pISSN 1976-913X
eISSN 2092-805X

J Inf Process Syst, Vol.9, No.3, September 2013 http://dx.doi.org/10.3745/JIPS.2013.9.3.395

Classifying Malicious Web Pages by Using an Adaptive Support Vector Machine

Young Sup Hwang*, Jin Baek Kwon*, Jae Chan Moon**
and Seong Je Cho**

Abstract—In order to classify a web page as being benign or malicious, we designed 14 basic and 16 extended features. The basic features that we implemented were selected to represent the essential characteristics of a web page. The system heuristically combines two basic features into one extended feature in order to effectively distinguish benign and malicious pages. The support vector machine can be trained to successfully classify pages by using these features. Because more and more malicious web pages are appearing, and they change so rapidly, classifiers that are trained by old data may misclassify some new pages. To overcome this problem, we selected an adaptive support vector machine (aSVM) as a classifier. The aSVM can learn training data and can quickly learn additional training data based on the support vectors it obtained during its previous learning session. Experimental results verified that the aSVM can classify malicious web pages adaptively.

Keywords— adaptive classification, malicious web pages, support vector machine

1. INTRODUCTION

A malicious web page can slow down a user's system by consuming computing resources. It can also leak personal information or make systems malfunction if a user visits the page and involuntarily downloads malware. To protect systems from malicious web pages, it is important for users to be able to know when they are browsing whether a web page is benign or malicious.

Most malicious web pages avoid detection by anti-virus software (AV) through obfuscation or polymorphism. Obfuscation methods can be broadly grouped into the following four types: those using ASCII values and Unicode values; those using XOR operation; those splitting a string; and those compressing a string and replacing the existing string with a meaningless string [1]. The obfuscation method makes it difficult for the AV to identify a malicious web page. However, malicious web pages tend to have common computable features such as a long string and frequent use of special characters. We investigated these features by analyzing 3,050 web page samples that had been collected by the Korea Internet and Security Agency (KISA). Based on our analysis, we designed 14 basic features and 16 extended features. These features can be used to train a classifier such as a support vector machine (SVM) or a multi-layer perceptron.

Because more malicious web pages are appearing, and because they change so rapidly, classi-

* This work was supported by the Sun Moon University Research Grant of 2011-0005.
Manuscript received May 11, 2012; accepted August 2, 2012.
Corresponding Author: Young Sup Hwang
* Dept. of Computer Science and Engineering, Sun Moon University, Korea ({young, jbkwon}@sunmoon.ac.kr)
** Dept. of Computer Science and Software Science, Dankook University, Korea ({answocks, sjcho}@dankook.ac.kr)

fiers that are trained by old data can misclassify some new pages. To overcome this problem, we used an adaptive support vector machine (aSVM) as a classifier [2, 3]. The aSVM can learn training data and can quickly learn additional training data based on the support vectors it obtained during its previous learning session. This adaptive capability makes it easy for the aSVM to deal with new training data.

To verify the proposed approach, we obtained an additional 441 malicious web page samples from KISA. We extracted the 16 feature sets from all of the malicious and benign web pages. We divided the feature data into training and test data. We trained the aSVM using the first training data and tested it, and then retrained the aSVM using the additional training data and tested it. The experimental results revealed that the aSVM can classify malicious web pages adaptively.

This paper is organized as follows: Section 2 summarizes related bodies of work; Section 3 describes the basic and extended features we designed; Section 4 describes the aSVM used in our approach; Section 5 presents the experimental results; and Section 6 offers conclusions and directions for future research.

2. RELATED WORK

Choi, Kim, and Choi analyzed malicious Java scripts and developed the following three metrics for detecting obfuscated strings: n-gram, entropy, and word size [4]. They assumed that eval() and document.write() are dangerous functions. However, according to Feinstein and Peck, benign web pages frequently use document.write() and eval() [5]. We designed not only simple metrics but also extended features by combining the basic features to improve the classification.

Kim et al. developed a method for detecting suspicious malicious web sites by using the strength analysis of a JavaScript obfuscation [6]. They used features such as a string that is separated by a special character (<,>,“,‘,[,],{,}, etc.) and that is longer than 200 characters; the frequency of certain functions like eval(), replace(), fromCharCode(), document.write(), document.create.Element(); the entropy of characters in JavaScript; and special character entropy, etc.

