High-speed Web Attack Detection through Extracting Exemplars from HTTP Traffic

Wei Wang
Interdisciplinary Centre for Security, Reliability and Trust (SnT Centre)
Université du Luxembourg, Luxembourg
wwangemail@gmail.com

Xiangliang Zhang
Division of Mathematical and Computer Sciences and Engineering
King Abdullah University of Science and Technolgy (KAUST), Saudi Arabia
xiangliang.zhang@kaust.edu.sa

ABSTRACT

In this work, we propose an effective method for high-speed web attack detection by extracting *exemplars* from HTTP traffic before the detection model is built. The smaller set of *exemplars* keeps valuable information of the original traffic while it significantly reduces the size of the traffic so that the detection remains effective and improves the detection efficiency. The Affinity Propagation (AP) is employed to extract the *exemplars* from the HTTP traffic. K-Nearest Neighbor(k-NN) and one class Support Vector Machine (SVM) are used for anomaly detection. To facilitate comparison, we also employ information gain to select key attributes (a.k.a. features) from the HTTP traffic for web attack detection. Two large real HTTP traffic are used to validate our methods. The extensive test results show that the AP based *exemplar* extraction significantly improves the real-time performance of the detection compared to using all the HTTP traffic and achieves a more robust detection performance than information gain based attribute selection for web attack detection.

1. INTRODUCTION

Web applications are rapidly growing to meet people's needs, accompanied with the increasing number of web flaws. In 2009, it is indicated [2] that 78 percent of the total reported vulnerabilities affected Web technologies. That number is about 10 percent higher than the number of flaws reported in the same period in 2008. A web-based Intrusion Detection System (IDS), therefore, is very desirable in order to build a defense-in-depth network security framework in current networks.

Anomaly detection has a potential to detect unforeseen attacks and thus attracts a lot of attention from the research communities. In practice, data in anomaly intrusion detection is massive and high dimensional in nature. High-speed processing of high dimensional massive audit data in most cases, therefore, is essential for a practical IDS to guarantee the real-time detection so that actions for response can be taken as soon as possible.

In general, as shown in Figure 1, there are four steps for anomaly intrusion detection: data collection, attribute (a.k.a. feature) construction, model building and anomaly detection. Many methods [11, 5, 6, 8] have been employed for anomaly intrusion detection. However, most of them mainly focus on attribute construction [5, 6, 8] and on detection algorithms [11]. In this paper, we aim at enhancing the capability of data processing for an IDS to achieve high-speed web attack detection.

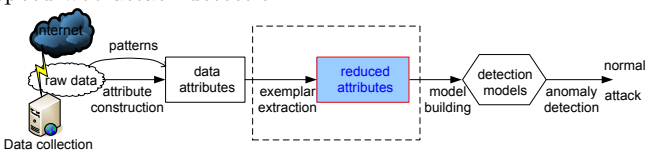


Figure 1: Steps for anomaly intrusion detection

Intuitively, the most direct way to high-speed process massive audit data is to reduce the data. There are generally three strategies to achieve the goal of data reduction. The first strategy is attribute abstraction, which transforms audit data in high-dimensional space to a space of fewer dimensions. The transformed attributes in the low-dimensional space can be regarded as new attributes abstracted from the original ones. The second is to find a subset of original attributes to represent the whole data. In this paper, we used the third strategy known as exemplar extraction. We extract *exemplars* and add a step as "*exemplar extraction*" between the step "attribute construction" and the step "model building" (in Figure1). The process of exemplar extraction is based on the training data. It extracts a smaller set of representative exemplars from the large amount of training data, so that the training is based on a smaller set and test is based on a compressed detection model. The performance of an IDS is thus significantly improved in the process of training and of test.

An *exemplar* refers to a factual data item (e.g., a HTTP request) that represents a number of similar data items. Comparing to randomly sampling data items (e.g., Netflow based network intrusion detection [1]), *exemplars* summarize massive audit data and thus better represent the audit data for anomaly detection.

