http://dx.doi.org/10.5755/j01.eee.20.7.8025

Improving Intrusion Detection with Adaptive Support Vector Machines

N. Macek[1], B. Dordevic[2], V. Timcenko[2], M. Bojovic[3], M. Milosavljevic[4]
[1]School of Electrical Engineering and Computing of Applied Studies,
Vojvode Stepe 283, 11000 Belgrade, Serbia
[2]Institute Mihailo Pupin d.o.o.,
Volgina 15, 11060 Belgrade, Serbia
[3]IT011,
Omladinskih brigada 88, 11070 Belgrade, Serbia
[4]Singidunum University, Danijelova 32,
11000 Belgrade, Serbia
macek.nemanja@gmail.com

***Abstract*—The research topic that this paper is focused on is intrusion detection in critical network infrastructures, where discrimination of normal activity can be easily corrected, but no intrusions should remain undetected. The intrusion detection system presented in this paper is based on support vector machines that classify unknown data instances according both to the feature values and weight factors that represent importance of features towards the classification. The major contribution of the proposed model is significantly decreased false negative rate, even for the minor categories that have a very few instances in the training set, indicating that the proposed model is suitable for aforementioned environments.**

***Index Terms*—Intrusion detection, machine learning, support vector machines, false negative rate.**

I. Introduction

Machine learning based intrusion detection system (IDS) learns to classify events based on knowledge which is obtained from the training set. Training set for the network IDS is the set of network connection records formed from a raw network data. Each record is described with a set of features and is labelled as member of appropriate class. After the training, the system is able to predict and classify previously unknown network traffic as normal or malicious. Previous researches have shown that IDS systems based on some machine learning algorithms can be computationally expensive if they are trained with a set that has a large number of features. As a solution, many authors have proposed feature reduction methods that select features important for classification and train the classifiers only with these features. Although these systems operate at much higher speed, it is easy to notice they are just a compromise between accuracy and speed and that they are not suitable for critical environments where no attack should pass undetected. Another downgrade is that all the features in the remaining set are considered equally, as if they contribute with the same amount of knowledge to the classification. Support vector machines (SVM) solve the first problem. The model proposed in this paper solves the second one.

In this paper, we proposed a SVM IDS model that classifies unknown data instances according to both feature values and the contribution of each feature towards the classification. Feature weights are calculated by scaling the accuracy change of a classifier from which one attribute is removed. Comparing to unmodified SVM classifier and classifiers trained with reduced feature sets, the proposed model significantly reduces false negative rate and increases detection rate for all attack categories, including minor ones.

II. Related Work

Although some authors have made some efforts into the similar research area which this paper deals with, their research ended with a simulation via feature reduction. For example, Yao et al. presented a good feature weight calculation method based on rough sets in [1], and perform an approximation of modified kernel function by cutting off features with low weights. Although testing results of their classifier indicate high detection rates and low false negative rates, the ability to detect specific categories like User to Root (U2R) and Remote to Local (R2L) remains unknown. This area is explored in approximation presented in [2]; authors report high detection rates (over 99 %) for all five categories of classifiers trained and tested with the smaller subsets. However, comparing the results reported in the literature is sometimes impossible due to lack of information in papers and some methodological factors – for example, how the training and testing subsets are created. Although this does not have an impact on normal traffic or probing attacks detection, it is a fundamental issue with the minor categories – U2R and R2L.

III. IDS Evaluation

True positive (TP) denotes if an IDS has correctly classified an intrusion as an intrusion. False positive (FP) denotes if an IDS has incorrectly classified normal data as an

Manuscript received December 5, 2013; accepted May 14, 2014.

intrusion. True negative (TN) denotes if an IDS has correctly classified normal data as normal. False negative (FN) denotes if an IDS has incorrectly classified an intrusion as normal. Equations (1), (2) and (3) define true positive rate (TPR), which is also referred to as sensitivity, false negative rate (FNR) and accuracy (A):

$$TPR = TP / (TP + FN), \tag{1}$$

$$FNR = FN / (TP + FN), \tag{2}$$

$$A = (TP + TN) / (TP + TN + FP + FN). \tag{3}$$

The Knowledge Discovery and Data Mining (KDD) Cup ’99 IDS evaluation data set [3] is derived from the data gathered at MIT Lincoln Laboratory under DARPA sponsorship with the purpose to evaluate IDS. Data is collected from a network that simulates a typical U.S. Air Force Local Area Network (LAN) attacked with various types of intrusions. There are three partitions of the KDD Cup ’99 data available: a full training set (4,898,431 instances), a 10 % version of training set, and a test set (311,029 instances), which includes 17 new attacks (attacks that are not included in the training sets). All intrusions are grouped into four categories, according to the taxonomy of Kendall [4]:

