Cluster-Oriented Ensemble Classifiers for Intelligent Malware Detection

Shifu Hou
Dept. of Computer Science and Electrical Engineering
West Virginia University
Morgantown, WV 26506, USA
shhou@mix.wvu.edu

Lifei Chen
School of Mathematics and Computer Science
Fujian Normal University
Fuzhou, 350117, China
clfei@fjnu.edu.cn

Egemen Tas, Igor Demihovskiy
Comodo Security Solutons, Inc,
New Jersey, NJ 07130, USA
egemen,igor@comodo.com

Yanfang Ye *
Dept. of Computer Science and Electrical Engineering
West Virginia University
Morgantown, WV 26506, USA
yanfang.ye@mail.wvu.edu

***Abstract*—With explosive growth of malware and due to its damage to computer security, malware detection is one of the cyber security topics that are of great interests. Many research efforts have been conducted on developing intelligent malware detection systems applying data mining techniques. Such techniques have successes in clustering or classifying particular sets of malware samples, but they have limitations that leave a large room for improvement. Specifically, based on the analysis of the file contents extracted from the file samples, existing researches apply only specific clustering or classification methods, but not integrate them together. Actually, the learning of class boundaries for malware detection between overlapping class patterns is a difficult problem. In this paper, resting on the analysis of Windows Application Programming Interface (API) calls extracted from the file samples, we develop the intelligent malware detection system using cluster-oriented ensemble classifiers. To the best of our knowledge, this is the first work of applying such method for malware detection. A comprehensive experimental study on a real and large data collection from Comodo Cloud Security Center is performed to compare various malware detection approaches. Promising experimental results demonstrate that the accuracy and efficiency of our proposed method outperform other alternate data mining based detection techniques.**

I. Introduction

Nowadays, as computers and Internet become increasingly ubiquitous, especially the rapid development of e-commerce, computer security becomes more and more important. Malware (short for ***mal***icious soft***ware***) is software that deliberately fulfills the harmful intent of an attacker [2], such as viruses, backdoors, spyware, trojans, worms and botnets. It has been used as the major weapon by the cyber-criminals to launch a wide range of security attacks, such as infiltrate users' computers to steal their confidential information, crash the networks, bring down servers and critical infrastructures, which present serious damages and significant financial loss to Internet users [10]. Currently, the most significant line of defense against malware is anti-malware software products, such as Symantec, Kaspersky, Comodo and Kingsofts Anti-Virus. Typically, these widely used malware detection software tools use the signature-based method [6][7] to recognize threats. Signature is a short string of bytes, which is unique for each known malware so that its future examples can be correctly classified with a small error rate [12]. However, driven by economic benefits, today's malware samples are created at a rate of thousands per day [28]. Meanwhile, in order to evade the signature-based detection, malware authors employ techniques such as polymorphism [4], metamorphism [7], packing, instruction virtualization, and emulation to bypass signatures. In order to remain effective, anti-malware industry calls for intelligent malware systems which can automatically detect malware from real and large daily sample collection.

Many research efforts have been conducted on developing intelligent malware detection systems applying data mining techniques. Such techniques have successes in classifying [5][11][13][1][21][22][29][25][32][33][34] or clustering [9][1][14][30] particular sets of malware samples, but they have limitations that leave a large room for improvement. Specifically, based on the analysis of the file contents extracted from the file samples, most of the existing researches apply only specific classification or clustering methods, but not integrate them together. Actually, the learning of class boundaries for malware detection between overlapping class patterns is a difficult problem.

To further clarify, let's first see a general data set with overlapping patterns from different classes. Excessive training of the classifier might result in overfitting thus misclassifying testing samples. On the contrary, to avoid overfitting, learning generalized boundaries will be at the cost of misclassifying some overlapping patterns. To solve this problem, in [17][18][19][20], the authors brought in clustering before classification learning. Clustering, a form of unsupervised learning, is the process of partitioning a given data set into groups (clusters), based on a defined distance measure, such that the data points in a cluster is closest to each other, and furthest from those in other clusters [27]. After clustering, the clusters can be well defined and easy to learn boundaries. If the classifier is trained on the modified data set, then they will learn the cluster boundaries with high accuracy [18]. For malware detection problem, let's consider the property of malware: malware samples can always be categorized into families by using clustering techniques [30] so that samples in the same family share some common traits. This property can be brought into classification learning for malware detection.

* Corresponding author

IEEE ICSC 2015, February 7-9, 2015, Anaheim, California, USA
978-1-4799-7935-6/15/$31.00 ©2015 IEEE

In this paper, resting on the analysis of Windows Application Programming Interface (API) calls extracted from the file samples, we develop the intelligent malware detection system using cluster-oriented ensemble classifiers. We first use clustering techniques with different Ks to generate the base classifiers, and then apply ensemble learning for conclusion fusion. To the best of our knowledge, this is the first work of applying such method for malware detection. A comprehensive experimental study on a real and large data collection from Comodo Cloud Security Center is performed to compare various malware detection approaches. Promising experimental results demonstrate that the accuracy and efficiency of our proposed method outperform other alternate data mining based detection techniques.

The rest of the paper is organized as follows. Section II discusses the related work. Section III presents the overview of our malware detection system. Section IV introduces our proposed cluster-oriented ensemble classification method for malware detection. In Section V, using the real data collection obtained from Comodo Cloud Security Center, we systematically evaluate the performance of our malware detection system in comparison with other proposed data mining methods. Finally, Section VI concludes.