In our previous work [12], we used Affinity Propagation (AP) and k-means to form exemplars from a HTTP traffic data set as well as from KDD 1999 data. We also used PCA (Principal Component Analysis) to abstract fewer attributes from the two data sets (the first strategy). The test

SAC'11 March 21-25, 2011, TaiChung, Taiwan.

results show that AP is more robust than k-means in exemplar extraction and that the AP based exemplar extraction has a better performance than attribute abstraction with PCA for intrusion detection. In this paper, we mainly investigate the performance of exemplar extraction for web attack detection. We employ AP for extracting exemplars from two large real HTTP traffic data (the third strategy). In order to facilitate comparison of data reduction methods in intrusion detection, we used Information Gain (IG) to extract the most relevant attributes from the audit data for high-speed detection (the second strategy).

Our contribution of this paper is summarized in the following three aspects. (1) We used clustering algorithms to extract exemplars from the original massive HTTP traffic. The detection process is thus significantly accelerated since only a smaller set of exemplars needs to be processed. The process of **Extracting *exemplars*** is further proved to be effective for anomaly intrusion detection; (2) we used the traditional attribute selection method (i.e., information gain) for web attack detection and compared it with exemplar extraction method introduced in this paper; (3) we used two real HTTP traffic data to validate our method. The extensive test results demonstrate the effectiveness and high-speed performance of our methods in web attack detection.

The remainder of this paper is organized as follows. Section 2 briefly introduces the related work. Section 3 describes the methods. Extensive experiments and comparative results are reported in Section 4. Concluding remarks follow in Section 5.

2. RELATED WORK

Krüegel and Vigna [6] are the first who used HTTP traffic for web attack detection. They investigated client queries with the parameters contained in these queries to detect potential attacks by six methods (e.g., length and structure of the parameters). Ingham and Inoue [5] collected a set of HTTP attacks and introduced a framework to compare different anomaly detection techniques for HTTP. Song et al. [8] used a mixture-of-markov-chains model for anomaly detection in web traffic. A most recent effort on dealing with unavailability of large training set for anomaly detection has been proposed in [7].

Most existing anomaly detection methods mainly focus on constructing attributes from audit data or on proposing an effective algorithm for the detection. In addition, most methods used all the attributes or the original data for building the detection models. However, some of the attributes may be redundant or even noise and therefore decrease the detection performance. Moreover, as many data may be very similar or even exactly the same, using all the data decreases the efficiency of building the detection models.

There are related work in data reduction for intrusion detection. Sung and Mukkamala [9] used Artificial Neural Networks (ANN) and SVM to identify some important attributes based on the performance comparison. In our previous work [10], we employed Information Gain and Bayesian Networks to select a key subset from the original set of attributes. The subset is then used for fast intrusion detection. In Netflow based network anomaly detection, many methods sampled data with a certain probability in order to deal with the massive traffic [1]. However, the randomly sampled traffic (e.g., flow or packet) is inherently a lossy process, discarding potentially useful information.

Different from attribute selection or randomly sampling, in this paper, we employ a strategy that finds *exemplars* from original data and then use the exemplars for training and detection. We also compare the strategy of exemplar extraction against attribute selection method for fast anomaly detection.

3. EXTRACTING EXEMPLARS WITH COMPARISON TO ATTRIBUTE SELECTION

In order to facilitate comparison, firstly we directly used k-Nearest Neighbor(k-NN) [11] and one class Support Vector Machine (SVM) [3] for anomaly detection, as these two methods have been demonstrated effective. We then used information gain for attribute selection as it has been proven to be effective [10]. Finally, we introduce Affinity Propagation (AP) [4] for exemplar extraction.

3.1 Direct use of k-NN and one class SVM

Both k-NN and one class SVM are used for building normal models (using attack-free data for training) based on which anomalies can be detected. A data vector normally consists of attributes. It represents an event or an object (e.g., a http request) that needs to detect whether it is normal or not. In the remainder of this paper, data item and data vector are used interchangeably. The methods of k-NN and SVM for anomaly detection are described in [12, 11].

