2013 IEEE 12th International Symposium on Network Computing and Applications

A Real Time Adaptive Intrusion Detection Alert Classifier for High Speed Networks

Hassen Sallay, Adel Ammar, Majdi Ben Saad
Al Imam Mohammad Ibn Saud Islamic University (IMSIU), KSA
hmsallay@imamu.edu.sa

Sami Bourouis
Taif University, KSA
s.bourouis@tu.edu.sa

***Abstract*—With the emergence of High Speed Network (HSN), the manual intrusion alert detection become an extremely laborious and time-consuming task since it requires an experienced skilled staff in security fields and need a deep analysis. In addition, the batch model of alert management is no longer adequate given that labeling is a continuous time process since incoming intrusion alerts are often collected continuously in time. Furthermore, the static model is no longer appropriate due to the fluctuation nature of the number of alerts incurred by Internet traffic fluctuation nature. This paper proposes an efficient real time adaptive intrusion detection alert classifier dedicated for high speed network. Our classifier is based an online self-trained SVM algorithm with several learning strategies and execution modes. We evaluate our classifier against three different data-sets and the performance study shows an excellent results in term of accuracy and efficiency. The predictive local learning strategy presents a good tradeoff between accuracy and time processing. In addition, it does not involve a human intervention which make it an excellent solution that satisfy high speed network alert management challenges.**

keyword - Intrusion detection, Alert classification, High speed network, Online and self-training learning, SVM.

I. Introduction

Intrusion Detection Systems (IDSs) are intended to protect digital assets and to keep up secured systems. Since they generate a huge number of alerts which are mixed with false positives, it will be difficult to manage incoming alerts and to take suitable actions for them. Thereby, the necessity to efficiently filter false alerts within an automatic method in real time is challenging since manual processing is extremely laborious and time-consuming. The emergence of High Speed Network (HSN) reaching rate of 10/40/100 Gig/sec adds difficulties along the alert management process. In HSN the IDS generates a huge amount of alerts in a short period of time due the high traffic speed of HSN. Helping the administrator to understand what the IDS is addressing, to analyze properly alerts, and to save his effort and time requires an efficient classification of these alerts. Several promising intrusion alert classification techniques have been developed in the past decades. An overview of current intrusion detection techniques and related issues was proposed in [1] [2]. The main common objective of existing classification-based methods is to classify all traffic as either normal or malicious in some manner in order to reduce false positives and false negatives alarms. For example, there are some approaches that correlate alerts based on similarity metrics between alert attributes such as source IP address, target IP address, port number for the source and the target, time, attack type, etc. [3], [4]. This category of approaches is considered effective; however, it fails to find the causal relationship between alerts. Another category refers to all methods that manage alerts by considering the dependencies between alerts and matching prerequisites with the consequences [5], [6]. The major advantage of this category is its capacity to find out the casual relationship between alerts. However, prerequisites and consequences need knowledge of individual attacks, and this process is time-consuming. Recently, data mining and machine learning approaches have been used in this growing area in order to improve the performance of existing systems [7], [8], [9], [10]. These works use machine learning algorithms to classify intrusion alerts. Examples include SVM that has been found to be an effective solution for current problems [11], [12], [13], [14].

Although several approaches have been developed in the literature, they still do not deal with HSN requirements which are related to efficiency, real-time and adaptability constraints as we explain in the following.

1) efficiency: the manual analysis is no longer possible given the huge number of alerts due to the high speed network throughput. Furthermore the labeling process is both expensive and time-consuming since it requires an experienced skilled staff in security fields and need a deep analysis.
2) Real-time: the batch model of alert management is no longer adequate given that labeling is a continuous time process since incoming intrusion alerts are often collected continuously in time.
3) Adaptability: the static model is no longer appropriate due to the fluctuation nature of the number of alerts incurred by Internet traffic fluctuation nature.

To the best of our knowledge this is the first work that attempts to tackle simultaneously the above security related challenges in the context of high speed networks. The rest of this paper is organized as follows. Section 2 presents the solution technical choices. Section 3 gives a deep performance study. Section 4 discusses the results and finally section 5 concludes the paper.

II. An online Self-trained Alert classifier

In this section, we present our online self-trained classifier and the motivations behind it. Our classifier is mainly based on support vector machines (SVM) [15] which is one of the most successful learning algorithms for binary classification [16], [17]. As it is known, the key idea of SVM-classification is

978-0-7695-5043-5/13 $26.00 © 2013 IEEE
DOI 10.1109/NCA.2013.16

to find a separation boundary (hyperplane), using the kernel trick, which maximizes the margin between two classes in the data. However, the major problem of SVM is the fact that it is time and space consuming especially in the case of large training data sets. Fortunately [18] provides Sequential minimal optimization (SMO) technique as an efficient iterative divide-and-conquer based strategy solving efficiently the aforementioned shortcomings. SMO proceeds by decomposing the overall quadratic programming problem into smaller quadratic programming sub-problems without invoking an iterative numerical routine for each sub-problem. The overall memory requirement of the SMO algorithm is linear in the size of training data and therefore it allows the use of large size samples in the training process.

In several real world problems input data is dynamic, then classical SVM in batch setting is no longer an efficient approach and thus an online extension is more appropriate. This is exactly our case as we shall explain below. In fact, in high speed network, incoming intrusion alerts are often collected continuously in time. So, a false alert filtering is considered as a continuous task. In contrast to batch model, the online model presents to the filter a sequence of false alerts on real time. An Online approach is therefore very intuitive to use in order to decrease misclassified data efficiently and in near real time.

