Henchiri dataset, most of the malware are computer viruses. Hence we ignore the categories of the malware and carry on 5-fold cross validation directly in the two datasets, respectively. The VXHeavens dataset contains 7128 malware which fall into six categories, so we split the dataset into six smaller datasets: backdoor, constructor, miscellaneous, trojan, virus and worm. The miscellaneous includes malware such as DoS, Nuker, Hacktool and Flooder, while the malware in the other five smaller datasets, respectively, fall into a category. We take 5-fold cross validation on each of the six smaller datasets.

In all the experiments, there is no overlap between a training set and a test set. That is to say, to a training set, the malware in a test set are unseen malware. This setting increases the reliability of the experiments.

Sum up, eight groups of experiments are taken on three public malware datasets using 5-fold cross validations. The 95% confidence intervals are computed to look into the stability of the proposed HCMD method.

The GC-based malware detection (GCMD) method proposed in [7] and the LC-based malware detection (LCMD) method presented in [9] are imported for comparisons.

C. Selection of Parameters

This section is to select the two parameters in the HCFE approach, i.e., the proportion of the genes ($P\%$) and the

number of the local areas (*N*).

The dataset used in this section consists of 1048 benign programs, randomly selected from the benign program dataset, and 1048 computer viruses from the VXHeavens dataset. We randomly split the benign programs into two sets with 524 programs for each set, one for training and the other for testing. The same partition was done to the computer viruses. The 524 benign programs and 524 viruses made up the training set, and the test set consisted of the remaining benign programs and viruses.

Here we optimize the two parameters using the grid search method, where $P = 5, 10, ..., 50$ and $N = 10, 20, ..., 100$. We do not try larger *P*. As we know, the larger *P* means that more genes with less information content are selected into the gene library. The class tendencies of these poor genes are unclear. They bring less information content for the classification and lead to false positive or false negative. The above analysis is supported by the experimental results below.

The experimental results on the above dataset are plotted in Figure 4 with cubic spline interpolation method.

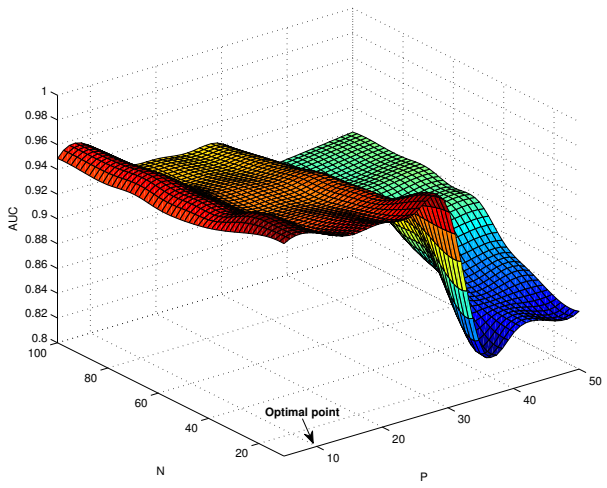

Fig. 4. The experimental results in the selection of parameters

Figure 4 illustrates that when $P = 10$, regardless of the value of the N, the AUCs of the HCMD method are fairly good and stable. The AUCs of the HCMD method drop down dramatically when $P > 30$. It is because there are too many genes with less information content in all the gene libraries. These poor genes are helpless for the classification, and bring in harmful information which confuses the classifier. This result proves the above analysis about the larger *P*. When we set $P = 10$, we found the influence of the N to the AUC of the HCMD method was not remarkable. When $N = 10$, the dimension of the HC is relatively lower, only 22 dimensions, which is much lower than that of the features in [11][12] which are usually hundreds of dimensions, and the HCMD method gets the optimal AUC: 0.9724. Hence we set $P = 10, N = 10$.

TABLE II

EXPERIMENTAL RESULTS OF THE HCMD METHOD

Dataset	Training set	Test set
CILPKU08	**0.9991 ± 0.000234**	**0.9984 ± 0.000758**
Henchiri	**0.9992 ± 0.000053**	**0.9981 ± 0.001113**
Backdoor	**0.9921 ± 0.000654**	**0.9749 ± 0.006304**
Constructor	**0.9810 ± 0.001016**	**0.9687 ± 0.007591**
Miscellaneous	**0.9804 ± 0.005168**	**0.9494 ± 0.009814**
Trojan	**0.9847 ± 0.001185**	**0.9596 ± 0.004817**
Virus	**0.9904 ± 0.000976**	**0.9731 ± 0.008639**
Worm	**0.9546 ± 0.011265**	**0.9331 ± 0.016261**