Likarish et al. studied obfuscated malicious Javascript detection using N-gram related features [1]. Among other things, these features included: the length of a string, the average length of a string per line, the average number of lines of a script, the number of strings, the number of Unicode symbols, and the number of calls of a method. Kim et al. and Likarish et al. used string length as a feature. We used the same two features, which are the number of strings and the number of calls of a function.

Seifert et al. used a heuristic method to classify malicious web pages [7]. They used a decision tree as a classifier. Their experiments revealed a false positive (FP) rate of 5.88% and a false negative (FN) rate of 46.15%. The simple decision tree and the use of eight features, which are insufficient, probably caused the high false negative rate. We tried to develop sufficient features and to improve the classification rate by using SVM, which is known to have a good generalization capability.

Yunsg-Tsung Hou et al. studied machine-learning methods including a decision tree, as well as naïve Bayes, SVM, and a boosted decision tree to classify malicious web pages [8]. They used 154 counts of a native JavaScript function, 9 HTML document levels, and 8 counts of ActiveX objects. Their experiments revealed that the boosted decision tree had low FP and high

classification accuracy. They trained the classifiers using 176 malicious samples, and did 10 fold cross-validation on the data sets. The high accuracy of their boosted decision tree may be related to their reuse of training data during testing. We used sufficient training data and tested the classifier using new testing data.

3. FEATURE DESIGN

3.1 Basic Features

To differentiate between benign and malicious web pages, we designed 14 basic features. These were selected to represent the essential characteristics of a web page, based on our analysis of 2,604 benign web pages and 444 malicious web pages that were collected by KISA [9]. We selected some features as in other papers and researches [1, 5, 6, 8] and designed some features by ourselves. The 14 basic features are as follows:

- SoF (Size of File): The size of a file does not differ statistically between malicious and benign web pages.
- NANC (Number of Non-Alphanumeric Characters): Non-alphanumeric characters, (i.e., special characters) appear more frequently in malicious web pages than in benign web pages.
- HT (Number of HTML Tags): The ratio of the number of HTML tags and the number of keywords in JavaScript differs greatly between malicious and benign web pages.
- CoS (Number of Operators to Concatenate Strings): Malicious web pages use many operators to concatenate strings.
- UDF (Number of User-Defined Functions): This feature detects encoding by using a user-defined function, rather than detecting encoding by using the JavaScript encoding function.
- JSKEY (Number of JavaScript Keywords): This feature reveals how many JavaScript keywords are used in a web page.
- LoS (Length of Script): Malicious web pages generally have a lengthy amount of JavaScript.
- Func (Number of Functions): Malicious web pages use many functions to perform malicious actions.
- Var (Number of Variables): Malicious web pages use many variables to perform malicious actions. This feature helps detect string concatenations by using variables.
- Oper (Number of Operators): Malicious web pages have more operators than benign web pages.
- Const (Number of Constants): Malicious web pages have many constants from many loops.
- NoL (Number of Literals): Malicious web pages have many literals. This feature helps detect obfuscation.
- TLoL (Total Length of Literals): This feature helps detect obfuscation.
- DeFunc (Number of Decoding Functions): Malicious web pages use decoding functions to decode obfuscated scripts.

CoS, NoL, and TLoL are calculated as follows:

```
Var String = “aaa”+“bbbb”+“cccc”;
// CoS = 2
// NoL = 3
// TLOL = 11
```

Table 1. Basic features

Features	Description
SoF	Size of File
NANC	Number of Non-Alphanumeric Characters
HT	Number of HTML Tags
CoS	Number of Operators to Concatenate Strings
UDF	Number of User Defined Functions
JSKey	Number of JavaScript Keywords
LoS	Length of Script
Func	Number of Functions
Var	Number of Variables
Oper	Number of Operators
Const	Number of Constants
NoL	Number of Literals
TLoL	Total Length of Literals
DeFunc	Number of Decoding Functions

Table 1 summarizes the 14 basic features to classify malicious web pages.

3.2 Extended Features

The numerical ranges of each basic feature vary significantly. For example, the length of a script may be more than 5,000, but the number of functions may be within hundreds. For this reason, we combined two basic features to enable relative comparison of extended features. These extended features are also more helpful to classifiers.