Basically online SVM does not need to update any correctly classified example outside the margins due to the Karush-Kuhn-Tucker conditions, which consider support vectors (SVs) closer to the hyperplane are most uncertain and informative [19]. Consequently, the computational cost can be greatly reduced and without affecting classification performance since the old hypothesis is used as the starting sample for re-training and discarding all non-SVs samples. In the last decade, only few works have been proposed for building of online SVM [20], [21], [22], [23]. In our work, we adopted the online SVM developed in [21], [19] which has shown excellent results especially in the case of large data-sets.

On the other hand the SVM is generally used as a supervised learning method because the only information available is a finite training data set. The supervised learning which uses only labeled data becomes an unrealistic approach in the context of high speed network since the manual analysis of all alerts is no longer possible, an intuitive choice is to use a semi-supervised learning (SSL) approach. Briefly, semi-supervised learning (SSL) aims to use both labeled and unlabeled data to improve the learning process performance [24], [25]. In self-training the learner uses its proper decision of labeling to perform the classification of the new coming alerts. SSL will be most useful when there are far more unlabeled data than labeled data. Obviously SSL could degrade the performance when mistakes reinforce themselves.

Briefly our algorithm of self-training is resumed by the following pseudo code.

Algorithm 1: Online Self Training SVM Algorithm

Require: $LSize \geq 100$, $TSize \geq 100$, $DatasetFile$ is the Data-set Source File
if $SVMDatasetFile$ does not exist **then**
  $SVMDatasetFile \leftarrow FormatToSVM(DatasetFile)$ {Format data-set source file content to SVM}
end if
$ILFile \leftarrow Shuffle(SVMDatasetFile, LearningSize)$ {Extract random entries for inital learning phase}
$SVS \leftarrow svm\_learn(ILFile)$ {Initial SVM Learning}
$GLSize \leftarrow 0$
$x \leftarrow 0$
repeat
  $TFile \leftarrow GetNextTEntries(SVMDatasetFile, TSize)$ {Extract next $TSize$ entries from Data-set file for testing phase}
  $TOutputFile_x \leftarrow svm\_test(TestingFile, SVS)$
  $GTOutputFile \leftarrow CONCAT(GTOutputFile, TOutputFile_x)$
  $GLSize \leftarrow GLSize + TSize$
  if $GlobalLearning$ and $GL\_Threshold$ reached **then**
    {Global Learning}
    $LFile \leftarrow CONCAT(ILFile, GetEntries(SVMDatasetFile, 1, GLSize))$
    if $PredictiveGlobalLearning$ **then**
      $LFile \leftarrow ReplaceLabels(LFile, GTOutputFile)$ {Replace each entry label with the corresponding one from global testing output file.}
    end if
    $SVS \leftarrow svm\_learn(LFile)$
  else
    {Local Learning}
    if $AccurateLocalLearning$ **then**
      $LFile \leftarrow CONCAT(ILFile, GetEntries(SVMDatasetFile, 1, GLSize))$
    else
      $SVMLabLFile \leftarrow ReplaceLabels(TFile, TOutputFile_x)$ {Set entries labels from the last testing output file}
      $LFile \leftarrow CONCAT(ILFile, GetEntries(SVMDatasetFile, 1, (GLSize - TSize)))$
      $LFile \leftarrow CONCAT(LFile, SVMLabLFile))$ {Append entries with new labels to $LFile$ }
    end if
    $SVS \leftarrow svm\_learn(LFile)$
  end if
  $x \leftarrow x + 1$
until End Of $SVMDatasetFile$ reached

A. Algorithms Design

For this work, we opt to two kind of update categories performed on two kinds of learning mode:

1) Accurate learning (AL): is based on data labeled by the network monitor/analyst specialist.
2) Predictive learning (PL): is based on a data labeled by our SVM classifier.

The update process (active learning) involves two types:

1) Local update (L): is an incremental process performed after each short period. The classification decision will depend incrementally on the previous labeled data.
2) Global update (G): is performed after long period. The classification decision will depend on all previous classified data.

Description of different Learning scenarios:

- A1 (ALL-NGL): we use Accurate Learning with Local update only (No Global Learning update).
- A2 (PLL-NGL): we use Predictive Learning with Local update only (No Global Learning update).
- A3 (NLL-AGL): we use Accurate Learning with Global update only (No Local Learning update).
- A4 (NLL-PGL): we use Predictive Learning with Global update only (No Local Learning update).
- A5 (ALL-AGL): we use Accurate Learning on both Local and Global updates.

- A6 (PLL-PGL): we use Predictive Learning on both Local and Global updates.
- A7 (ALL-PGL): we use Accurate Learning on the Local update and Predictive Learning on the Global update.
- A8 (PLL-AGL) : we use Predictive Learning on the Local update and Accurate Learning on the Global update.

Since predictive global learning strategy will intuitively introduce a great learning mistake reinforcement, we limit in this study only on the A1, A2, A3, A5, and A8 algorithms.

III. VALIDATION

A. Datasets description

In this work, in order to surround the behavior of our algorithms, we evaluate them according to three different data sets: Realistic data set (Kyoto data set), Virtualized data Set (KDD data set) and Synthetized data set (ISCX data set) which are described briefly in what follows.