TABLE III

EXPERIMENTAL RESULTS OF THE GCMD METHOD

Dataset	Training set	Test set
CILPKU08	0.9984 ± 0.000209	0.9976 ± 0.000526
Henchiri	0.9984 ± 0.000109	0.9970 ± 0.001283
Backdoor	0.9887 ± 0.000777	0.9711 ± 0.007953
Constructor	0.9759 ± 0.001725	0.9651 ± 0.011248
Miscellaneous	0.9624 ± 0.004145	0.9288 ± 0.014902
Trojan	0.9753 ± 0.001397	0.9525 ± 0.004842
Virus	0.9851 ± 0.001288	0.9650 ± 0.009839
Worm	0.9156 ± 0.016786	0.8942 ± 0.032133

TABLE IV

EXPERIMENTAL RESULTS OF THE LCMD METHOD

Dataset	Training set	Test set
CILPKU08	0.9983 ± 0.000259	0.9973 ± 0.000559
Henchiri	0.9983 ± 0.00018	0.9966 ± 0.001944
Backdoor	0.9908 ± 0.000685	0.9740 ± 0.006196
Constructor	0.9800 ± 0.000861	0.9672 ± 0.00904
Miscellaneous	0.9739 ± 0.005254	0.9438 ± 0.009501
Trojan	0.9802 ± 0.001742	0.9527 ± 0.003589
Virus	0.9888 ± 0.001037	0.9692 ± 0.008283
Worm	0.9537 ± 0.011121	0.9322 ± 0.016678

In order to compare to the GCMD and LCMD methods fairly, we optimize the parameters of the two methods in the same way. When $P = 10$, the GCMD method gets the best AUC. And the optimal parameters for the LCMD method are: $P = 10, N = 10$.

D. Experimental Results

Eight groups of experiments are conducted on three public malware datasets in this section. The experimental results of the proposed HCMD method are shown in Table II. The experimental results of the GCMD method and the LCMD method are listed in Table III and Table IV, respectively, for comparison. The results in the bold font indicate the best results in the three methods.

In all the training sets, the HCMD method performed better than the GCMD and LCMD methods. The results suggest that the HCMD method is able to learn much better than the other two methods since the HC extracted by the HCFE approach contains much information than the GC and

LC, which characterizes a sample in two resolutions and has a strong discriminating ability.

Table III and IV show that the GCMD method is 0.03% better than the LCMD method in the test sets of the CILP-KU08 dataset and Henchiri dataset, whereas the LCMD method is better than the GCMD method in the test sets of the other six experiments for about 1.04% on average. The above results demonstrate that the GCMD method and LCMD method could not gain any great advantage over the other.

We can see that the proposed HCMD method is very stable and always better than the other two methods from Table II, regardless of the training sets and test sets. In all the test sets of the whole experiments, the HCMD method is 1.05% better than the GCMD method which is a big increase, and the average AUC of the HCMD method is 0.28% larger than that of the LCMD method without any losing in any experiments.

Without increasing the time complexity, the HCMD method performs very well and stably with a little more computing, so the HC is considered to be able to characterize a sample more precisely and completely than the GC and LC alone, and could be regarded as a replacement of the GC and LC. The time complexity to extract the GC, LC and HC, i.e., the time complexity of the CFC, LCFE and HCFE approaches, will be discussed in detail in the next section.

The 95% confidence intervals of the three methods were relatively small from Table II, III, IV. They suggested that the results of these methods were very stable and believable.

VI. Conclusion

The HCFE approach extracts the HC of a sample in both the global resolution and the local resolution. With the help of the co-operation of the global and local information, the HC is able to characterize a sample more precisely and completely, discarding the bias of the GC to the global information and the LC to the local information, respectively. When the GCMD method and the LCMD method lead to divergence, the HCMD method is considered to be able to make a more reasonable classification.

Extensive experimental results have demonstrated that the proposed HCMD method is effective to detect unseen malware. It outperforms the GCMD method and LCMD method in the eight groups of experiments on the three public malware datasets for about 1.08% and 0.28% on average, respectively.

In future work, we intend to study the co-operation of the concentrations in different resolutions in depth, and construct a better feature.

VII. ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China under grants No. 61170057 and 60875080.

References are important to the reader; therefore, each citation must be complete and correct. If at all possible, references should be commonly available publications.