II. Related work

Signature-based methods are widely used in anti-malware industry for malware detection [7]. A signature is a short string of bytes which is unique for each known malware. However, this classic signature-based method always fails to detect variants of known malware or previously unknown malware. The problem lies in the signature extraction and generation process, and in fact these signatures can be easily bypassed [22]. For example, to evade the widely-used signature-based detection, malware authors can employ techniques such as polymorphism [4], metamorphism [3], packing, instruction virtualization, and emulation. Not only the diversity and sophistication of malware have significantly increased in recent years [28], driven by economic benefits, today's malware samples are also created at a rate of thousands per day [28]. In order to remain effective, anti-malware industry calls for intelligent malware systems which can automatically detect malware from real and large daily sample collection. Accordingly, many research efforts have been conducted on developing intelligent malware detection systems applying data mining techniques.

Various classification approaches have also been applied for malware detection. Neural Networks as well as immune system were used by IBM for computer virus recognition [23]. Naive Bayes was applied to detect malicious code by Schultz et al. [21]: they used commercial virus scanner to labeled 1,001 benign software and 3,265 malware for training, and based on their testing set (206 benign files and 38 malware), they claimed that Naive Bayes classifier perform better than traditional signature-based method. Wang et al. [25] used Decision Tree to detect malware based on the same data set described in [21]. Their experimental results showed that the performance of Decision Tree was better than Naive Bayes classifier. Kolter et al. [13] compared different classification methods, including Naive Bayes, Support Vector Machine(SVM) and Decision Tree for malware detection based on the data set with 3,622 file samples (1,971 benign files and 1,651 malware). Associative classification [15], with its ability to utilize relationships among attributes, has also been applied in [29][35]. To further improve the system performance, ensemble classification has been used for malware detection [31]. An ensemble of classifiers is a set of classifiers whose individual decisions are combined in some way (typically by weighted or unweighted voting) to classify new samples [21]. Ensembles are often much more accurate than the individual classifiers that make them up [21]. However, they fail to establish any mechanism to improve the learning domain of the individual base classifiers [19].

Actually, the learning of class boundaries for malware detection between overlapping class patterns is a difficult problem. To put this into perspective, let's first consider a general data set with overlapping patterns from different classes. Excessive training of the classifier might result in overfitting thus misclassifying testing samples. On the contrary, to avoid overfitting, learning generalized boundaries will be at the cost of misclassifying some overlapping patterns. To solve this problem, in [17][18][19][20], the authors brought in clustering before classification learning. The clusters can be well defined and easy to learn boundaries [19]. If the classifier is trained on the modified data set, then they will learn the cluster boundaries with high accuracy [19]. For malware detection problem, let's see the property of malware: malware samples can always be categorized into families by using clustering techniques so that samples in the same family share some common traits. In recent years, there are several initiatives in automatic malware categorization using clustering techniques [9]. Bayer et al. [2] used locality sensitive hashing and hierarchal clustering to efficiently group large datasets of malware samples into clusters. Lee et al. [14] adopted k-medoids clustering approach to categorize the malware samples. Several efforts have also been reported on computing the similarities between different malware samples using Edit Distance (ED) measure [8] or statistical tests [24]. Resting on the analysis of instruction frequency and function-based instruction sequences, Ye et al. [30] developed an automatic malware categorization system to group malware samples into families that share some common characteristics using a cluster ensemble by aggregating the clustering solutions generated by different base clustering algorithms. This property of malware can also be brought into classification learning for malware detection.

Different from previous researches which applied only specific classification or clustering methods for malware detection, in this paper, based on a real and large data collection from Comodo Cloud Security Center, we propose using cluster-oriented ensemble classifiers for malware detection with the aim to achieve better learning and improve the detection accuracy.

III. System Architecture

In this paper, resting on the analysis of Windows API (Application Program Interface) calls extracted from the file samples which can reflect the behavior of program code pieces, we develop the intelligent malware detection system using cluster-oriented ensemble classifiers. Figure 1 shows the architecture of our malware detection system.

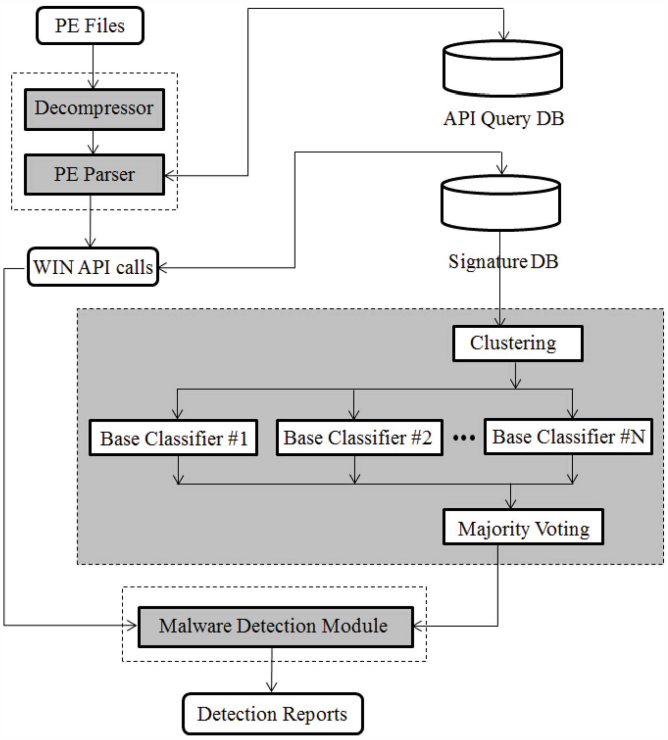

Fig. 1: Intelligent malware detection system based on cluster-oriented ensemble classifiers