1) Virtualized data set: KDDCUP'99 Data Set
The KDDCUP'99 is the mostly widely used dataset for the evaluation of the network based anomaly detection systems. It was established by Stolfo and al. [26] and is built based on the data captured in DARPA'98 IDS evaluation program [27]. KDDCUP'99 is about 4 gigabytes of compressed raw (binary) tcpdump data of 7 weeks of network traffic and consists of 41 features and is labeled as either normal or an attack. The attacks fall in one of the following categories: denial of service attack (DoS), user to root attack (U2R), remote to local attack (R2L) and probing attack. However, currently this dataset presents several issues [28], [29] related to its accuracy and ability to reflect real network behavior. Indeed, it encompasses huge number of redundant records [28] which causes the learning algorithms to be biased towards the frequent records and thus prevents them from learning infrequent records which are usually more harmful to networks. As Kyoto dataset, KDDCup'99 dataset include only description of the traffic and not the real load.

2) Realistic data set :Kyoto Data Set
The Kyoto 2006+ [30] is an evaluation dataset of network detection mechanism obtained from diverse honeypots from November 2006 to August 2009. This dataset capture the real network traffic without any human alteration or deletion. It encompasses the recent trends of network attacks distinguished from normal traffic via the use of honeypots. It consists of 24 statistical features where 14 conventional features extracted from KDDCUP'99 dataset and 10 additional features that may enable to investigate more effectively what kind of attacks happened in the networks. However, Kyoto dataset include only description of the traffic without delivering the real load which limits its scope of uses by researchers.

3) Synthetized Data set: ISCX Data Set
The Information Security center of Excellence (ISCX) dataset [31] is built on 2011 for the purpose to generate benchmark datasets for intrusion detection during one week. ISCX dataset consists of 11 statistical features (time_stamp, source_bytes, dst_bytes, source_packets, dst_packets, protocol, direction, Tag, source_ip, dst_ip) captured via the use of single interface on the switch that all traffic are directed to it. The different attack scenario are executed at different periods and each attack composed by 5 steps: information gathering and reconnaissance (passive or active), vulnerability identification and scanning, gaining access and compromising a system, maintaining access and creating backdoors, and covering tracks. The main advantage of the ISCX dataset is its inclusion of the captured load and not only description of it which enables several testing and validation of detection mechanisms.

B. Experiments metrics

We introduce in this section the measures used in our experiments. We define the true positive (TP), false positive (FP), true negative(TN), false negative (FN) as follows:

- TP : a true positive alert classified as true positive.
- FP : a false positive alert classified as true positive.
- TN : a false positive alert classified as false positive.
- FN : a true positive alert classified as false positive.

In order to study the performance of our different alert classification algorithms, we compute the following measures: Sensitivity ($[TP/(TP+FN)]$), Fall-out ($[FP/(FP+TN)]$), Accuracy ($[(TP+TN)/(TP+FN+FP+TN)]$), Specificity ($[TN/(FP+TN)]$), Precision ($[TP/(TP+FP)]$), Negative predictive value ($[TN/(TN+FN)]$), False discovery rate ($[FP/(FP+TP)]$), Matthews correlation coefficient, F measure, online learning processing time.

C. Results and analysis

We have conducted our experiments in two modes.

- With Limit Mode: in witch we interrupt the execution of learning process when it reaches 5% of the overall number of support vectors (SV) of the overall alerts learning data set.
- Without Limit Mode: in which the learning process is not interrupted and we all generated support vectors are considered.

In all our experiments, we opt for cross-validation by varying the number of learning and testing data. We fixed the learning size while varying the testing size four times from 250 to 1000 alerts. This test was repeated 4 times by considering the learning size as 500, 1000, 1500, and 2000.

It is noteworthy that we have conducted all possible experiments related different scenarios. To avoid redundancy, we select as an example the case where the learning data size is 1000 alerts and the testing size is 500 alerts for both No-limit and with-limit modes and for the 3 data sets.

1) Results on the KDD'99 Data set:

- No-limit Learning case

Figure 1 (table) presents the obtained evaluation measure values for this specific case. According to this table, we can notice that all strategies perform slightly similarly for all used metrics. The best result was obtained with the ALL-AGL strategy which is an expectable result since it is based on the intervention of the analysis all time. The preferred scenario for us is PLL-NGL since it provides automatically good and acceptable result as compared to other scenarios without the intervention of the analyst (i.e. in real time and completely independent from the user).

	ALL-AGL	ALL-NGL	NLL-AGL	PLL-AGL	PLL-NGL
Sensitivity	0.98908	0.98834	0.98690	0.98397	0.98247
Specificity	0.99893	0.99858	0.99858	0.99751	0.99663
FPR	0.00107	0.00142	0.00142	0.00249	0.00337
FDR	0.00400	0.00600	0.00600	0.01000	0.01400
PPV	0.99560	0.99413	0.99413	0.98974	0.98607
NPV	0.99734	0.99716	0.99681	0.99610	0.99574
Accuracy	0.99770	0.99657	0.99629	0.99486	0.99386
MCC	0.99000	0.98900	0.98800	0.98400	0.98000
F2	0.99000	0.98900	0.98800	0.98500	0.98300
F1/2	0.99400	0.99300	0.99300	0.98900	0.98500
F1	0.99200	0.99100	0.99100	0.98700	0.98400

Fig. 1. Results for the KDD'99 data set in the case of No-limit mode where learning size is L=1000 and Test size is T=500 (L1000-T500) for different strategies.

To evaluate the performance of our classifier for different misclassification costs, we have used also the ROC analysis as shown in figure 2. According to this figure, we can conclude that all strategies perform similarly. In particular, PLL-NGL gives good tradeoff between false positives rate and true positives rate.

