Improving the Detection of Unknown Computer Worms Activity using Active Learning

Robert Moskovitch, Nir Nissim, Dima Stopel, Clint Feher, Roman Englert, and Yuval Elovici
Deutsche Telekom Laboratories at Ben-Gurion University, Be'er Sheva, 84105 Israel.
{robertmo, nirni, stopel, clint,englert,elovici@bgu.ac.il

Abstract. Detecting unknown worms is a challenging task. Extant solutions, such as anti-virus tools, rely mainly on prior explicit knowledge of specific worm signatures. As a result, after the appearance of a new worm on the Web there is a significant delay until an update carrying the worm's signature is distributed to anti-virus tools. We propose an innovative technique for detecting the presence of an *unknown* worm, based on the computer operating system measurements. We monitored 323 computer features and reduced them to 20 features through feature selection. Support vector machines were applied using 3 kernel functions. In addition we used active learning as a selective sampling method to increase the performance of the classifier, exceeding above 90% mean accuracy, and for specific unknown worms 94% accuracy.

Keywords: Classification, Active Learning, Support Vector Machines, Malcode Detection.

1 Introduction

The detection of malicious code (malcode) transmitted over computer networks have been researched intensively in recent years. One type of abundant malcode is *worms*, which proactively propagate across networks while exploiting vulnerabilities in operating systems or in installed programs. Nowadays, excellent technology (i.e., antivirus software packages) exists for detecting *known* malicious code, typically, through detection of known signatures. Nevertheless, it is based on prior explicit knowledge of malcode signatures rendering helpless against unknown malcode. This solution has obvious demerits, however, since worms propagate very rapidly. [1].
Intrusion detection, commonly done at the network level, a more local technique is *Host-based Intrusion Detection* (*HIDs*). Our suggested approach can be classified as *HIDs*, but the novelty here is that it is based on the computer behavior, reflected by the operating system measurements, in which the knowledge is acquired automatically based on a set of known worms. In a previous study we performed this task using several classification algorithms [2]. In this study we employ Support Vector Machines (SVM), which are known in their outperforming capabilities in binary classification tasks. Active Learning is commonly used to reduce the amount of labeling required from an expert – often time consuming and costly. However, in this study, in which all the examples are labeled, we are using the Active Learning approach, as a selective sampling method to improve the classification accuracy.

3. Methods

3.1. Support Vector Machines and Feature Selection

SVM is a binary classifier, which finds a linear hyperplane that separates the given examples of two classes, known to handle large amount of features. Given a training set of labeled examples in a vector format: $x_i = <f_1,f_2 \ldots f_m, y_i>$, where f_i' is a feature, and its label $y_i = \{-1,+1\}$. The SVM attempts to specify a linear hyperplane that has the maximal margin, defined by the maximal (perpendicular) distance between the examples of the two classes. The examples lying closest to the hyperplane are the "supporting vectors". The Normal vector of the hyperplane, *denoted as w* in formula 1,is a linear combination of the supporting vectors, multiplied by LaGrange multipliers (alphas). Often the dataset cannot be linearly separated, thus a kernel function K is used. The SVM actually projects the examples into a higher dimensional space to create a linear separation of the examples. We examined the 3 commonly used kernels: *Linear*, *Polynomial* and *RBF*. Generally, the SVM classifier is in the form shown in formula 1, in which n is the number of the training examples. We used the Lib-SVM implementation [3] which also provides a multiple classification.

$$f(x) = sign(w \cdot \Phi(x)) = sign\left(\sum_{1}^{n} \alpha_i y_i K(x_i\, x) \right) \tag{1}$$

To reduce the amount of features we employed three *filtering* feature selection measures: *Chi-Square (CS), Gain Ratio* [4, 5] *(GR)* and *ReliefF* [6]. After having the ranks for each feature, the top 5, 10, 20 and 30 features were selected, based on each measure and their ensemble.

3.2. Active Learning

Active Learning (A.L) is usually used to reduce the efforts in labeling examples, a commonly time consuming and costly task, while maintaining a high accuracy rate. In *A.L* the learner actively asks to label specific examples from a given pool, which expected to result in a better classifier. In our study all the examples are labeled, however, we employed this approach as a selective sampling method to increase the accuracy. We implemented a pool based active learner which aims to reduce the expected *ge*neralization error, named *Error-Reduction*, presented by [7], in which, an example is acquired from a pool only if it dramatically expected to improve the confidence of the current classifier over all the examples in the pool. Through the use of a log-loss function the error degree of acquiring a specific example, having a specific label, enables the self-estimation of the mean error. Finally, the example with the lowest self-estimated mean error is selected and added to the training set.

3.4. DataSet Creation

Since there is no benchmark dataset we created a dataset. A controlled network of computers was deployed, into which we could inject worms, monitor and log the computer features. Finally all the data was aggregated into a vector of 323 features for every second. Five available computer worms, representing a variety of worms, were selected: **(1)** ***W32.Dabber.A***, **(2)** ***W32.Deborm.Y***, **(4)** ***W32.Sasser.D***, **(5)** ***W32.Slackor.A***. All the worms perform port scanning and possess different characteristics. A detailed description of the dataset is in [2]. To examine the influence of the machine hardware or software and user activity on the accuracy, we performed the monitoring operation on eight combinations, resulting from three binary aspects: *old* and *new* computer configuration, background activity of software or absent, and User activity or absent. More details on these aspects in [2]. Thus, we had eight datasets including six class examples of the five worms and none activity.