– Probing – scanning a network of computers to gather information or find known vulnerabilities;

– Denial of Service (DoS) – causing the unavailability of resources;

– User to Root (U2R) – exploiting vulnerabilities to gain root access to the system;

– Remote to Local (R2L) – obtaining access to remote system without having a user account;

Proportions of attack instances in KDD Cup ’99 dataset are given in Table I.

TABLE I. PROPORTIONS OF INSTANCES IN KDD CUP ’99 DATASET.

Category	Training set	10 % train set	Test set
Normal	972,780 (19.86 %)	97,278 (19.69 %)	60,593 (19.48 %)
Probing	41,102 (0.84 %)	4,107 (0.83 %)	4166 (1.34 %)
DoS	3,883,370 (79.30 %)	391,458 (79.24 %)	229,853 (73.90 %)
U2R	52 (~0.00 %)	52 (0.01 %)	70 (0.02 %)
R2L	1,126 (0.02 %)	1,126 (0.23 %)	16,347 (5.26 %)

The KDD Cup ’99 network traffic data is connection based. Each data record, described with 7 categorical and 34 numerical attributes, corresponds to a connection between two IP addresses. In addition, a label is provided, indicating whether the record is normal or it belongs to one of the four attack categories [5]. Categorical features that have two possible values (e.g., logged-in or land) are represented by a binary entry with the value of 0 or 1. During the preprocessing phase, categorical features with more than two possible values (e.g., protocol or service) are transformed into a set of binary features. For example, feature that has possible values tcp, udp and icmp is mapped into three features: (0, 0, 1), (0, 1, 0) and (1, 0, 0).

IV. Support Vector Machines

Support vector machines [6], [7] are supervised learning algorithms that they learn very effectively from high dimensional data [8], which eliminates the need for feature reduction. The basic idea is to find a hyper-plane which separates the *d*-dimensional data perfectly into two classes. If the data is often not linearly separable, SVM’s introduce the kernel induced feature space which casts the data into a higher dimensional space where it is separable.

The data for a two class learning problem consists of *n* objects x_i (I = 1, … n) labelled with one of two labels y_i corresponding to the two classes: +1 (positive class) or -1 (negative class). Let **x** denotes a vector with components x_i, **w** the weight vector and *b* the bias which translates hyper-plane from the origin. A linear classifier is based on a linear discriminant function *f*(**x**)

$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b = \sum w_i x_i + b. \tag{4}$$

The hyper-plane

$$\{\mathbf{x} : f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b = 0\}, \tag{5}$$

divides the space in two, while the sign of function *f*(**x**) denotes the side of the hyper-plane (as shown in the Fig. 1).

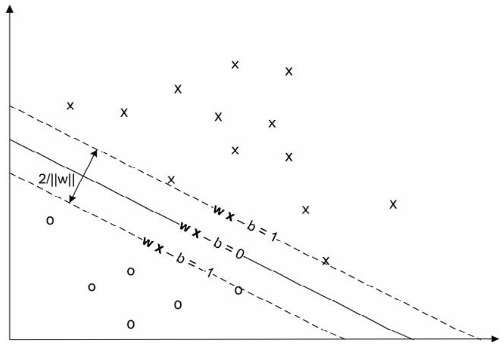

Fig. 1. Maximum margin hyper-plane division of the feature space for two class problem.