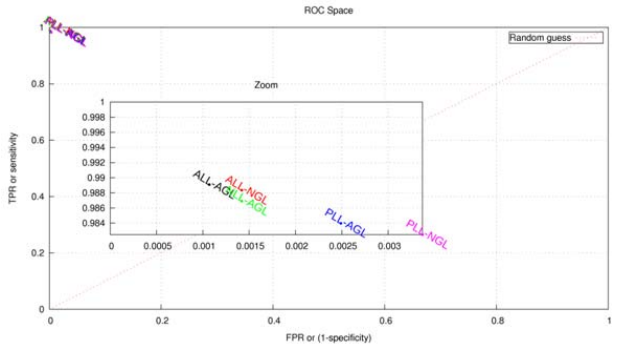

Fig. 2. ROC curves for all different strategies.

The loss ratio in accuracy of the predictive local learning is not significant. This ratio is in the range of [0.001, 0.0072] (see figure 3). This prove that we can rely on the PLL strategy since it represents a good tradeoff between accuracy, time processing and without human involvement.

- With-limit Learning case

Figure 4 (table) shows the obtained evaluation measures for with-limit case. According to this table, we deduced the same conclusion as in the No-limit case. In other words, all

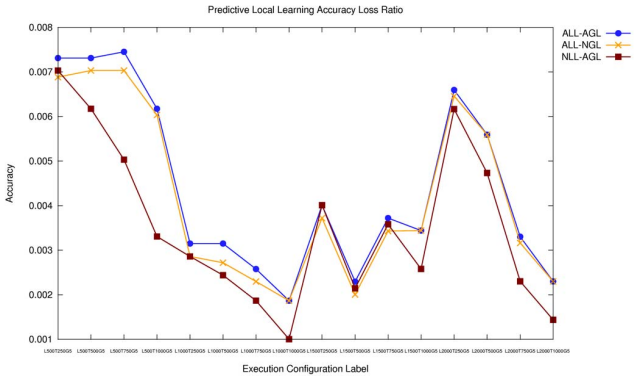

Fig. 3. PLL accuracy loss ratio for the KDD'99 data set in the case of no-limit mode.

strategies perform similarly, the ALL-AGL strategy gives the best results and the PLL-NGL provides good results without the intervention the analyst.

	ALL-AGL	ALL-NGL	NLL-AGL	PLL-AGL	PLL-NGL
Sensitivity	0.98908	0.98834	0.98690	0.98397	0.98247
Specificity	0.99893	0.99858	0.99858	0.99751	0.99663
FPR	0.00107	0.00142	0.00142	0.00249	0.00337
FDR	0.00400	0.00600	0.00600	0.01000	0.01400
PPV	0.99560	0.99413	0.99413	0.98974	0.98607
NPV	0.99734	0.99716	0.99681	0.99610	0.99574
Accuracy	0.99700	0.99657	0.99629	0.99486	0.99386
MCC	0.99000	0.98900	0.98800	0.98400	0.98000
F2	0.99000	0.98900	0.98800	0.98500	0.98300
F1/2	0.99400	0.99300	0.99300	0.98900	0.98500
F1	0.99200	0.99100	0.99100	0.98700	0.98400

Fig. 4. Results for the KDD'99 data set in the case of With-limit mode where learning size is L=1000 and Test size is T=500 (L1000-T500) for different strategies.

The loss ratio in accuracy of the predictive local learning is not significant. This ration is in the range of [0.001, 0.016] (see figure 5).

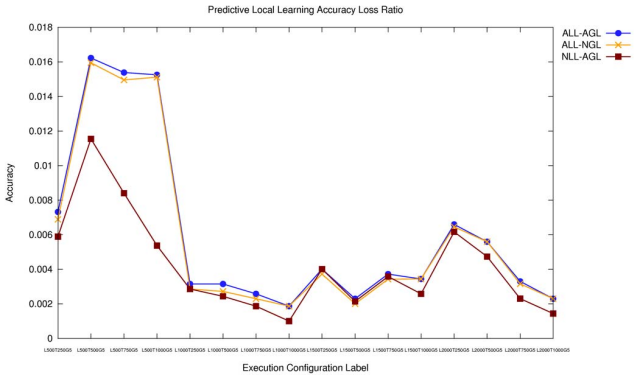

Fig. 5. PLL accuracy loss ratio for the KDD'99 data set in the case of With-limit mode.

2) Results Obtained with Kyoto Data Set:

- No-limit Learning case

Figure 6 (table) shows the evaluation measure values. According to this table, we can remark that all strategies give good similar results for all used metrics. For the PLL-NGL strategy, we obtain very good results. Thus, we can again opt for this strategy for alert classification. These results encourage us to direct our effort to an online unsupervised approach.

	ALL-AGL	ALL-NGL	NLL-AGL	PLL-AGL	PLL-NGL
Sensitivity	0.97489	0.97489	0.97455	0.97137	0.96995
Specificity	0.99217	0.99217	0.99023	0.99026	0.99024
FPR	0.00783	0.00783	0.00977	0.00974	0.01176
FDR	0.00800	0.00800	0.01000	0.01200	0.01200
PPV	0.99168	0.99168	0.98960	0.98753	0.98753
NPV	0.97633	0.97633	0.97605	0.97303	0.97165
Accuracy	0.98371	0.98371	0.98257	0.98000	0.97929
MCC	0.96800	0.96800	0.96500	0.96000	0.95900
F2	0.97800	0.97800	0.97800	0.97500	0.97300
F1/2	0.98800	0.98800	0.98700	0.98400	0.98400
F1	0.98300	0.98300	0.98200	0.97900	0.97900

Fig. 6. Results for the Kyoto data set in the case of No-limit mode where learning size is L=1000 and Test size is T=500 (L1000-T500) for different strategies.