3.2 Attribute selection with information gain

The information gain of a given attribute X with respect to the class attribute Y (e.g., normal or one of the attacks) is the reduction in uncertainty about the value of Y, after observing values of X. It is denoted as $IG(Y|X)$. The uncertainty about the value of Y is measured by its entropy defined as:

$$H(Y) = -\sum_i P(y_i) \log_2 P(y_i) \tag{1}$$

where $P(y_i)$ is the prior probabilities for all values of Y. The uncertainty about the value of Y after observing values of X is given by the conditional entropy of Y given X defined as:

$$H(Y|X) = -\sum_j P(x_j) \sum_i P(y_i|x_j) \log_2 P(y_i|x_j) \tag{2}$$

where $P(y_i|x_i)$ is the posterior probabilities of Y given the values of X. The information gain is thus defined as $IG(Y|X) = H(Y) - H(Y|X)$. According to this measure, an attribute X is regarded more correlated to class Y than attribute Z, if $IG(Y|X) > IG(Y|Z)$. By calculating information gain, we can rank the correlations of each attribute to the class and select key attributes based on this ranking.

3.3 Exemplar extraction with AP

AP is a newly developed exemplar-based clustering algorithm [4]. Let $\mathcal{E} = \{e_1, \ldots, e_N\}$ be a set of data items, and let $d(e_i, e_j)$ denote the distance (e.g., an Euclidean distance) between items e_i and e_j: $d(e_i, e_j) = \|e_i - e_j\|$. The fitness function is defined by:

$$\mathbf{E}(c) = \sum_{i=1}^{n} S(e_i, e_{c(i)}) \tag{3}$$

where $c(i)$ is the index of the exemplar representing the item e_i in a cluster; $S(e_i, e_j)$ is set to $-d(e_i, e_j)^2$ if $i \neq j$, and otherwise is set to a small constant $-s^*$ ($s^* \geq 0$).

$-s^*$ represents a preference that e_i itself be chosen as an exemplar. AP finds the mapping $\mathbf{c}$ that maximizes the fitness function $\mathbf{E}(c)$ defined by (3) to cluster the data items. The resolution of this optimization problem is achieved by a message passing algorithm [4].

In practical use, there may be some items that are exactly the same in the audit data. In our study, we used Weighted AP (WAP) [13] that adds more weights to the multiple-appear items so as to let one of them have more probability to be an exemplar. Let data set $\mathcal{E}' = \{(e_i, n_i)\}$ involve n_i copies of item e_i, for $i = 1, \ldots, L$. WAP considers the similarity matrix defined as:

$$S'(e_i, e_j) = \begin{cases} -n_i d(e_i, e_j)^2 & \text{if } i \neq j \\ -s^* & \text{otherwise} \end{cases}$$

Unlike k-means or k-centers, AP has no need to pre-define how many exemplars or clusters exist in the data. Instead, AP specifies the penalty s^* for allowing an item to become an exemplar. Note that for $s^* = 0$, the best solution is the trivial one, selecting every item as an exemplar.

4. EXPERIMENTS AND RESULTS

4.1 Data

We detect web attacks with real HTTP traffic streams[1]. In the experiments, we collected two large HTTP traffic on the main Apache server of an research institute. The two real traffic data sets represent two attack scenarios in the networks. The first traffic (http large) contains rare attacks while the second (http small) contains bursty attacks. The data are well labeled. The attacks in the traffic mainly includes JS XSS attacks, input validation error, URL decoding error, SQL-injection attacks, PHP remote file inclusion attacks and DoS attacks. In detail, the first traffic contains 36 attack requests distributed in more than 5.7 million requests while the second contains 239 attacks occurring in a very short interval in more than 1.4 million requests. The two traffic data sets are described in Table 1. Our method examines individual HTTP requests and models the content of script inputs. In order to reduce noise contained in the data streams, we filtered out most static requests (e.g., .html, .txt, .pdf) as well as widely known search engine robots (e.g., googlebot, Msnbot) before detection, because a static request cannot be an attack to the server.

In the experiments, we used character distribution of each path source in HTTP requests as the attributes. There are 256 types of ASCII code in total but only 95 types of ASCII code (between 33 and 127) appear in the HTTP request. The character distribution is computed as the frequency of each ASCII code in the path source of HTTP requests. As a consequence, each HTTP request (data item) is represented by a vector of 95 dimensions. Our goal is to identify whether an item is normal or anomalous.