We analyzed 2,604 benign and 444 malicious web pages using the 16 extended features. Ta-

Table 2. Extended features

Number	Extended Features
1	LoS/SoF
2	NANC/LoS
3	CoS/Oper
4	Func/Var
5	TLoL/CoS
6	TLoL/NoL
7	TLoL/LoS
8	Const/Var
9	Const/Oper
10	JSKey/HT
11	DeFunc/Func
12	Var/UDF
13	Func/LoS
14	Const/LoS
15	Var/LoS
16	NANC/TLoL

Table 3. The analysis results of web pages

	LoS/SoF	NANC/LoS	CoS/Oper	Func/Var
Benign	0.130	0.133	0.094	0.398
Malicious	0.832	0.174	0.202	45.400
	TLoL/CoS	TLoL/NoL	TLoL/LoS	Const/Var
Benign	71.372	29.036	0.138	0.830
Malicious	771.745	254.718	0.520	2.405
	Const/Oper	JSKey/HT	DeFunc/Func	Var/UDF
Benign	0.051	0.519	0.044	2.552
Malicious	0.154	8.037	0.386	10.408
	Func/LoS	Const/LoS	Var/LoS	NANC/TLoL
Benign	0.004	0.002	0.006	0.137
Malicious	0.009	0.009	0.007	0.139

ble 3 and Figure 1 present the results of our analyses.

Figure 1 shows the averaged relative ratios for malicious (white bars) and benign (black bars) web pages. For example, the average NANC/LoS value is 0.133 for benign web pages and 0.174 for malicious web pages, so the white and black bars have similar lengths. However, large differences appear for LoS/SoF values. Analyses revealed that 4 of the features (NANC/LoS, Func/LoS, Var/LoS, and NANC/TLoL) did not differ greatly, but because they are dependent on other features, we can utilize them if we use a machine learning method. We selected the SVM as a classifier because it is known to have a good generalization capability.

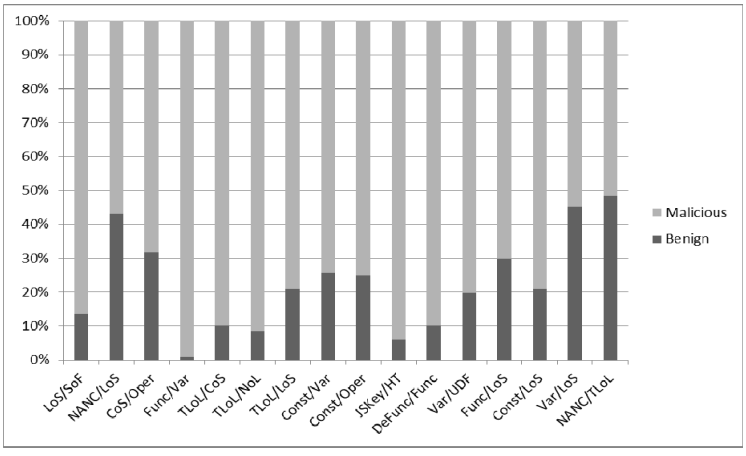


Fig. 1. The analysis results of web pages

4. ADAPTIVE SUPPORT VECTOR MACHINE

The main idea behind the SVM is, "Given a training sample, the support vector machine constructs a hyper plane as the decision surface in such a way that the margin of separation between

positive and negative examples is maximized" [10]. By applying the kernel method, the SVM can handle non-linearly separable patterns. The support vectors are those data points that lie closet to the optimal hyper plane. These support vectors play a central role in classification. The SVM is the most widely used kernel-learning algorithm and is a state-of-the-art method by virtue of its good generalization performance, relative ease of use, and rigorous theoretical foundations [10].

Because malicious web pages are increasingly being generated, a standard SVM that has been trained using current samples may not handle new samples correctly. It needs to be re-trained using all previous samples plus these new samples. The increased sample size makes the training go slower, and if all new samples have characteristics that differ from current samples, it may be necessary to change the hyper plane adaptively.

Yang proposed the aSVM, which is a generic framework for function-level classifier adaptation based on regularized loss minimization [3]. The aSVM directly modifies the decision function of a source classifier (of any type) that has been trained from a source domain into a target classifier for a target domain. The adaptation only requires limited labeled examples from the target domain, and no raw data from the source domain, making this framework very efficient and flexible [11]. The implementation of the aSVM is based on the LIBSVM package [2].