The accuracy when changing the learning and testing data size is high for all strategies. An illustration of this is shown in fig 7 According to this figure, we can see clearly that the best result is obtained with ALL-AGL which is expected because this scenario uses both local and global accurate learning. We can see also that for all scenarios the accuracy increases as we increase the learning size.

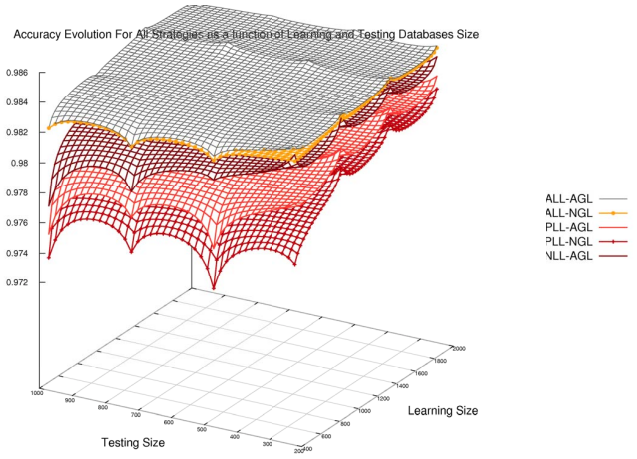

Fig. 7. Accuracy evolution for all strategies for different learning and testing sizes.

On the other hand, the loss ratio in accuracy of the predictive local learning is not significant. This ratio is well illustrated in figure 8.

The study of the testing phase processing time as a function of SVs number is given in figure 9. We notice that the difference in computational time between the 5 strategies is not significant as we increase the number of support vectors.

- With-limit Learning case

Figure 10 (table) shows the evaluation measure values. According to this table, we can remark that all strategies give

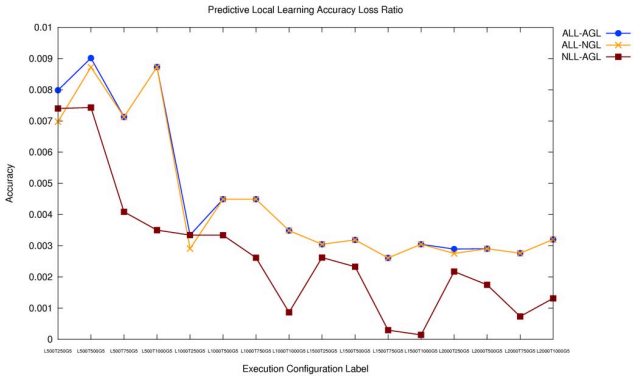

Fig. 8. PLL accuracy loss ratio for the Kyoto data set in the case of No-limit mode.

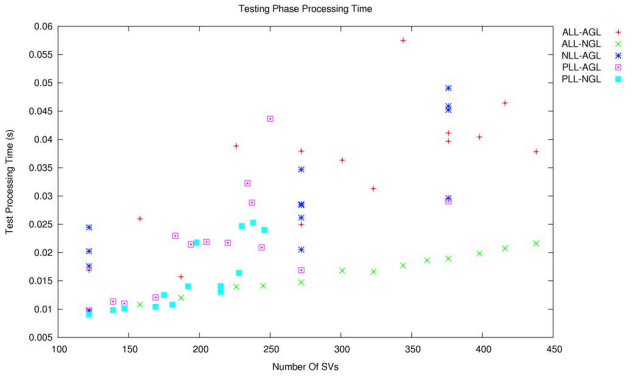

Fig. 9. Testing Phase Processing Time as function of Number of SV for the Kyoto data set in the case of No-limit mode.

good similar results for all used metrics. For the PLL-NGL strategy, we obtain very good results. Thus, we can again opt for this strategy for alert classification. These results encourage us to direct our effort to an online unsupervised approach.

	ALL-AGL	ALL-NGL	NLL-AGL	PLL-AGL	PLL-NGL
Sensitivity	0.96780	0.96724	0.96448	0.96493	0.96323
Specificity	0.99127	0.99127	0.99237	0.98930	0.98872
FPR	0.00873	0.00873	0.00763	0.01070	0.01128
FDR	0.00900	0.00900	0.00800	0.01100	0.01200
PPV	0.99079	0.99079	0.99198	0.98871	0.98812
NPV	0.96945	0.96890	0.96614	0.96669	0.96504
Accuracy	0.97971	0.97943	0.97857	0.97729	0.97614
MCC	0.96000	0.95900	0.95700	0.95500	0.95300
F2	0.97200	0.97200	0.97000	0.97200	0.96800
F1/2	0.98600	0.98600	0.98600	0.98400	0.98300
F1	0.97900	0.97900	0.97800	0.97700	0.97600

Fig. 10. Results for the Kyoto data set in the case of With-limit mode where learning size is L=1000 and Test size is T=500 (L1000-T500) for different strategies.

The study of the learning phase processing time gives also the same results found for no-limit case. This results is well illustrated in the figure 11.