- **Training:**
  1) *Feature Extractor*: It includes the modules of Decompressor and PE Parser. The functionality of the PE parser is to extract the Windows API calls (e.g., KERNEL32.DLL, OpenProcess) from the collected PE (Portable Executable) files, including malware samples and benign files. If a PE file is previously compressed by a third party binary compress tool such as UPX and ASPack Shell or embedded a homemade packer, it needs to be decompressed at first.
  2) *Base Classifiers*: With clustering, we can generate the base classifiers. To achieve diversity between the base classifiers, the extracted features in the training dataset are independently partitioned n times using K-medoids clustering algorithm with different Ks. We use the terminology n *layers* to refer to n alternative clusterings of the training dataset. For each layer, we build a base classifier (See Section IV-B for details). The decision generated by the base classifiers trained on the n alternate clusters at n *layers* can be fused to obtain the final verdict of the testing file sample.
- **Prediction:**
  For the coming unknown file samples, after feature extraction, the base classifiers are used for prediction. Then, a simple voting scheme is used to combine the base classifiers. (See Section V for details.)

IV. Cluster-Oriented Ensemble Classifiers for Malware Detection

A. Motivation

The learning of class boundaries for malware detection in real and large sample collection are not easy, which might result in either overfitting or poor generalization, mainly because of overlapping patterns from different classes in the real data set. In both cases (overfitting and poor generalization), it causes classification errors for malware detection. The situation can be explained in Fig. 2. The file samples in Fig. 2(a) contain overlapping patterns from both benign files and malware samples. Accurate learning from the training sample set by a generic classifier will generate the class boundaries (solid curve in Fig. 2(b)) which leads to overfitting and thus misclassification of test malware samples; though an alternate solution to this problem can be achieved by reducing penalties for misclassification during training, the simple decision boundaries (dashed curve in Fig. 2(b)) will cause misclassification of training as well as test file samples.

For malware detection, actually, variants of same malware family (e.g., Trojan.GameThief.YUC) always share same behaviors represented by similar Windows API calls. Considering this property, clustering, process of grouping similar patterns [19], can be used to generate multiple decision boundaries for each class (e.g., malware and benign files). Clustering the training samples in Fig. 2(a) with overlapping patterns will result in smaller clusters as in Fig. 2(c). Note that the cluster boundaries (as shown in Fig. 2(d)) are simple and easy to learn. By training, a generic base classifier can learn simple cluster boundaries that neither causes overfitting nor poor generalization. In the section below, we will introduce the base classifier construction approach.

B. Base Classifier Construction

With clustering, we can generate the base classifiers. To achieve diversity, in [17], the authors used different initial clustering parameters (e.g., seeds in k-means clustering algorithm) independently partitioning the dataset n times. Based on each time's partition, they built the corresponding base classifier. In our application, for malware detection, it's difficult to decide the number of malware families and benign file sample clusters. To solve this problem, we use different cluster validity indices to obtain the possible Ks: F1-Measure [36], Macro-F1 [36], Micro-F1 [36] and an improved *Fukuyama-Sugeno* index(NFS) [26]. NFS evaluates the partition by exploiting the compactness within each cluster and the distances between the cluster representatives. It is defined as