Suppose the weight vector can be expressed as a linear combination of the training examples, i.e. $\mathbf{w} = \sum wa_i\, x_i$. This is known as dual representation of decision boundary. The discriminant function takes the form

$$f(\mathbf{x}) = \sum \sim_i x_i^T \mathbf{x} + b. \tag{6}$$

Let *k*(**x**, **x’**) denote the kernel function and Φ its effect on an object. In the feature space, (6) takes the form

$$f(\mathbf{x}) = \sum \sim_i k(\mathbf{x}, \mathbf{x}_i) + b = \sum \sim_i \Phi(x_i)^T \Phi(\mathbf{x}) + b. \tag{7}$$

As the feature space may be high dimensional, the kernel function must be computed efficiently. The maximum margin classifier is the discriminant function that maximizes the geometric margin 1/‖**w**‖, where ‖**w**‖ is the norm of the weight vector. This leads to constrained optimization problem: minimize ½ $\|\mathbf{w}\|^2$, subject to

$$y_i (\mathbf{w}^T x_i + b) \ge 1, \tag{8}$$

where $i = 1, \dots, n$. The constraints in this formulation ensure that the maximum margin classier classifies each example correctly, which is possible if the data is linearly separable. In practice, data is often not linearly separable; and even if it is, a greater margin can be achieved by allowing the classier to misclassify some points. To allow misclassification, (8) is modified with the slack variables ξ_i, as shown in (9). Slack variables allow examples to be in the margin error ($0 \leq \xi_i \leq 1$) or to be misclassified ($\xi_i > 1$). The bound of misclassified examples is $\sum \xi_i$

$$y_i (\mathbf{w}^T x_i + b) \geq 1 - \P_i, \tag{9}$$

where $i = 1, \dots, n$. The constant C > 0 sets the relative importance of maximizing the margin and minimizing the amount of slack. This formulation is called the soft-margin SVM, and was introduced in [7]. The optimization problem for soft-margin classifier becomes minimizing expression (10) subject to (9)

$$½ \| \mathbf{w}\|^2 + C \sum \P_i. \tag{10}$$

Using the Lagrange multipliers (dual formulation), the optimization problems now becomes maximizing

$$\sum \sim_i - ½ \sum_i \sum_j y_i y_j \sim_i \sim_j \mathbf{x}_i{}^T \mathbf{x}_j. \tag{11}$$

This formulation leads to an expansion of weight factor

$$\mathbf{w} = \sum y_i \sim_i \mathbf{x}_i = 0. \tag{12}$$

The examples x_i for which $\alpha_i > 0$ are points that are on or within the margin: these points are called support vectors. The expansion in terms of the support vectors is often sparse, and the level of sparsity (fraction of the data serving as support vectors) is an upper bound on the error rate of the classifier [9].

The dual formulation of the SVM optimization problem depends on the data only through dot products. The dot product can therefore be replaced with a non-linear kernel function, thereby performing large margin separation in the feature-space of the kernel. The SVM optimization problem was traditionally solved in the dual formulation, and only recently it was shown that the primal formulation can lead to efficient kernel-based learning [10].

If compared to polynomial, Radial Basis Function (RBF, also mentioned as a Gaussian in the literature) kernel has fewer numerical difficulties. One key point is that values range between 0 and 1, in contrast to polynomial kernels of which kernel values may go to infinity or zero while the degree is large. The performance of intrusion detection that use support vector machines with different kernels is compared in [11]. Their experiment proved that that SVM that uses RBF kernel gives the best performance of an SVM based IDS system.

V. CONSTRUCTING THE ADAPTIVE SVM MODEL

Systems based on feature weight calculation have been simulated with the simple feature reduction that cuts off features with low weights [1], [12], experimentally tested and provided high detection rate. Lack of approximation is a small set of features extracted for U2R and R2L categories that contain the most dangerous attacks and have the least instances in the training set.

The model presented in this paper does not perform a simulation. A model is trained with the following set of instances: {[*label*] 1:$i_1 \cdot w_1$ 2: $i_2 \cdot w_2$ 3: ... 41: $i_{41} \cdot w_{41}$}, where i_1, i_2, …, i_{41} denote feature values and w_1, w_2, …, w_{41} weights assigned to corresponding features.

There are various methods to calculate weight factors. One method is based on rough set theory, as described in [1]. The method based on achieving weights directly from support vector decision function is presented in [2]. Although obtaining this information is possible only if a trained L2-loss linear model is used [13], authors do not provide sufficient information needed for further discussion.