The loss ratio in accuracy of the predictive local learning is not significant. This ratio varies from -0.008 to +0.01 and the accuracy variation for all configurations (different learning

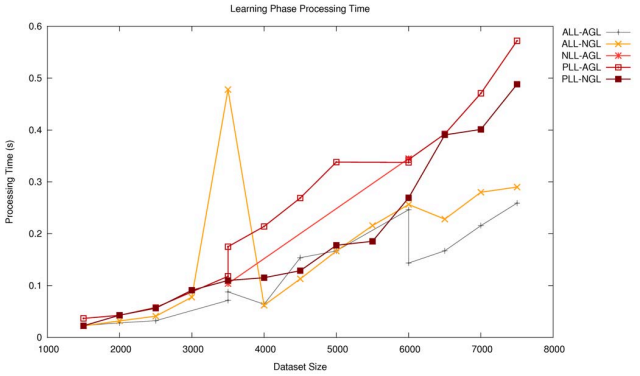

Fig. 11. Learning Phase Processing Time as function of data set size for Kyoto data set in the case of With-limit mode.

size and different testing size) is shown in figure 12. In this case our predictive learning outperforms the others strategies since we have a negative ratio value -0.008.

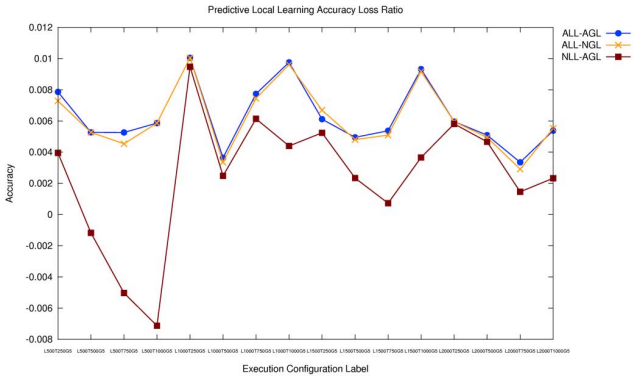

Fig. 12. PLL accuracy loss ratio for the Kyoto data set in the case of With-limit mode.

3) Results Obtained with ISCX Data Set:

- No-limit Learning case

Figure 13 (table) shows the evaluation measure values. Good results for all used metrics are obtained and especially for the PLL-NGL strategy.

	ALL-AGL	ALL-NGL	NLL-AGL	PLL-AGL	PLL-NGL
Sensitivity	0.93224	0.93224	0.93261	0.93313	0.93332
Specificity	0.99535	0.99535	0.99451	0.99201	0.99160
FPR	0.00465	0.00465	0.00549	0.00799	0.00840
FDR	0.00300	0.00300	0.00300	0.00400	0.00500
PPV	0.99746	0.99746	0.99700	0.99561	0.99538
NPV	0.88235	0.88235	0.88310	0.88423	0.88460
Accuracy	0.95357	0.95357	0.95357	0.95314	0.95314
MCC	0.90300	0.90300	0.90300	0.90200	0.90200
F2	0.94500	0.94500	0.94500	0.94500	0.94500
F1/2	0.98400	0.98400	0.98300	0.98200	0.98200
F1	0.96400	0.96400	0.96400	0.96300	0.96300

Fig. 13. Results for the ISCX data set in the case of No-limit mode where learning size is L=1000 and Test size is T=500 (L1000-T500) for different strategies.

The accuracy when changing the learning and testing data size is high also for all strategies. This is can be easily shown in fig 7.

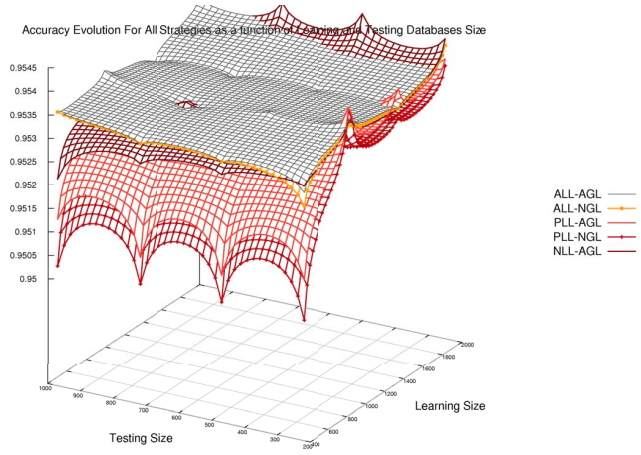

Fig. 14. Accuracy evolution for all strategies for different learning and testing sizes.

Moreover, the loss ratio in accuracy of the predictive local learning is not important (figure 15).

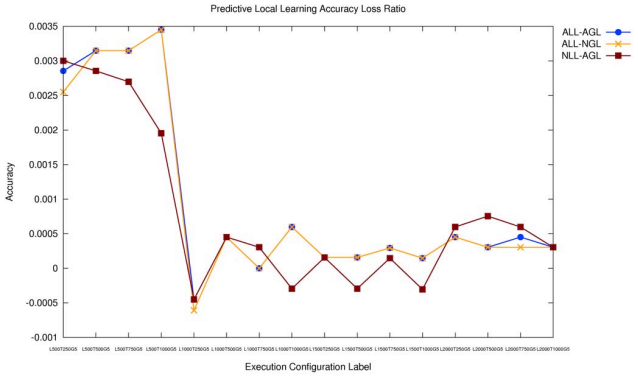

Fig. 15. PLL accuracy loss ratio for the Kyoto data set in the case of No-limit mode.

We notice also that the difference in computational time between the 5 strategies is not important as we increase the number of support vectors (see figure 16).