References

[1] http://en.wikipedia.org/wiki/Malware.
[2] F-Secure Corporation, “F-Secure Reports Amount of Malware Grew by 100% during 2007”, Press release, 2007.
[3] S. Forrest, A.S. Perelson, L. Allen L, et al, “Self-nonself Discrimination in a Computer,” 1994 IEEE Computer Society Symposium on Research in Security and Privacy, pp. 202-212, 1994.
[4] S. Forrest, S.A. Hofmeyr, A. Somayaji, et al, “A Sense of Self for Unix Processes,” In Proceedings of 1996 IEEE Symposium on Security and Privacy, pp. 120-128, 1996.
[5] Y. Tan, C. Deng, G.C. Ruan, “Concentration Based Feature Construction Approach for Spam Detection,” Proceedings of International Joint Conference on Neural Networks, pp. 3088-3093, 2009.
[6] G.C. Ruan, Y. Tan, “A Three-layer Back-propagation Neural Network for Spam Detection Using Artificial Immune Concentration,” Soft computing, Vol. 14, pp. 139-150, 2010.
[7] W. Wang, P.T. Zhang, Ying Tan, “An Immune Concentration Based Virus Detection Approach Using Particle Swarm Optimization,” Lecture Notes in Computer Science, vol. 6145, pp. 347-354, 2010.
[8] Y.C. Zhu, Y. Tan, “A Local Concentration Based Feature Extraction Approach for Spam Filtering,” IEEE Transactions on Information Forensics and Security, vol. 6, no. 2, pp. 486-497, 2011.
[9] W. Wang, P.T. Zhang, Y. Tan, and X.G. He, “An Immune Local Concentration-Based Virus Detection Approach,” Journal of Zhejiang University-SCIENCE C (Computers and Electronics), vol. 11, no. 3, pp. 1-13, 2011.
[10] 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. 38C49, 2001.
[11] J.Z. Kolter, M.A. Maloof, “Learning to Detect Malicious Executables in the Wild,” In Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 470C478, 2004.
[12] J.Z. Kolter, M.A. Maloof, “Learning to Detect and Classify Malicious Executables in the Wild,” Journal of Machine Learning Research vol. 7, pp. 2721-2744, 2006.
[13] W.J. Li, K. Wang, S.J. Stolfo, B. Herzog, “Fileprints: Identifying Filetypes by N-gram Analysis,” IEEE Information Assurance Workshop, USA, IEEE Press, 2005.
[14] S.J. Stolfo, K. Wang, W.J. Li, “Towards Stealthy Malware Detection,” Advances in Information Security, vol. 27, pp. 231-249, Springer, USA, 2007.
[15] W.J. Li, S.J. Stolfo, A. Stavrou, E. Androulaki, A.D. Keromytis, “A Study of Malcode-Bearing Documents,” International Conference on Detection of Intrusions & Malware, and Vulnerability Assessment (DIMVA), pp. 231-250, Springer, Switzerland, 2007.
[16] S.M. Tabish, M.Z. Shafiq, M. Farooq, “Malware Detection using Statistical Analysis of Byte-Level File Content,” In Proceedings of the ACM SIGKDD Workshop on CyberSecurity and Intelligence Informatics, pp. 23-31, 2009.
[17] T. Li, “Dynamic Detection for Computer Virus Based on Immune System(In Chinese),” Sci China Inf Sci, vol. 39, no. 4, pp. 422-430, 2009.
[18] P.T. Zhang, W. Wang, Y. Tan, “A Malware Detection Model based on a Negative Selection Algorithm with Penalty Factor,” Sci China Inf Sci, vol. 53, no. 12, pp. 2461-2471, 2010.
[19] Y.F. Ye, T. Li, Q.S. Jiang, et al, “CIMDS: Adapting Postprocessing Techniques of Associative Classification for Malware Detection,” Systems, Man, and Cybernetics, vol. 40, no. 3, pp. 298-307, 2010.
[20] D. Komashinskiy, I. Kotenko, “Malware Detection by Data Mining Techniques Based on Positionally Dependent Features,” 18th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), pp. 617-623, 2010.
[21] W. Wang, P.T. Zhang, Y. Tan, et al, “A Hierarchical Artificial Immune Model for Virus Detection,” International Conference on Computational Intelligence and Security, pp. 1-5, 2009.
[22] O. Henchiri, N. Japkowicz, “A Feature Selection and Evaluation Scheme for Computer Virus Detection,” Sixth International Conference on Data Mining, pp. 891-895, 2006.
[23] Y. Yang, J.O. Pedersen, “A Comparative Study on Feature Selection in Text Categorization,” In Proceedings of ICML-97, 14th International Conference on Machine Learning, Nashville, pp. 412-420. Morgan Kaufmann, SanFrancisco, 1997.
[24] LibSVM, available on online at: “http://www.csie.ntu.edu.tw/ cjlin/libsvm/”.