4.2 Experiment settings

For exemplar extraction, in order to generate different number of exemplars, the preference parameter ($-s^*$) in AP needs to be set to various values. In the experiments, we set $-s^*$ as the $(l * N^2)$-th largest value of the N^2 similarities between all pairs of items and set $l = 1/2, 1/4, 1/8, 1/16, 1/32, 1/64$, respectively. Different from exemplar extraction in which only normal data is needed for training, attribute selection additionally requires attack data. We then used all the normal training data (http large plus http small) and randomly selected half number of attack data to rank the importance of attributes. For detection algorithm k-NN, we set $k = 1$ as it is a good choice for the detection [11] and it is thus *de facto* Nearest Neighbor (NN). We made our own MATLAB programs for all the algorithms except SVM for which LibSVM tools [3] were used. The data, the number of exemplars generated by AP from their corresponding training data as well as the number of attributes after using IG for attribute selection are given in Table 2.

[1]All the data as well as the programs used in this paper are available upon request to the first authors.

4.3 Exemplars extracted and attributes selected

As shown in Figure 2, an exemplar is a factual data item (i.e., a http request) presented in rows while an attribute is a value describing the data item and is presented in columns. For example, the second http request has been extracted by AP as an exemplar and attribute 2,5 and 92 are selected by IG for anomaly detection.

Two exemplars extracted by AP based on the HTTP large data are shown as:

1. /axis/publications/login.php?ref=/axis/publications /add.php&forgot=y
2. /mascotte/Stages/?ID=138&annee=2006_2007&lang=fr

The two exemplars are clearly shown to represent a set of similar http requests in the training data. For attribute selection with IG, the experiments show that only 52 out of 95 attributes (characters) contribute to the detection. The important attributes (characters) are listed (ranked by their contribution to the detection) as:

. / r ? e \ - = w c o |) (; b g t p h j A f > " a k O P m s < u v _ d R [ ] I 5 X * % T l i 7 , ‘ x W

4.4 Detection speed

For **exemplar extraction**, the computing time for generating exemplars, for training the models and for the detection are summarized in Table 3. For **attribute selection**, the computing time for attribute selection (with IG), for training the models and for the detection are also listed in Table 4. All the experiments were carried out on a machine with Intel Dual Core 2.80GHz and memory 4GB.

It is seen from Table 3 and Table 4 that the process of exemplar extraction improves a lot the detection speed for both NN and SVM. For example, while using all the data for building the detection models, NN needs 7670 seconds for the detection. In contrast, it only requires 235 seconds for the detection with 298 exemplars generated by AP. As a result, the process of exemplar selection improves the computation time by more than 23 times in the best case. It is also observed from Table 4 that attribute selection with IG improves the detection speed, although it is not as effective as exemplar extraction.

4.5 Detection accuracy on http large data set

Table 1: The HTTP traffic used in the experiments

Data set	Before Filtering		After Filtering		# normal requests	# attack requests	Duration
	File size	# requests	File size	# requests			
http large	1,536MB	5,700,949	53.5MB	265,752	265,716	36	10 days 21 hrs
http small	561.2MB	1,449,379	9.5MB	40,095	39,856	239	3 days 3 hrs

Table 2: The data, the number of exemplars generated from training data with AP and the number of attributes selected with IG

Data set	# training items	# test items	# attributes	# exemplars with AP						#attributes with IG
				$l = 1/2$	$l = 1/4$	$l = 1/8$	$l = 1/16$	$l = 1/32$	l=1/64	
http large	8000	257751	95	298	384	461	532	678	977	52
http small	7000	33095	95	420	529	613	692	781	975	52