- With-limit Learning case

Figure 17 (table) shows the evaluation measure values. According to this table, we can remark that all strategies give good similar results for all used metrics. For the PLL-NGL strategy, we obtain very good results. Thus, we can again opt for this strategy for alert classification. These results encourage us to direct our effort to an online unsupervised approach.

In addition, the loss ratio in accuracy of the predictive local learning is not significant. This ration varies from -0.007 to +0.027 and the accuracy variation for all strategies (see figure 18).

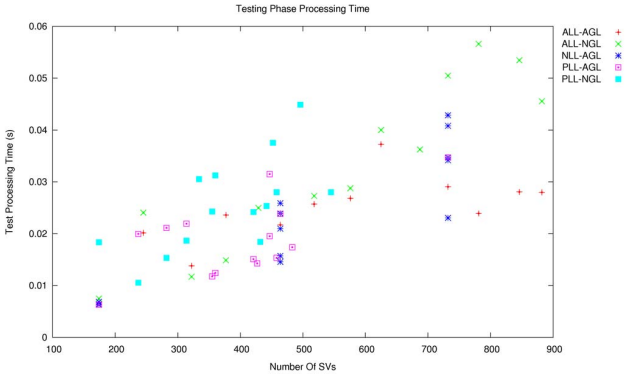

Fig. 16. Testing Phase Processing Time as function of Number of SV for the ISCX data set in the case of No-limit mode.

	ALL-AGL	ALL-NGL	NLL-AGL	PLL-AGL	PLL-NGL
Sensitivity	0.93931	0.93931	0.94223	0.98199	0.99284
Specificity	0.94596	0.94596	0.93928	0.86509	0.85469
FPR	0.05404	0.05404	0.06072	0.13491	0.14531
FDR	0.03200	0.03200	0.03600	0.09400	0.10400
PPV	0.96837	0.96837	0.96398	0.90649	0.89633
NPV	0.89846	0.89846	0.90408	0.97302	0.98951
Accuracy	0.94171	0.94171	0.94114	0.93186	0.93186
MCC	0.87600	0.87600	0.87500	0.86300	0.86600
F2	0.94500	0.94500	0.94600	0.96600	0.97200
F1/2	0.96200	0.96200	0.96000	0.92100	0.91400
F1	0.95400	0.95400	0.95300	0.94300	0.94200

Fig. 17. Results for the ISCX data set in the case of With-limit mode where learning size is L=1000 and Test size is T=500 (L1000-T500) for different strategies.

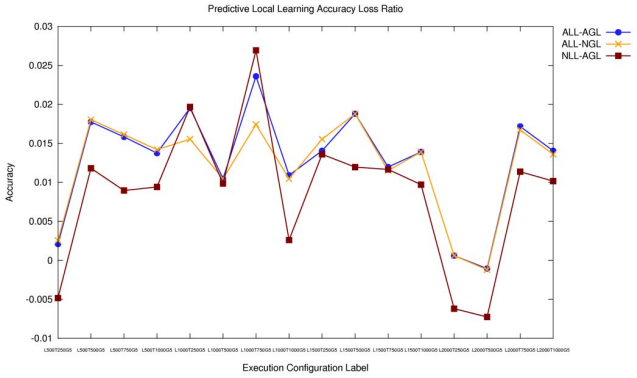

Fig. 18. PLL accuracy loss ratio for the Kyoto data set in the case of With-limit mode.

4) Discussion and Statements:

Our main concluded statements are the following:

- The performance results for both cases are not significantly different (based on the obtained measure values). Thus, taken only a small part of the SVs does not affect the classification result.
- The processing time for with-limit case is lowest that without limit. This gain becomes more significant when the data size becomes large.
- By comparing the small difference in performance and the important gain in processing time we can conclude that the limit case is more better for us in our further investigation.
- Concerning the different algorithms, the performance study shows that the PLL-NGL strategy gives an excellent results. Its loss ratio in accuracy is not significant. Furthermore, it outperforms in some cases the other strategies. We conclude that PLL-NGL with the limit-mode is an excellent strategy for the alert classification since it satisfies exactly our requirements in high speed network in term of efficiency, real time and adaptability.

IV. Conclusion and Future Work

In this paper we have proposed an efficient real time adaptive intrusion detection alert classifier for high speed networks. The classifier is based on an online self-trained support vector machines (SVM). We evaluate our classifier against three different data sets. The performance study shows that the PLL-NGL strategy is a good candidate that cope with our high speed network constraints. It represents a good tradeoff between accuracy and time processing. Furthermore, it does not require a Human involvement witch makes it an efficient and a cost-effective solution for the narrow context of high speed network.

These results encourage us to investigate as a future work other machine learning algorithms especially in an online and self-training configuration. This is will lead to fix the strength and weakness of each approach in the context of high speed network.

V. Acknowledgements

This paper is a partial result of a research project granted by King Abdul Aziz City for Sciences and Technology (KACST), Riyadh, Kingdom of Saudi Arabia, under grant number 08-INF36-8

References

[1] C. V. Zhou, C. Leckie, and S. Karunasekera, "A survey of coordinated attacks and collaborative intrusion detection," *Computers & Security*, vol. 29, no. 1, 2010.

[2] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, and W.-Y. Lin, "Intrusion detection by machine learning: A review," *Expert Systems with Applications*, vol. 36, no. 10, pp. 11 994 – 12 000, 2009.

[3] K. Julisch, "Clustering intrusion detection alarms to support root cause analysis," *ACM Trans. Inf. Syst. Secur.*, vol. 6, no. 4, pp. 443–471, 2003.