The proposed algorithm for feature weight calculation is derived from a feature reduction algorithm presented in [2]. Feature weights are calculated according to the accuracy change of a classifier trained with a set from which one feature was removed. Let *a* denotes the accuracy of classifier trained with all features, and let a_i denotes the accuracy of a classifier trained with all features except feature *i*. Accuracy change for that classifier Δa_i is given with the expression

$$\Delta a_i = a - a_i. \tag{13}$$

The smallest and the largest accuracy changes (Δa_{min} and Δa_{max}) are defined with (14) and (15):

$$\Delta a_{min} = min(\Delta a_i), \tag{14}$$

$$\Delta a_{max} = max(\Delta a_i), \tag{15}$$

where $i = 1, \dots 41$. Feature weight w_i of the feature *i* is calculated with (16) and scaled to a range [0, 1]

$$w_i = (\Delta a_i - \Delta a_{min}) \;/\; (\Delta a_{max} - \Delta a_{min}). \tag{16}$$

VI. EXPERIMENTS

Experiments were conducted with LibSVM 3.16 using generic RBF kernel. After preprocessing the dataset (linear scaling of numerical attributes and conversion of categorical attributes to binary) and importing it into LibSVM format, the experiment has been conducted as follows:

– Determine optimal hyper-parameters (soft margin constant *C* and Gaussian kernel parameter *γ*) of the model using v-fold cross validation and grid search; optimal pair (*C*, *γ*) provides a classifier that can accurately predict unknown data;

– Train the SVM classifier with a 10 % training set;

– Calculate feature weights wi (scaled to [0, 1]);

– Scale the training and test set with feature weights;

– Find the optimal hyper-parameters of the new model;

– Train the SVM classifier with scaled training set;

– Test the new model with three randomly generated test sets (50,000 instances each).

Feature weights calculation based on classifier accuracy change required 41 additional experiments in which features were removed one at a time. Detection rates and false negative rates of the proposed model are measured and compared to original model. Performance of the original model (SVM) is given in Table II, performance of the proposed model (AC+SVM) is given in Table III, and the comparison of classifiers is given in Table IV.

TABLE II. PERFORMANCE OF THE ORIGINAL SVM CLASSIFIER.

	Test set 1	Test set 2	Test set 3
Testing time (sec)	11.41	9.62	10.07
Accuracy (%)	98.15 %	97.99 %	97.55 %
FN (%)	1.24 %	1.25 %	1.62 %

TABLE III. PERFORMANCE OF THE MODEL EXPANDED WITH FEATURE WEIGHTS CALCULATED FROM ACCURACY CHANGES (AC + SVM).

	Test set 1	Test set 2	Test set 3
Testing time (sec)	12.5	9.61	9.68
Accuracy (%)	99.12 %	99.16 %	99.18 %
FN (%)	0.53 %	0.42 %	0.38 %

TABLE IV. IDS PERFORMANCE COMPARISON. VALUES IN THE TABLE ARE AVERAGES FOR ALL THREE TESTING SETS.

	SVM	AC+SVM
Testing time (sec)	10.6	10.6
Accuracy (%)	97.90 %	99.15 %
FN (%)	1.37 %	0.44 %

VII. COMPARISON TO FEATURE REDUCTION-BASED SVM

To prove the benefits of the model presented in this paper, its performance is compared to the performance of classifiers trained with reduced feature sets. Reduced sets are generated with the empirical method presented in [2], F-score ranking method presented in [14] and rough set feature reduction algorithm presented in [1].

TABLE V. AVERAGE PERFORMANCE OF THE FEATURE REDUCTION BASED CLASSIFIERS.

	Empirical	F-score	Rough set
Features	31	23	26
Testing time (sec)	8.93	6.86	8.48
Accuracy (%)	97.55 %	97.73 %	97.02 %
FN (%)	1.29 %	1.62 %	1.66 %

TABLE VI. IDS PERFORMANCE COMPARISON. VALUES IN THE TABLE ARE AVERAGES FOR ALL THREE TESTING SETS.

Model	Accuracy (%)	FN (%)
Proposed model (AC+SVM)	99.15 %	0.44 %
Empirical feature reduction	97.55 %	1.29 %
F-score feature reduction	97.73 %	1.62 %
Rough set feature reduction	97.02 %	1.66 %

The aforementioned methods generate reduced feature sets with 31, 23 and 26 features upon which classifier models are built. As with the proposed model, reduced set classifiers have been tested on three test sets with 50,000 randomly selected instances. Average results of feature reduction-based classifiers are given in Table V. The comparison of the proposed model and feature reduced classifiers is given in Table VI.