The usage of the aSVM is similar to the LIBSVM package, except that the aSVM can use a source model file. The source model file is an output model file from the aSVM or a standard SVM. The aSVM can efficiently and adaptively learn new data from this file. The aSVM is reportedly 13–15 times faster than the standard SVM when training adaptively with new data [2]. The aSVM package can be obtained from http://www. cs.cmu.edu/~juny/AdaptSVM/index.html.

5. EXPERIMENTAL RESULTS

5.1 First Training

We extracted the 16 extended features from 3,050 web pages that were obtained from KISA. After removing illegal data that were not appropriate for the aSVM, we obtained 1,458 benign and 404 malicious samples. From these, we selected 200 benign and 200 malicious samples as training data. The remaining 1,258 benign and 204 malicious data were used to test the classifier.

Table 3 shows that the average NANC/LoS value was 0.174, but the average TLoL/CoS value was 771.745 for malicious web pages. Because greater values can dominate over smaller values when training the aSVM, we normalized all data to the range [0, 1].

After the first training, the aSVM successfully classified the training data in 98.5% (394/400) of the cases and the test data in 93.7% (1,370/1,462). The support vectors (SV) numbered 77 and the number of iterations was 1,022. The FP rate for the test data was 6.9%. Table 1 lists detailed information about the experimental results for the test data.

Table 5 compares the experimental results with those of previous experiments.

Because the goals and the data used differed for each experiment, the comparison is not objective. Still, certain trends are evident. In [8], when the FP rate is 0.2%, the FN rate is 14.8%. Our method is comparable with the FP rate. Tables 4 and 5 show that the 16 extended features proposed for use with the aSVM can be used to classify malicious web pages.

Table 4. The experimental results for the test data

	The amount of data		Percentage	
	Benign	Malicious	Benign	Malicious
Benign (1,258)	1,180	78	93.8	6.2
Malicious (204)	19	140	6.9	93.1

Table 5. Comparisons with other experimental results

	FN	FP
Proposed Method	6.2%	6.9%
J. Lee[9]	3.9%	14.4%
B. Kim[12]	3.8%	12.1%
Peter Likarish[1]		10.5%
Yung-Tsung Hou[8]	7.4–14.8%	0.2–7.6%

5.2 Adaptive Training

We obtained 441 more malicious website samples from KISA, from the year after the previous dataset was obtained. We divided this new data into five groups. Four of the groups included 100 new malicious samples and 100 benign samples from the previous dataset and the final group included only 41 new malicious samples and 42 benign samples.

The adaptive training using the first new group, based on the SVs from the first training, was completed quickly. The SVs numbered 76 and the number of iterations was 696. The classification rates were 100% for all groups, with the exception of group 4 (99%). The classification rate for the previous dataset was 94.1%.

The adaptive training using the second new group, based on the SVs from the previous training, was completed even more rapidly. The SVs numbered 60 and the number of iterations was 235. The classification rates were 100% for all groups, with the exception of group 4 (99%). The classification rate for the previous dataset was 94.1%. To maintain the previous learning, the SVs were decreased only minimally. The number of iterations decreased considerably because few new data were involved (200).

Table 6 presents the results of the adaptive training. 'Group1' refers to the results after adaptive training uses 'Group1' data. 'Accuracy' refers to the accuracy of the test for previous test data, which are Group1, Group2, Group3, and Group4, respectively. Because all accuracy results were the same after the adaptive training of each group, we presented them in one row. The last row presents the accuracy for the previous test data (see Table 4) after each adaptive training session.

Figure 2 shows that the number of SVs decreased as the adaptive training continued. The

Table 6. Results of the adaptive training.

	First	Group1	Group2	Group3	Group4
nSV	77	76	60	67	43
#iter	1,022	696	235	249	361
accuracy	93.7	100	100	100	99
test	93.7	94.1	94.1	94.0	94.3

small number of SVs means that the aSVM can classify more rapidily.

Figure 3 shows that the number of iterations for training also decreased. This finding means that training happened faster. Figure 4 presents the accuracy for the test data that was used in the first training. The accuracy changed minimally, indicating that the new data shared similar characteristics with the previous training data.