[4] F. Cuppens, "Managing alerts in a multi-intrusion detection environment," in *Proceedings of the 17th Annual Computer Security Applications Conference*, ser. ACSAC '01, 2001.

[5] P. Ning, Y. Cui, and D. S. Reeves, "Constructing attack scenarios through correlation of intrusion alerts," in *Proceedings of the 9th ACM conference on Computer and communications security*, 2002, pp. 245–254.

[6] P. Ning, Y. Cui, D. S. Reeves, and D. Xu, "Techniques and tools for analyzing intrusion alerts," vol. 7, no. 2, 2004, pp. 274–318.

[7] S. Zanero and S. M. Savaresi, "Unsupervised learning techniques for an intrusion detection system," in *Proceedings of the 2004 ACM symposium on Applied computing*, 2004, pp. 412–419.

[8] T. Pietraszek and A. Tanner, "Data mining and machine learning-towards reducing false positives in intrusion detection," *Inf. Secur. Tech. Rep.*, vol. 10, no. 3, pp. 169–183, 2005.

[9] P. Laskov, P. Dssel, C. Schfer, and K. Rieck, "Learning intrusion detection: supervised or unsupervised," in *IMAGE ANALYSIS AND PROCESSING, PROC. OF 13TH ICIAP CONFERENCE.*, 2005, pp. 50–57.

[10] A. Hofmann and B. Sick, "Online intrusion alert aggregation with generative data stream modeling," *IEEE Transactions on Dependable and Secure Computing*, vol. 8, no. 2, pp. 282–294, 2011.

[11] S.-J. Horng, M.-Y. Su, Y.-H. Chen, T.-W. Kao, R.-J. Chen, J.-L. Lai, and C. D. Perkasa, "A novel intrusion detection system based on hierarchical clustering and support vector machines," *Expert Syst. Appl.*, vol. 38, no. 1, pp. 306–313, 2011.

[12] D. Fisch, A. Hofmann, and B. Sick, "On the versatility of radial basis function neural networks: A case study in the field of intrusion detection," *Information Sciences*, vol. 180, no. 12, pp. 2421–2439, 2010.

[13] R. Smith, N. Japkowicz, M. Dondo, and P. Mason, "Using unsupervised learning for network alert correlation," in *21st conference on Advances in artificial intelligence*, ser. Canadian AI'08, 2008, pp. 308–319.

[14] L. Khan, M. Awad, and B. Thuraisingham, "A new intrusion detection system using support vector machines and hierarchical clustering," *The VLDB Journal*, vol. 16, no. 4, pp. 507–521, 2007.

[15] C. Cortes and V. Vapnik, "Support-vector networks, machine learning," *Computers and Security*, vol. 20, no. 3, pp. 273–297, 1995.

[16] S. L. H. Byun, "A survey on pattern recognition applications of support vector machines," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 17, pp. 459–486, 2003.

[17] W. Hu, "Robust support vector machines for anomaly detection," in *In Proc. 2003 International Conference on Machine Learning and Applications (ICMLA03*, 2003, pp. 23–24.

[18] P. John, "Fast training of support vector machines using sequential minimal optimization," in *In Advances in Kernel Methods: Support Vector Learning.*, 1998.

[19] G. W. D. Sculley, "Relaxed online svms for spam filtering," in *SIGIR*, 2007.

[20] J. Kivinen, A. J. Smola, and R. C. Williamson, "Online learning with kernels," vol. 52, no. 8, 2004, pp. 2165–2176.

[21] W. Q. Lau KW, "Online training of support vector machine," vol. 36, no. 8, 2003, pp. 1913–1920.

[22] R. S, "Incremental learning with support vector machines," vol. 21, 2001, pp. 641–642.

[23] G. Cauwenberghs and T. Poggio, "Incremental and decremental support vector machine learning," vol. 22, 2000, pp. 409–415.

[24] C. M. . Z. A. Chapelle, O., "A continuation method for semisupervised svms," *23rd International Conference on Machine Learning, ICML06, Pittsburgh, USA*, 2006.

[25] X. Zhu, "Semi-supervised learning literature survey," *Technical Report 1530, Department of Computer Sciences, University of Wisconsin, Madison*, 2005.

[26] W. L. A. L. P. Salvatore Stolfo, Wei Fan and P. K. Chan, "Cost-based modeling for fraud and intrusion detection: Results from the jam project," *disex*, vol. 2, p. 1130, 2000.

[27] R. L. et al., "Evaluating intrusion detection systems: the 1998 darpa off-line intrusion detection evaluation," *DARPA Information Survivability Conference and Exposition, 2000. DISCEX '00. Proceedings*, vol. 2, pp. 12–26, 2000.

[28] W. L. M. Tavallaee, E. Bagheri and A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," *IEEE Symposium. Computational Intelligence for Security and Defense Applications, CISDA*, vol. 2, pp. 1–6, 2009.

[29] J. Mchugh, "Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory," *ACM Trans. Inf. Syst. Secur*, vol. 3, pp. 262–294, 2000.

[30] M. E. D. I. Hiroki Takakura, Yasuo Okabe and K. N. J. Song, "Statistical analysis of honeypot data and building of kyoto 2006+ dataset for nids evaluation," *Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS '11), New York, NY, USA*, 2011.

[31] A. A. G. A. S. Hadi Shiravi, Mahbod Tavallaee, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Computers and Security*, vol. 31, no. 3, pp. 357–374, 2012.