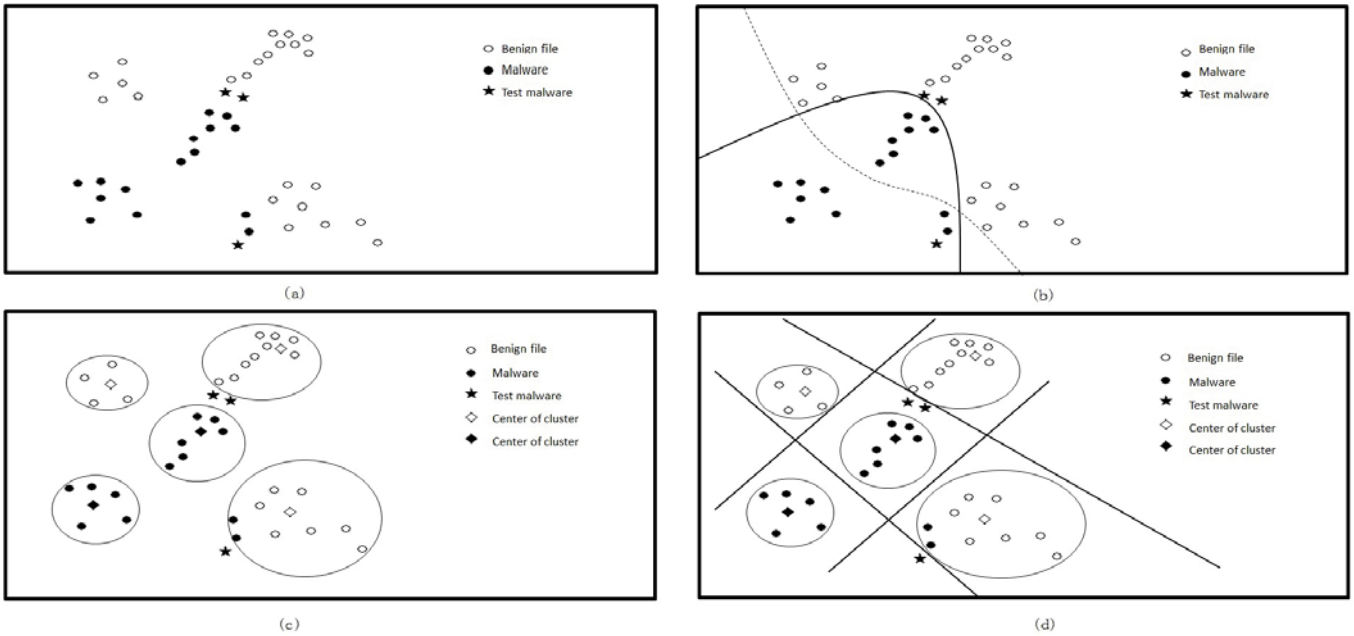

Fig. 2: Impact of clustering on a sample set. (a) Original sample set with overlapping patterns. (b) Overfitting or misclassification with different classification boundaries. (c) Clusters generated from the sample set after clustering. (d) Decision boundaries learned from clusters.

$$
\begin{aligned}
V_{NFS} &= scat(c) - sep(c) \\
&= \sum_{i=1}^{c}\sum_{k=1}^{n_i} dist(x_{i,k} - v_i) \\
&\quad - \sum_{i=1}^{c} dist(v_i - \tilde{v}) * (n_i - 1)
\end{aligned} \tag{1}
$$

where c is the number of clusters, n_i is the number of points in cluster i, and $x_{i,k}$ corresponds to the k_{th} data point that belong to cluster i.

v_i is the centroid of cluster i defined as

$$
v_i = \{x_{ik} | \underbrace{min}_{k=1,...n_i} \sum_{j=1}^{n_i} dist(x_{ik} - x_{ij}), x_{ik} \in C_i\} \tag{2}
$$

$\tilde{v}$ is the centroid of the whole dataset with n data points, which can be defined as follow

$$
\tilde{v} = \{x_k | \underbrace{min}_{k=1,...n} \sum_{j=1}^{n} dist(x_k - x_j)\} \tag{3}
$$

Scat(c) represents the compactness of the obtained clusters, while Sep(c) represents the separation between clusters.

In our case, the extracted features in the training dataset are independently partitioned *n* times using K-medoids clustering algorithm with different Ks. We use the terminology *n layers* to refer to n alternative clusterings of the training dataset. When clustering is used to partition the training file samples, the resultant clusters can be of two types: *atomic* and *nonatomic*. An *atomic* cluster contains patterns that belong

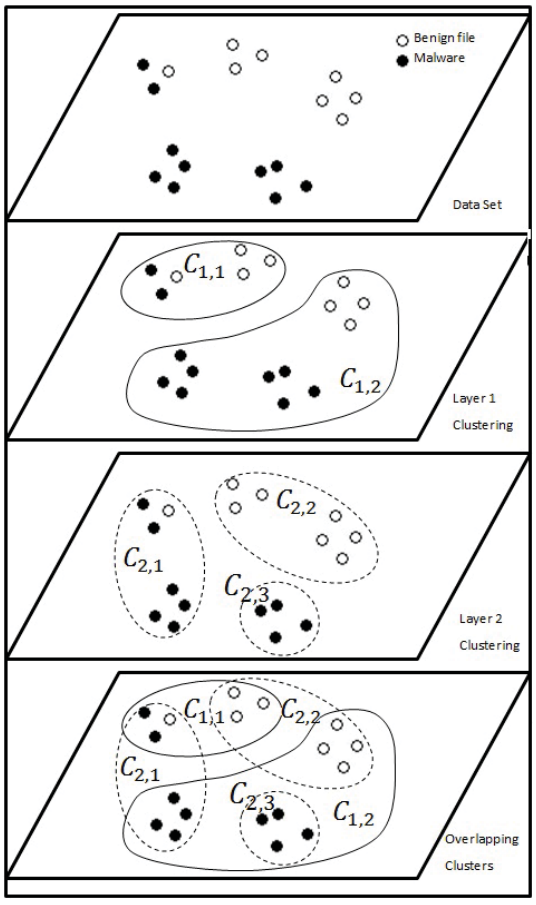

Fig. 3: Clustering of file sample set at different layers.

to the same class whereas a *nonatomic* cluster is composed of patterns from multiple classes [19]. Fig. 3 illustrates an example of a training sample set partitioned into different clusters. For *Layer 2*, $C_{2,3}$ and $C_{2,2}$ are atomic clusters whereas $C_{2,1}$ is a nonatomic cluster. Once the clustering is finished, a base classifier can be trained based on each layer: (1) for atomic clusters, the class label of each atomic cluster can be memorized for future prediction; (2) for nonatomic clusters, we will construct classification model (e.g., Support Vector Machine) based on the whole training dataset to learn the boundaries for later prediction. The training of base classifiers is further illustrated in Fig. 4 and Algorithm 1.