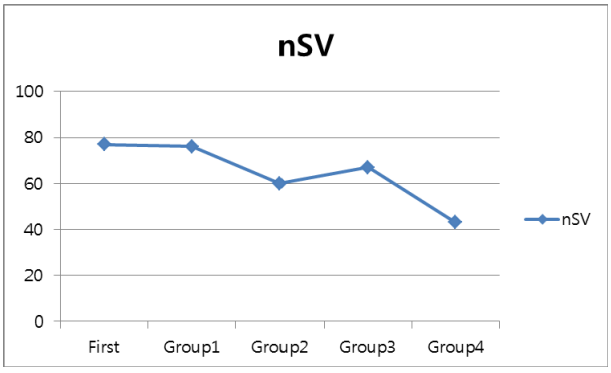


Fig. 2. Changes in the number of SVs

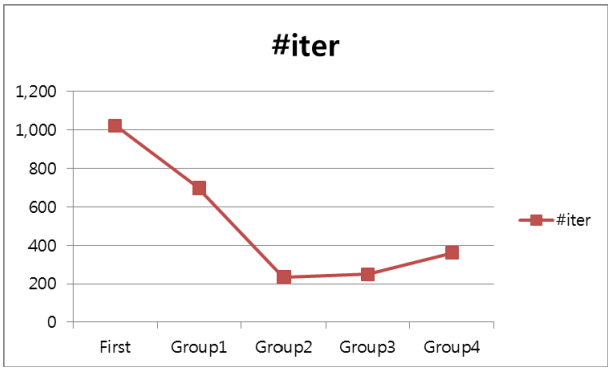


Fig. 3. Changes in the number of iterations

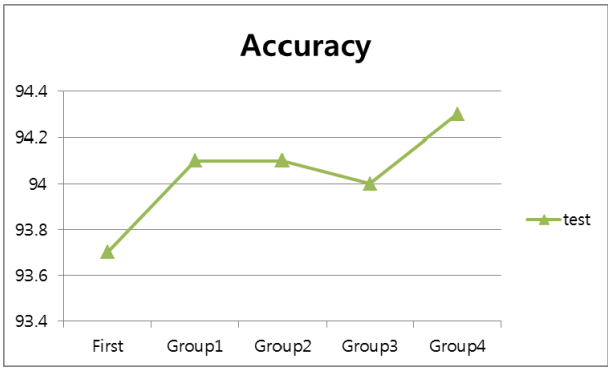


Fig. 4. Changes in accuracy for the test data

For comparison, we trained the standard SVM using the previous training data plus the first two new groups. The SVs numbered 220 and the number of iterations was 2,781. The classification rate for the previous test data was 96.7% and the classification rate for the remaining data was 96.8%. As noted in Section 4.1, the original accuracy was 93.7%, so accuracy was improved by 3%. The number of SVs increased from 77 to 220, and the number of iterations increased from 1,022 to 2,781.

The experimental results demonstrated that the aSVM can learn new data quickly and adaptively, and that it can preserve the classification capability from previous learning. We also found that aggregated training makes the standard SVM more powerful than the aSVM, but this required more time and redundant processing.

5. CONCLUSION

We designed 14 basic and 16 extended features to classify malicious web pages. We trained the aSVM using these features and trained the aSVM adaptively, adding new data that was based on the previous learning status. The experimental results revealed that the features can help classify malicious web pages and that the aSVM can adaptively learn new data.

In future research, when new malicious web pages are detected, we will work to train the aSVM adaptively using new data. If the new data has different characteristics, the number of SVs will increase to represent the new characteristics. In our experiments, the number of SVs decreased, meaning that the new 441 samples were similar to the first training data. Our next experiments will be conducted using new data with different characteristics.

Additionally, the proposed method requires a 'teacher' – a supervisor to detect whether or not a web page is malicious. This teacher may be a dynamic malware detector or a human expert. Another area for future research will be improving our method of classifying unlabeled data and learning it adaptively.

REFERENCES

[1] Peter Likarish, E. Jung, I. Jo, "*Obfuscated Malicious JavaScript Detection using Classification Techniques*," *Proceedings of the 4th International Conference on Malicious and Unwanted Software*, pp.47–54, 2009.