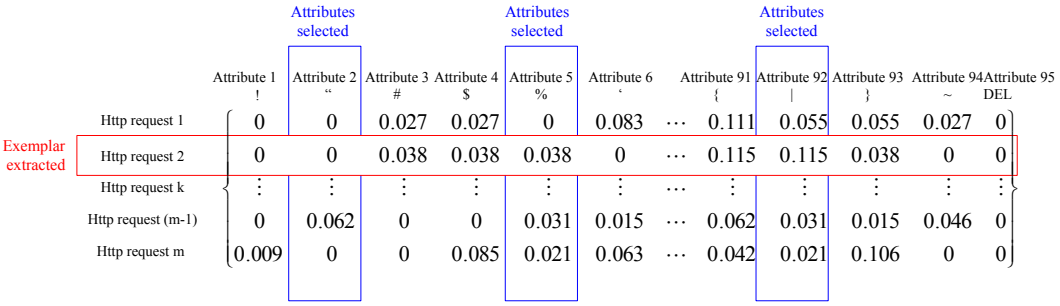


Figure 2: Exemplars and attributes in the data

Table 3: The computing time (sec.) for anomaly detection *with or without the process of exemplar extraction*.

Data set	training data	time for) forming exemplars	time for anomaly detection	
		AP	NN	SVM
http large	8000 items	/	7670	919
	298 exemplars	94	235	28
	384 exemplars	108	294	36
	461 exemplars	246	478	43
	532 exemplars	251	645	51
	678 exemplars	97	617	63
	977 exemplars	125	628	94
http small	7000 items	/	592	101
	420 exemplars	139	29	5
	529 exemplars	141	38	6
	613 exemplars	158	45	8
	692 exemplars	158	48	8
	781 exemplars	176	56	10
	975 exemplars	162	67	12

Table 4: The computing time (sec.) for anomaly detection *with or without attribute selection* (IG)

Data set	training data	time for attribute selection	time for anomaly detection	
		AP	NN	SVM
http large	8000 items	/	4114	558
	293 exemplars	218	116	19
	378 exemplars	180	154	25
	452 exemplars	187	175	31
	538 exemplars	190	215	37
	653 exemplars	291	287	44
	1029 exemplars	200	448	70
http small	7000 items	/	382	72
	395 exemplars	84	20	3
	497 exemplars	87	26	4
	592 exemplars	85	31	5
	664 exemplars	97	35	6
	736 exemplars	371	39	6
	819 exemplars	247	45	7

The comparative results in terms of detection accuracy on the http large data set are shown in Figure 3 and Figure 4. It is seen that using a smaller set of exemplars does not affect and in many cases even improves the detection accuracy. Normally as the number of exemplar increases, the detection accuracy is improved too. However, as shown in the Figures, selecting approximately 10 percent of the total number of data items as exemplars gives satisfied detection results. In many cases it performs better than using all the data. For example, it is observed from Figure 3 that only using 977 exemplars achieves better detection accuracy than using all the original data (8000 items) for k-NN and achieves comparable results for SVM.

It is observed in Figure 3 and 4 that using only 52 attributes instead of 95, the detection accuracy is very similar. Regarding the detection algorithms, k-NN is shown to be more effective than SVM for detection of http large data. It is also seen from the Figures that using the combination of exemplar extraction and attribute selection still achieves comparable detection accuracy comparing to using all the number of data items and attributes.

4.6 Detection accuracy on http small data set

The comparative results in terms of detection accuracy on the http small data are shown in Figure 5. It is observed from the Figures that exemplar selection is a very good tool. Selecting a smaller set of exemplars for training does not decrease detection accuracy and in most cases it even improves the detection performance. Figure 5 also shows that k-NN is better than SVM for the detection and this finding is consistent with the results on the http large data.

5. CONCLUDING REMARKS

Data for web attack detection is increasingly massive. Processing a large amount of HTTP traffic in high speed, therefore, has become a challenge for real-time detection. In this paper, we extract a smaller set of representative exemplars from the large set of original data. The exemplars are then fed as data input for training the detection models. The detection process hence speeds up a lot since the incoming test data is only based on a compressed detection model.

A newly developed clustering algorithm Affinity Propagation (AP) [4] is employed to extract exemplars. Exemplar selection is a necessary step for intrusion detection. We have observed that many data items are exactly the same in the real data collected from our institutes. In the large HTTP traffic, for example, there are only 1943 distinct data items among 8000 data items. Exemplar selection reduces reduplicate data items as well as similar ones in the original data so that the detection process is accelerated a lot. For example, for k-NN, a test data item only needs to compare itself with a smaller set of exemplars other than with the large amount of original data items.