Input: Training sample set
Output: L base cluster-oriented classification models

Set L as the number of layers;
for $j = 1$ *to* L **do**
  Set K_j as the number of clusters;
  Randomly initialize K_j centroids and use K-medoids for partitioning;
  for $i = 1$ *to* K_j **do**
    if *all patterns belong to same class* **then**
      Memorize class label $\beta_{i,j}$ for atomic cluster $C_{i,j}$;
    end
  end
end
Build classification model using Linear SVM.

Algorithm 1: The algorithm of base cluster-oriented classifier construction

C. Cluster-Oriented Ensemble Classifiers for Malware Detection

The class of a test file sample can be predicted by first finding the appropriate cluster based on its distance (e.g., using Jaccard distance measure [16]) from the cluster centroids and then using the class label (for an atomic cluster) or the corresponding classifier (for a nonatomic cluster). The decision generated by the base classifiers trained on the *n* alternate clusters at *n layers* can be fused by using majority voting to obtain the final verdict of the test file sample. The prediction of the test sample set is shown in Fig. 5 and Algorithm 2.

V. Experimental results and analysis

In this section, we conduct three sets of experimental studies using our data collection obtained from Comodo Cloud Security Center to fully evaluate the performance of our developed malware detection system: (1) In the first set of experiments, we compare the detection performance of different classifiers; (2) In the second part of experiments, we evaluate the detection effectiveness of base cluster-oriented classifiers; (3) In the last set of experiments, we evaluate our proposed cluster-oriented ensemble classifiers on malware detection.

Input: Test sample set and
L base cluster-oriented classification models
Output: Class label for each test sample

for *each test sample* s **do**
  count = 0;
  for $j = 1$ *to* L **do**
    Compute Jaccard distances between test sample s and the centroids of the clusters;
    Find the cluster $C_{i,j}$ whose distance is smallest with s;
    if *the cluster* $C_{i,j}$ *is an atomic cluster* **then**
      $Class_j(s) = \beta_{i,j}$;
    end
    else
      Use SVM model to get class label $Class_j(s)$;
    end
    $count+ = Class_j(s)$;
  end
  if $count < Threshold$ **then**
    Label test sample s as malware;
  end
  if $count > Threshold$ **then**
    Label test sample s as benign file.
  end
end

Algorithm 2: The algorithm of test sample prediction

A. Experimental Setup

We measure the malware detection performance using the following evaluation measures:

- **True Positive (TP)**: the number of samples correctly classified as malicious files.
- **True Negative (TN)**: the number of samples correctly classified as benign files.
- **False Positive (FP)**: the number of samples mistakenly classified as malicious files.
- **False Negtive (FN)**: the number of samples mistakenly classified as benign files.
- **Accuracy (ACY)**: $\frac{TP+TN}{TP+TN+FP+FN}$
- **Recall (RC)**: $\frac{TP+TN+FP+FN}{TheNumberOfTotalFileCollection}$

The dataset we obtained from Comodo Cloud Security Center includes 50,000 file samples, half of which are benign files, while half of which are malware. After feature extraction, we got 9,648 Windows API calls from these file samples. All the experiments are conducted under the environment of Windows 8.1 operating system plus Inter (R) Core(TM) i7-4790 CUP @ 3.60GHZ and 16 GB of RAM.

B. Comparisons of Different Classifiers

In this set of experiments, we evaluate the effectiveness of malware detection results based on different classifiers: K-