VIII. CONCLUSIONS

A new SVM based intrusion detection system that classifies unknown data instances according to the feature values and feature weights has been presented in this paper. Model’s performance is compared to original, unmodified SVM classifiers and classifiers based on training sets formed by different feature reduction methods. System is capable to detect even the minor attack categories with high detection accuracy, and false negative rate is significantly decreased.

Although detection accuracy is not a major improvement, significantly reduced false negative rate provides an IDS system with high sensitivity, capable of detecting R2L and U2R attacks, which represent the most dangerous attacks in the training set.

System is capable to self-determine optimal hyper-parameters of the classifier and operates at high speed (one dot product per classification).

In the further researches we will analyse the multi-class SVMs [15] expanded by feature vectors and form a system capable to readjust feature weights and optimal model hyper-parameters according to changes in the environment where the system is deployed.

REFERENCES

[1] J. Yao, S. Zhao, L. Fan, “An Enhanced Support Vector Machine Model for Intrusion Detection”, Department of Computer Science, University of Regina, Canada, 2010.

[2] S. Mukkamala, A. H. Sung, “Feature Selection for Intrusion Detection with Neural Networks and Support Vector Machines”, *Journal of the Transportation Research Board of the National Academies*, vol. 1822, pp. 33–39, 2003. [Online]. Available: http://dx.doi.org/10.3141/1822-05

[3] *KDD Cup 1999 Data*, UCI KDD Archive, University of California, Irvine, http://kdd.ics.uci.edu/databases/kddcup99/

[4] K. Kendall, “A Database of Computer Attacks for the Evaluation of Intrusion Detection Systems”, M.S. thesis, Massachusetts Institute of Technology, USA, 1999.

[5] Y. Liao, V. R. Vemuri, *Enhancing Computer Security with Smart Technology*, Auerbach Publications, Taylor & Francis Group, USA, 2006, ch. 5.

[6] M. Burgess, “Computer immunology”, in *1998 Proc. of the 12th USENIX conference on System administration*, pp. 283–298.

[7] C. Cortes, V. Vapnik, “Support-vector networks”, *Machine Learning*, pp. 273–297, 2005.

[8] B. E. Boser, I. M, Guyon, V. N. Vapnik, “A training algorithm for optimal margin classifiers”, in *1992 Proc. of the fifth annual workshop on Computational learning theory*, pp. 144–152. [Online]. Available: http://dx.doi.org/10.1145/130385.130401

[9] B. Scholkopf, A. Smola, *Learning with Kernels*, MIT Press, Cambridge, MA, USA, 2002.

[10] L. Bottou, O. Chapelle, D. DeCoste, J. Weston, *Large Scale Kernel Machines*. MIT Press, Cambridge, MA, 2007.

[11] V. Das, V. Pathak, S. Sharma, R. Sreevathsan, MVVNS Srikanth, G. Kumar, “Network intrusion detection system based on machine learning algorithms”, *Int. Journal of Computer Science & Information Technology*, vol. 2, no. 6, pp 138–151, 2010. [Online]. Available: http://dx.doi.org/10.5121/ijcsit.2010.2613

[12] S. Mukkamala, A. H. Sung, “Artificial Intelligent Techniques for Intrusion Detection”, in *Proc. of 2003 IEEE Int. Conf. Systems, Man, and Cybernetics*, pp. 1266–1272.

[13] I. Guyon, J. Weston, S. Barnhill, V. Vapnik, “Gene selection for cancer classification using support vector machines“, *Machine Learning*, pp. 389–422, 2002. [Online]. Available: http://dx.doi.org/10.1023/A:1012487302797

[14] Y.-W. Chang, C-J. Lin, “Feature Ranking Using Linear SVM”, in *Proc. 2008 JMLR Conf. Proc., (WCCI 2008) workshop on causality*, pp. 53–64.

[15] K. Crammer, Y. Singer, “On the algorithmic implementation of multiclass kernel-based vector machines“, *J. Mach. Learn. Res.*, pp. 265–292, 2002.