[2] Chih-Chung Chang and Chih-Jen Lin, *LIBSVM: A Library for Support Vector Machines*, 2001. Software available at http://www.csie.ntu.edu.tw /~cjlin/libsvm.

[3] Jun Yang, Rong Yan and Alex Hauptmann, "Cross-Domain Video Concept Detection using Adaptive SVMs," *ACM Multimedia*, pp.188–197, 2007.

[4] Y. Choi, T. Kim, and S. Choi, "Automatic Detection for JavaScript Obfuscation Attacks in Web Pages through String Pattern Analysis," *International Journal of Security and Its Applications*, Vol.4, No.2, pp.13–26, Apr. 2010.

[5] B. Feinstein and D. Peck, "*Caffeine Monkey: Automated Collection, Detection and Analysis of Malicious JavaScript*," *Black Hat USA*, 2007.

[6] B. Kim, C. Im, H. Jung, "Suspicious Malicious Web Site Detection with Strength Analysis of a JavaScript Obfuscation," *International Journal of Advanced Science and Technology*, vo.26, Jan, 2011.

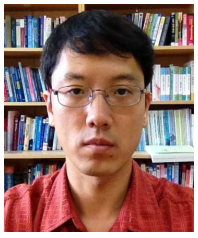
[7] Christian Seifert, Ian Welch, Peter Komisarczuk, "*Identification of Malicious Web Pages with Static Heuristics*," *Telecommunication Networks and Applications Conference*, pp.91–96, Dec. 2008.

[8] Y.-T. Hou, Y. Chang, T. Chen, C.-S. Laih and C.-M. Chen, "Malicious Web Content Detection by Machine Learning," *Expert Systems with Applications*, Vol.378, pp.55–60, 2010.

[9] J. Lee, J. Moon, S. Cho, Y. Lee, M. Park and W. Choi, "*Malicious Web Page Detection using Malicious Code Spreading Pattern*," *The 3rd International Conference on Internet (ICONI)*, pp.195–200, Dec. 2011.
[10] Simon Haykin, *Neural Networks and Learning Machines*, Chapter 6, Prentice Hall, 2008.
[11] Jun Yang and Alex Hauptmann, "*A Framework for Classifier Adaptation and its Applications in Concept Detection*," *ACM Int'l Conf. on Multimedia Information Retrieval* (MIR), Vancouver, Canada, 2008.
[12] B. Kim, C. Im, and H. Jung, "Suspicious Malicious Web Site Detection with Strength Analysis of a JavaScript Obfuscation," *International Journal of Advanced Science and Technology*, Vol.26, pp.19–32, Jan. 2011.

Young Sup Hwang

He received the B.E. in Computer Engineering from Seoul National Univ. in 1989, the M.E. and the Ph.D. in Computer Engineering from POSTECH in 1991, 1997 respectively. During 1997-2002, he was a senior researcher in ETRI. He is an associate professor in the department of computer science and engineering, Sun Moon University, Korea. His research interests include pattern recognition, neural networks, bio-informatics, machine learning and software security.



Jin Baek Kwon

He is an associate professor in the department of computer science and engineering, Sun Moon University, Republic of Korea. He has received a B.S. in statistics from Hankuk University of Foreign Studies in 1998, an M.S. and a Ph.D. in computer science from Seoul National University in 2000 and in 2003, respectively. His research interests include distributed systems, multimedia systems and operating systems.

Jae Chan Moon

He received the B.S. degree in Computer Science and the M.E. degree in Computer Engineering from Dankook University in 2011 and 2012 respectively. During 2009~2012, he was a member of Secure Software & System Lab. His research interests include computer security, internet web security, smartphone security, and software protection, licensing and copy protection.

Seong Je Cho

He received the B.E., the M.E. and the Ph.D. in Computer Engineering from Seoul National University in 1989, 1991 and 1996 respectively. He was a visiting scholar at Department of EECS, University of California, Irvine, USA in 2001, and at Department of Electrical and Computer Engineering, University of Cincinnati, USA in 2009 respectively. He is a Professor in Department of Computer Science and Software Science, Dankook University, Korea from 1997. His current research interests include computer security, operating systems, software protection, real-time scheduling, and embedded software.