In order to have a comparative view of data reduction in web attack detection, we also introduced Information Gain based attribute selection for the detection. While exemplar extraction methods extract a smaller set of representative data items, attribute selection aims to select a smaller set of important attributes. The process of attribute selection can also speed up the detection. In order to select a representative subset of attributes from the original set, a large amount of attack data as well as normal data need to feed to the method. In practise, however, a large amount of attack data is difficult to collect. This may in some circumstances prevent the use of IG based attribute selection method for intrusion detection. This also helps to explain why IG based attribute selection is not as effective as exemplar extraction for anomaly detection.

Two real HTTP traffic sets collected in our institutes were used to validate our methods. The extensive experimental results show that with the same detection algorithm k-NN, the AP based exemplar selection improves the detection efficiency by 23 times in the best cases compared to using all the data. It behaves more robust than IG based attribute selection. In order to enhance the detection performance, as a result, exemplar selection is highly suggested to be made before building the detection model.

Our future work is summarized in two main aspects: (1) using more attributes of the HTTP traffic to improve the detection performance; (2) dealing with the problem of "concept drift" in the traffic data.

6. REFERENCES

[1] D. Brauckhoff, K. Salamatian, and M. May. A signal-processing view on packet sampling and anomaly detection. In *INFOCOM*, 2010.

[2] Cenzic. Web application security trends report q1-q2, 2009. *Technique Report*, 2009.

[3] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm.

[4] B. J. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, 315(5814):972–976, 2007.

[5] K. L. Ingham and H. Inoue. Comparing anomaly detection techniques for http. In *RAID*, 2007.

[6] C. Krügel and G. Vigna. Anomaly detection of web-based attacks. In *ACM CCS*, 2003.

[7] W. Robertson, F. Maggi, C. Kruegel, and G. Vigna. Effective anomaly detection with scarce training data. In *NDSS*, 2010.

[8] Y. Song, A. D. Keromytis, and S. J. Stolfo. Spectrogram: A mixture-of-markov-chains model for anomaly detection in web traffic. In *NDSS*, 2009.

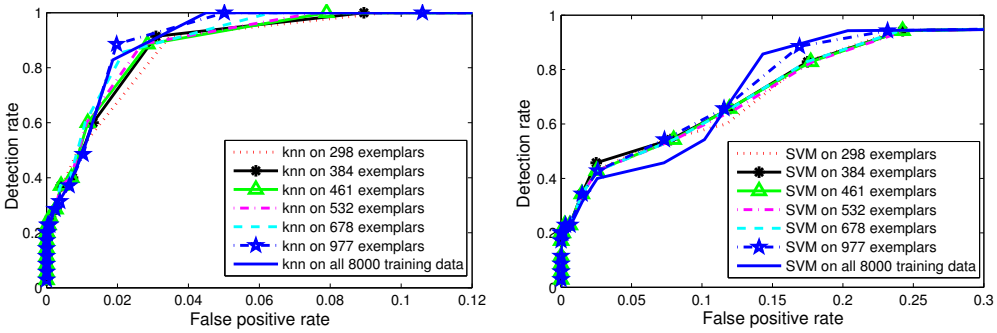
[9] A. H. Sung and S. Mukkamala. Feature selection for intrusion detection using neural networks and support vector machines. In *82nd Annual Meeting of the Transportation Research Board*, 2003.

[10] W. Wang, S. Gombault, and T. Guyet. Towards fast detecting intrusions: using key attributes of network trafic. In *ICIMP*, Jul 2008.

[11] W. Wang, X. Zhang, and S. Gombault. Constructing attribute weights from computer audit data for effective intrusion detection. *J. Sys. and Soft.*, 82(12), 2009.