3.5. Evaluation Measures

For the purpose of evaluation we used the *True Positive (TP)* measure presenting the rate of instances classified as *positive* correctly (true), *False Positive (FP)* presenting the rate of *positive* instances misclassified and the *Total Accuracy* – the rate of the entire *correctly* classified instances, either positive or negative, divided by the entire number of instances.

4. Experiments and Results

In the first part of the study, we wanted to identify the best feature selection measure, the best kernel function. In the second part we wanted to measure the possibility of classifying unknown worms using a training set of known worms, and the possibility to increase the performance using selective sampling.
We performed a wide set of experiments, called *e1*, in which we evaluated each kernel function, feature selection and top selection combination to determine the outperforming combination. We trained each classifier on a single dataset i and tested iteratively on the other eight datasets. Note, the classification task included five worms or none (worm) activity. The mean performance of the top 20 features (Gain Ratio) runs outperformed, which we used in the next experiments.

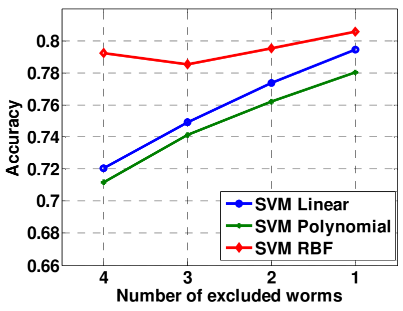

Fig 1. The performance monotonically increases as fewer worms are excluded (and more worms appear in the training set), RBF kernel has quite different accuracy trend that had one decreasing.

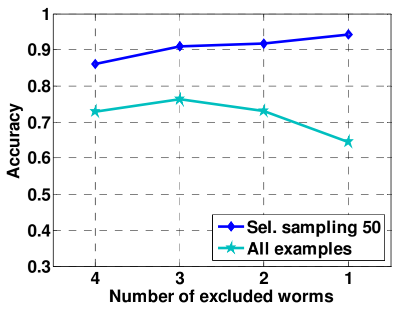

Fig 2. The considerable improvement when using selective sampling compare to training the SVM on all the examples. When 1 worm was excluded there was almost 30% improvement in accuracy.

4.1. Experiment II – Unknown Worms Detection

To evaluate the capability to classify an *unknown worm* activity, an experiment, called *e2*, was performed. In this experiment the training set consisted of *(5-k)* worms and the testing set contained the *k* excluded worms, while the *none* activity appeared in both datasets. This process repeated for all the possible combinations of the *k* worms, for *k* = *1* to *4*. In these experiments there were two classes: (generally) *worm,* for any type of worm*,* and *none* activity.
Figure 1 presents the results of *e2*. A monotonic improvement observed, as more worms included in the training set, in which the *RBF* outperformed. However, in specific worms, when a single worm was excluded, 95% accuracy observed.

4.2. Experiment 3 – Using Selective Sampling

To maximize the performance achieved by the RBF kernel, in *e3* we employed selective sampling methods, using error reduction criterion. Generally, we performed the same experiment as in *e2*, however, the training set included all the eight datasets and the test set only one of them. For the selective sampling process, initially six examples from each class (worms and none) were selected randomly, and then in each AL iteration additional single example selected. Figure 2 presents the results of *e3*. The selective sampling had improved significantly the baseline performance even when 50 samples were selected, as shown in figure 2.

5. Conclusions and Future Work

We presented the concept of detecting *unknown* computer worms based on a host behavior, using the SVM classification algorithm with several kernels. Using feature selection we shown that even with 20 features a high accuracy is achieved. Often the *RBF kernel* outperformed the other kernels. In the detection of unknown worms (*e2*), a high level of accuracy was achieved (exceeding 80% in average); as more worms were in the training set. To reduce the noise in the training set and improve the performance we employed the A.L approach as a selective sampling method which increased the accuracy after selecting 50 examples to above 90% of accuracy and 94% when the training set contained four worms. These results are highly encouraging and show that unknown worms can be detected in real time. The advantage of the suggested approach is the automatic acquisition and maintenance of knowledge, based on inductive learning. This avoids the need for a human expert who is not always available and familiar with the general rules. This is possible these days, based on the existing amount of known worms, as well as the generalization capabilities of classification algorithms. However, in order to detect more sophisticated worms, we develop a temporal classification method.

References

1. Craig Fosnock, Computer Worms: Past, Present and Future. East Carolina University (2005) Kabiri, P., Ghorbani, A.A. (2005) "Research on intrusion detection and response: A survey," International Journal of Network Security, vol. 1(2), pp. 84-102.
2. Robert Moskovitch, Ido Gus, Shay Pluderman, Dima Stopel, Yisrael Fermat, Yuval Shahar and Yuval Elovici, Host Based Intrusion Detection Using Machine Learning, In Proceedings of Intelligence and Security Informatics, Rutgers University, May 2007.
3. Chang, C.-C. and Lin, C.-J. (2001). LIBSVM: a library for support vector machines. Softwareavailable at http://www.csie.ntu.edu.tw/~cjlin/libsvm.
4. Quinlan, J.R. (1993). C4.5: programs for machine learning. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
5. Mitchell T. (1997) Machine Learning, McGraw-Hill.
6. H Liu, H Motoda and L Yu, A Selective Sampling Approach to Active Selection, Artificial Intelligence, 159 (2004) 49-74.
7. N. Roy and A. McCallum. Toward optimal active learning through sampling estimation of error reduction. In Proceedings of ICML-2001, 18th International Conference on Machine Learning, pages 441–448, 2001.