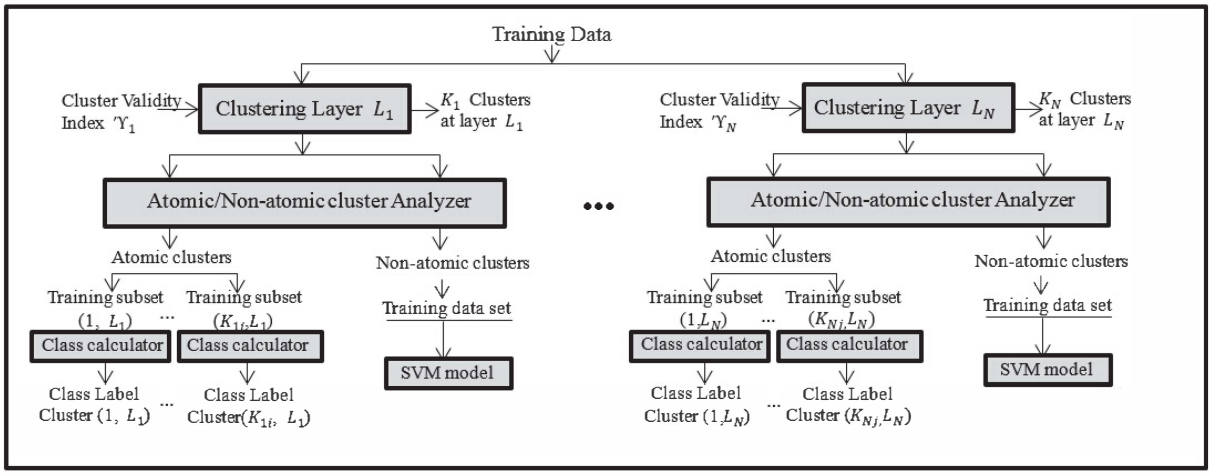

Fig. 4: Training process of base cluster-oriented classifiers

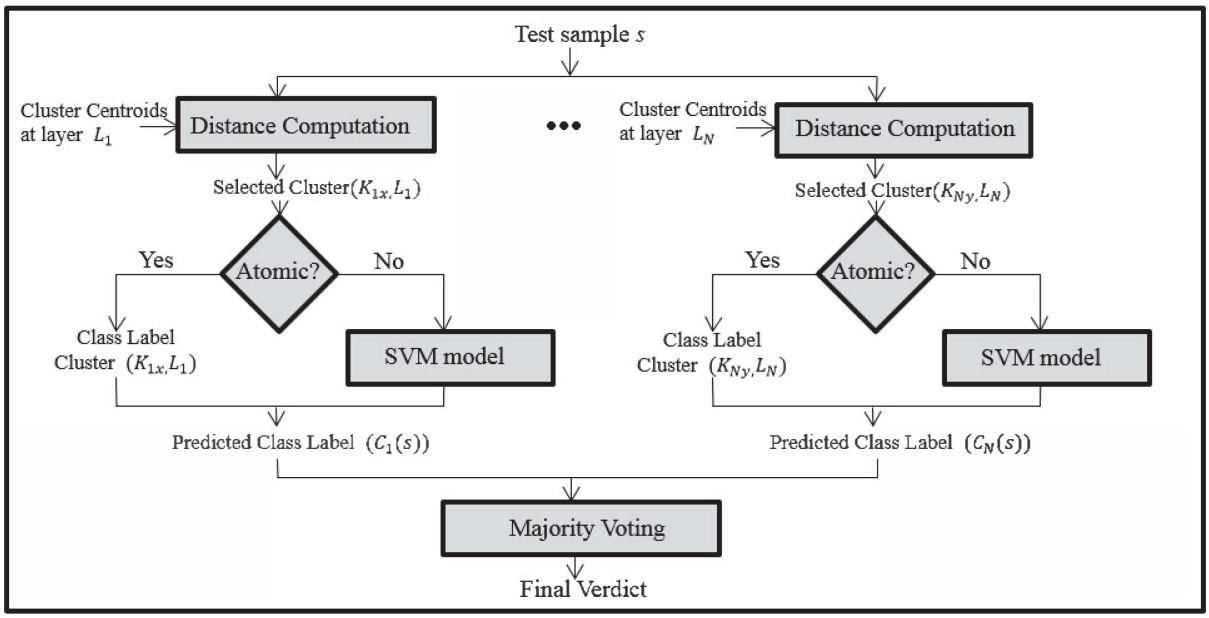

Fig. 5: Prediction for test sample set

NN (K=1), Support Vector Machine (SVM), Decision Tree and Naive Bayes classifier. Based on the real and large data collection (50,000 file samples) and using 10-fold cross validation, the experiment results shown in Table I and Fig. 6 demonstrate that Support Vector Machine (SVM) perform best compared with other popular classification methods in malware detection. Therefore, in the sections below, we use SVM for base classifier training.

Methods	TP	TN	FP	FN	Accuracy
K-NN	23433	23529	1471	1567	93.924%
SVM	23525	23554	1446	1475	94.158%
Decision Tree	23779	22377	2623	1221	92.312%
Naive Bayes	18783	13807	11139	6217	65.180%

TABLE I: Comparisons of different classification methods

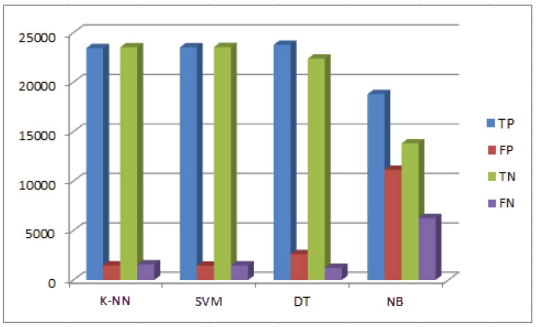

Fig. 6: Comparisons of different classification methods

C. Evaluation of Base Cluster-Oriented Classifier

In order to further analyze and visualize how cluster-oriented classifier performs based on our sample collection.

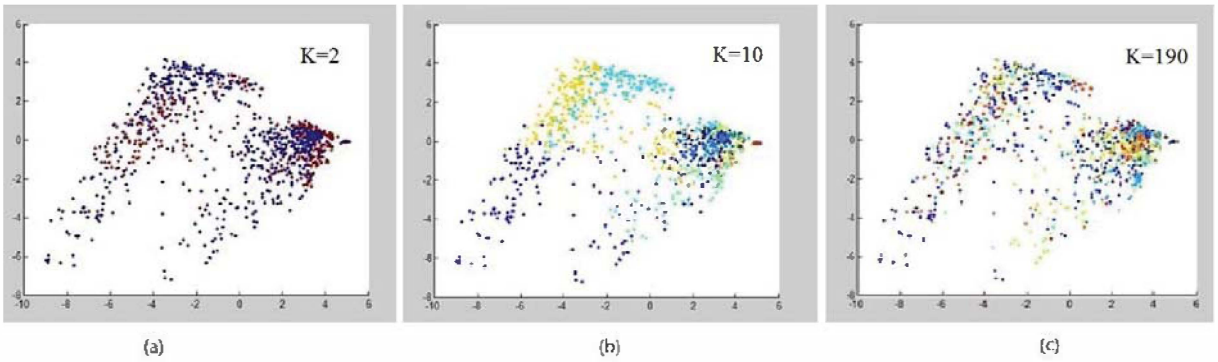

Fig. 7: File distributions by using PCA to reduce the dimensions

We randomly select 1,000 file samples from the whole data set (half of which are benign and half of which are malicious), and then use Principal Component Analysis (PCA) to reduce the feature space into two-dimension space for visualization. As shown in Fig 7, we can see that, there are many atomic clusters after clustering with proper Ks, each of which is with the same class label.