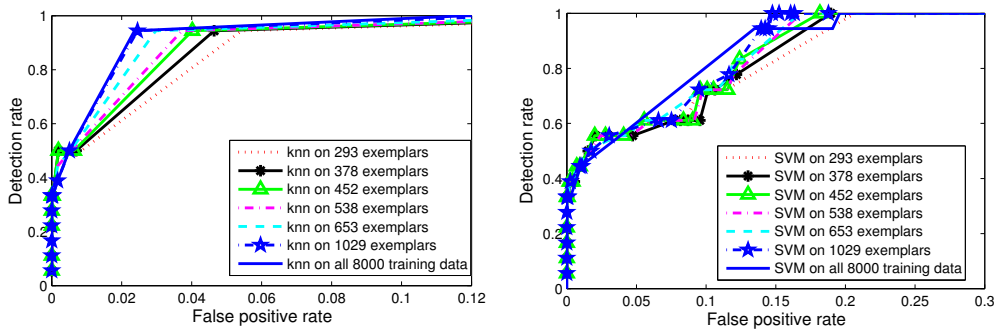
[12] W. Wang, X. Zhang, and G. Pitsilis. Abstracting audit data for lightweight intrusion detection. In *Proceedings of 6th International Conference on Information Systems Security (ICISS' 2010)*, 2010.

[13] X. Zhang, C. Furtlehner, and M. Sebag. Data streaming with affinity propagation. In *ECML/PKDD*, 2008.



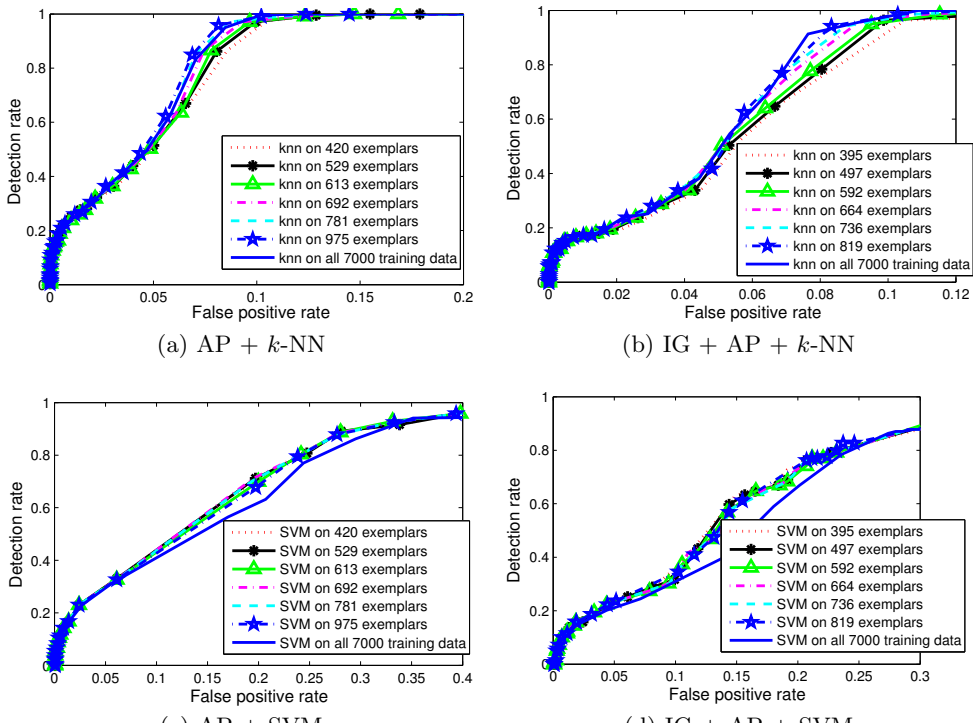
(a) AP + k-NN (b) AP + SVM

Figure 3: Comparative results on the large HTTP traffic data *with or without exemplar extraction*



(a) IG + AP + k-NN (b) IG + AP + SVM

Figure 4: Results on the large HTTP traffic data with exemplar selection after attribute selection with IG.



(a) AP + k-NN (b) IG + AP + k-NN

(c) AP + SVM (d) IG + AP + SVM

Figure 5: Results on the small HTTP traffic data with exemplar selection as well as its performance after attribute selection with IG.