To further evaluate the performance of the base cluster-oriented classifiers, we use 10-fold cross validation. Based on 50,000 file collection, we use different cluster validity indices to obtain the possible Ks for the base classifier construction: F1-Measure (K=20500), Macro-F1 (K=21500), Micro-F1 (K=14500) and an improved *Fukuyama-Sugeno* index(NFS) (K=16500). The results in Table II show that using the labels of atomic clusters for prediction can achieve high detection accuracy, but the recall is not as high as expected. The experimental results shown in Table III demonstrate that: (1) by integrating clustering and classification, the detection recall is improved compared with using atomic clusters only; (2) each of our constructed base cluster-oriented classifier perform better than the single Support Vector Machine (SVM) classifier and the base cluster-oriented classifier using the method in [19] on malware detection.

K	Recall	ACY of atomic clusters
14500	82.562%	97.119%
16500	83.646%	97.243%
20500	89.000%	96.411%
21500	90.042%	96.293%

TABLE II: Detection performance using atomic clusters

D. Cluster-Oriented Ensemble Classifiers for Malware Detection

In this experiment, we compare the cluster-oriented ensemble classifiers with individual base classifiers for malware detection. The experimental results in Table IV and Figure 8 show that the cluster-oriented ensemble classifiers outperform each individual base classifier for malware detection.

VI. Conclusion and Future Work

In this paper, resting on the analysis of Windows Application Programming Interface (API) calls extracted from the file

Methods	TP	TN	FP	FN	Accuracy
SVM	23525	23554	1446	1475	94.158%
The base cluster-oriented classifier construction method in [19]					
14500	23571	23902	1008	1429	94.946%
16500	23783	23797	1203	1217	95.160%
20500	23642	23837	1163	1358	94.958%
21500	23585	23905	1095	1415	94.980%
Our base cluster-oriented classifier construction method					
14500	23692	24005	995	1308	95.394%
16500	23930	23900	1100	1070	95.660%
20500	23718	23983	1017	1282	95.402%
21500	23683	24009	991	1317	95.384%

TABLE III: Comparisons of different base cluster-oriented classifiers

K	TP	TN	FP	FN	Accuracy
14500	23692	24005	995	1308	95.394%
16500	23930	23900	1100	1070	95.660%
20500	23718	23983	1017	1282	95.402%
21500	23683	24009	991	1317	95.384%
Ensemble	24086	24054	928	932	96.280%

TABLE IV: Comparisons of cluster-oriented ensemble classifiers and individual base classifiers for malware detection

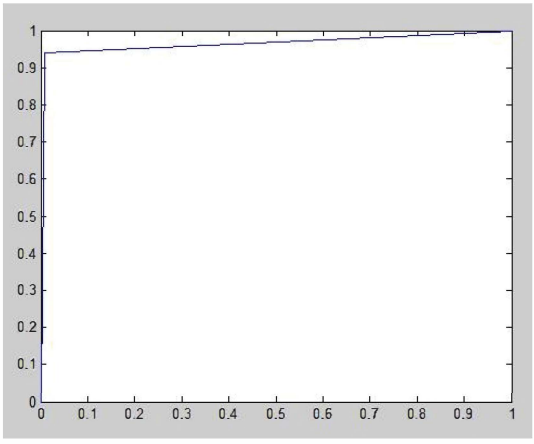

Fig. 8: ROC curve of cluster-oriented ensemble classifiers on malware detection

samples, we develop the intelligent malware detection system using cluster-oriented ensemble classifiers. To the best of our knowledge, this is the first work of applying such method for malware detection. A comprehensive experimental study on a real and large data collection from Comodo Cloud Security Center is performed to compare various malware detection approaches. Promising experimental results demonstrate that the accuracy and efficiency of our proposed method outperform other alternate data mining based detection techniques. In our future work, we will further design and implement our proposed methods in MapReduce framework for distributed computing to further improve the training efficiency.

ACKNOWLEDGMENT

The authors would also like to thank the anti-malware experts of Comodo Security Lab for their helpful discussions and suggestions. The work of Lifei Chen is partially supported by China NSF-61175123.

REFERENCES

[1] M. Bailey, J. Oberheide, J. Andersen, Z. M.Mao, F. ahanian, and J. Nazario. Automated classification and analysis of internet malware. RAID 2007, LNCS, 4637:178-197, 2007.

[2] U. Bayer, A. Moser, C. Kruegel, and E. Kirda. Dynamic analysis of malicious code. J Comput Virol, 2:67-77, May 2006.

[3] P. Beaucamps and E. Filiol. Metamorphism, formal grammars and undecidable code mutation. In Journal in Computer Science, 2007.

[4] P. Beaucamps and E. Filiol. On the possibility of practically obfuscating programs towards aunified perspective of code protection. In Journal in Computer Virology, 2007.

[5] M. Christodorescu, S. Jha, and C.Kruegel. Mining specifications of malicious behavior. In Proceedings of ESEC/FSE07, pages 5-14, 2007.

[6] E. Filiol.: Malware pattern scanning schemes secure against black box analysis. J. Comput. Virol. 2006.

[7] E. Filiol, G. Jacob, M. L.Liard.: Evaluation methodology and theoretical model for antiviral behavioural detection strategies. Journal in Computer Virology, 2007.

[8] M. Gheorghescu. An automated virus classification system. Virus Bulletin Conference, 2005.

[9] I. Gurrutxaga, O. Arbelaitz, J. M. Perez, J. Muguerza, J. I. Martin, I. Perona. Evaluation of malware clustering based on its dynamic behaviour. Seventh Australasian Data Mining Conference, 2008.

[10] X. Hu. Large-Scale Malware Analysis, Detection, and Signature Generation. Ph.D. Dissertation Thesis, 2011.

[11] X. Jiang and X. Zhu. vEye: behavioral footprinting for self-propagating worm detection and profiling. Knowledge and Information System,2009.

[12] J. Kephart, W. Arnold. Automatic extraction of computer virus signatures. Proceedings of 4th Virus Bulletin International Conference,1994.

[13] J. Kolter and M. Maloof. Learning to detect malicious executables in the wild. SIGKDD, 2004.

[14] T. Lee, J. J.Mody. Behavioral classification. EICAR, 2006.

[15] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. Proceedings of KDD, 1998.

[16] S. Niwattanakul, J. Singthongchai, E. Naenudorn, S. Wanapu. Using of Jaccard Coefficient for Keywords Similarity. Proceedings of the International MultiConference of Engineers and Computer Scientists, 2013.

[17] A. Rahman, B. Verma. A Novel Ensemble Classifier Approach using Weak Classifier Learning on Overlapping Clusters. Neural Networks (IJCNN), 2010.

[18] A. Rahman, B. Verma. Cluster-Oriented Ensemble Classifier: Impact of Multicluster Characterization on Ensemble Classifier Learning. IEEE Transactions on Knowledge and Data Engineering, 2012.

[19] A. Rahman, B. Verma. Novel Layered Clustering-Based Approach for Generating Ensemble of Classifiers. IEEE Transactions on Neural Networks, 2011.

[20] A. Rahman, B. Verma. Cluster-based ensemble of classifiers. Expert Systems, 2013.

[21] M. Schultz, E. Eskin, and E. Zadok. Data mining methods for detection of new malicious executables. In Proccedings of 2001 IEEE Symposium on Security and Privacy, 2001.

[22] A. Sung, J. Xu, P. Chavez, and S. Mukkamala. Static analyzer of vicious executables (save). In Proceedings of the 20th Annual Computer Security Applications Conference, 2004.

[23] G.J. Tesauro, J.O. Kephart, G.B. Sorkin. Neural networks for computer virus recognition. IEEE Expert, 1996.

[24] R. Tian, L.M. Batten, and S.C. Versteeg. Function length as a tool for malware classification. 3rd International Conference on Malicious and Unwanted Software (MALWARE), 2008.

[25] J. Wang, P. Deng, Y. Fan, L. Jaw, and Y. Liu. Virus detection using data mining techniques. Proccedings of ICDM03, 2003.

[26] Y. Wang, Y. Ye, H. Chen, Q. Jiang. An Improved Clustering Validity Index for Determining the Number of Malware Clusters, International Conference on Anti-counterfeiting, Security, and Identification (ASID), 2009.

[27] R. Xu, D. Wunsch, Survey of Clustering Algorithms. IEEE Transactions on Neural Networks, 2005.

[28] Y. Ye, T. Li, S. Zhu, W. Zhuang, E. Tas, U. Gupta, M. Abdulhayoglu. Combining File Content and File Relations for Cloud Based Malware Detection, Proceedings of ACM International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD), 2011.

[29] Y. Ye, D. Wang, T. Li, and D. Ye. IMDS: Intelligent malware detection system. SIGKDD, 2007.

[30] Y. Ye, T. Li, Y. Chen, Q. Jiang. Automatic Malware Categorization Using Cluster Ensemble. Proceedings of ACM International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD), 2010.

[31] Y. Ye, T. Li, Q. Jiang, Z. Han, L. Wan. Intelligent File Scoring System for Malware Detection from the Gray List. Proceedings of ACM International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD), 2009.

[32] Y. Ye, T. Li, Q. Jiang, Y. Wang. CIMDS: Adapting post-processing techniques of associative classification for malware detection system. IEEE Transactions on Systems, Man and Cybernetics-Part C: Applications and Reviews, 2010.

[33] Y. Ye, L. Chen, D. Wang, T. Li, Q. Jiang, M. Zhao. SBMDS: an interpretable string based malware detection system using SVM ensemble with bagging. Journal in Computer Virology, 2009.

[34] Y. Ye, T. Li, K. Huang, Qi. Jiang, Y. Chen. Hierarchical Associative Classifier (HAC) for Malware Detection from the Large and Imbalanced Gray List, Journal of Intelligent Information Systems, 2009.

[35] Y. Ye, D. Wang, T. Li, D. Ye, Q. Jiang. "An Intelligent PE-Malware Detection System Based on Association Mining", Journal in Computer Virology, 2008.

[36] Y. Ye. Research on intelligent malware detection methods and their applications. Ph.D. Dissertation Thesis, 2010.