SERBIAN JOURNAL OF ELECTRICAL ENGINEERING

Reducing U2R and R2L Category False Negative Rates with Support Vector Machines

Nemanja Maček[1], Milan Milosavljević[2,3]

Abstract: The KDD Cup '99 is commonly used dataset for training and testing IDS machine learning algorithms. Some of the major downsides of the dataset are the distribution and the proportions of U2R and R2L instances, which represent the most dangerous attack types, as well as the existence of R2L attack instances identical to normal traffic. This enforces minor category detection complexity and causes problems while building a machine learning model capable of detecting these attacks with sufficiently low false negative rate. This paper presents a new support vector machine based intrusion detection system that classifies unknown data instances according both to the feature values and weight factors that represent importance of features towards the classification. Increased detection rate and significantly decreased false negative rate for U2R and R2L categories, that have a very few instances in the training set, have been empirically proven.

Keywords: Intrusion detection, Machine learning, Support Vector Machines, False Negatives.

1 Introduction

Intrusion into computer system or network is defined as any unauthorized attempt to access, modify or destroy information, or render the system unreliable or unusable [1]. Intrusion detection system (IDS) detects this type of activities by monitoring events and searching for signatures or behavior that indicate intrusions [2]. Common commercial IDS system identifies attacks based on their signatures. IDS systems that perform signature detection have several drawbacks. One major drawback is large number of missed alarms, caused by the inability to detect attacks that are not found in the signature database, or, at least slight derivations of known attacks. These deficiencies can be remedied by applying machine learning methods.

Machine learning based IDS system learns to classify events based on knowledge which is obtained from the training set. Training set for the network

[1]School of Electrical Engineering and Computing of Applied Studies, Vojvode Stepe 283, 11000 Belgrade, Serbia; E-mail: macek.nemanja@gmail.com
[2]Singidunum University, Danijelova 32, 11000 Belgrade, Serbia; E-mail: mmilosavljevic@singidunum.ac.rs
[3]School of Electrical Engineering, Bulevar Kralja Aleksandra 73, 11000 Belgrade, Serbia

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 labeled as member of appropriate class. Machine learning algorithm is trained using this set. After the training, the system is able to predict and classify previously unknown 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 drawback 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 solve the first problem. The model proposed in this paper solves the second one and increases minor category detection accuracy.

Support vector machines are suitable classifiers for IDS systems [3]. They operate fast and do not require feature reduction of the training set, as they achieve high learning efficiency from high-dimensional data sets. Support vector machines are relatively insensitive to the number of data points and the dimensionality of the feature space [4], so they can potentially learn a larger set of patterns and thus be able to scale better than neural networks. In this paper, we proposed a SVM IDS model that classifies unknown data instances according to both feature values and feature weight factors (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 and by using F-scores of features. Comparing to unmodified SVM classifier, the proposed model significantly reduces false negative rate and increases detection rate for minor categories – U2R and R2L.

2 Literature Overview

2.1 Support vector machines in intrusion detection

Support vectors are binary classifiers; one SVM is sufficient to classify data as normal or intrusive [5]. Although there are some applications of multi-class SVM reported in the literature, a common approach used to classify data into more than two classes is to combine several two-class SVM, as described in [6]. Five-class taxonomy presented in [7] is based on “winner-takes-all” strategy. Five classifiers are used and each distinguishes one category from the rest of the data.

The detailed analysis of SVM with different kernels and their abilities to successfully detect different attack types is given in [8]. As expected, Gaussian kernel provides the best classification abilities on KDD dataset.

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 [9], and performed 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 U2R and R2L remains unknown. This area is explored in approximation presented in [10]; 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 some times impossible due to lack of information in papers and some methodological factors – for example, how the training and testing subsets are created. Although subset selection does not have an impact on normal traffic or probing attacks detection, it is a fundamental issue with U2R and R2L categories.

2.2 KDD Cup '99 dataset U2R and R2L detection issues

The KDD Cup '99 data set [11] is derived from the data gathered at MIT Lincoln Laboratory under DARPA sponsorship with the purpose to evaluate IDS Systems. There are three partitions of the KDD Cup '99 data available online: 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). 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. Features are grouped into basic features (derived from packet headers without inspecting the payload), content features (used to access the payload of the original TCP packets), time-based traffic features (designed to capture properties that mature over a two second temporal window) and host-based traffic features, which utilize a historical window estimated over the number of connections. In addition, a label is provided, indicating whether the record is normal or intrusive.

All intrusions are grouped into four categories: probing (scanning a network of computers to gather information or find known vulnerabilities), DoS (causing the unavailability of resources), U2R (exploring vulnerabilities to gain root access to the system) and R2L (obtaining access to remote system without having a user account). Proportions of attack instances in KDD Cup '99 dataset are given in **Table 1**.

While the DoS and probing category detection is is frequently analyzed in the literature, the U2R and R2L categories remained open issue in intrusion detection. These categories contain attacks that exploit operating system or software vulnerabilities with a purpose to gain root privileges or remote access.

Although these categories contain the most dangerous attacks (towards confidentiality and integrity) from the KDD dataset, training and testing sets are far too complex for researchers to form a machine learning model which can detect these attacks with high detection rate [12].

Table 1
Proportions of traffic instances in KDD Cup '99 *dataset*

Category	Training set	10% training 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%)	4,166 (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 complexity comes from the distribution and the proportions of U2R and R2L instances and the existence of R2L attack instances too similar or identical to normal traffic [13], which prevents them to be detected successfully and leads to misclassification [14 – 16]. For example, there are only 52 U2R instances in the training and 10% training set and 70 instances in the test set. Test set contains 16,347 R2L instances and 14 attack types, while the training set contains only 1,126 R2L instances and 8 attack types. 8,054 instances of normal traffic are identical to snmp-get-attack instances – 7,273 of those belong to the test set and the remainder in the 10% training set. There are more examples of the proportion and distribution that make accurate classifier model building even more complex. For example, there are only two spy instances and 1,020 warez-client instances, which is 90.59% of all R2L instances in the training set.

Aside from that, some methodological issues, like instance or feature subset selection, can also occur in the researches, resulting in extremely high or low accuracy. If the training and testing subsets are created, the number of U2R and R2L instances in subsets has an impact on classification abilities. To explain this in simple words: the more U2R and R2L instances the training subset has, it is more likely that trained model will detect these categories. If the reduced feature sets are created, a subset of important features for both categories must be provided to the classifier; if the features are selected for the entire set, it is not likely that all features relevant for minor categories will be selected. Due to

reasons mentioned above, not only the different, but also contradictory results have been reported in the literature. For example, detection rate reported in [10] is over 95%, while it is stated in [17] that minor categories cannot be detected. Results like that make some comparisons not realistic or even unavailable.

3 Building the Adaptive Classification Model

3.1 Binary classification with support vector machines

Support vector machines [18, 19] are supervised learning algorithms that they learn very effectively from high dimensional data [3], which eliminates the need for feature reduction. The basic idea is to find a hyper-plane which separates the into two classes; if the data is often not linearly separable, SVM casts the data into a higher dimensional space where it is separable.

The data for a two class learning problem consists of objects x_i labeled with one of two labels y_i corresponding to the two classes: +1 (positive class) or -1 (negative class). Let $\boldsymbol{x}$ denote a vector with components x_i. A linear classifier is based on a linear discriminant function based on the dot product between two vectors:

$$f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b = \sum w_i x_i + b \,. \tag{1}$$

The vector $\boldsymbol{w}$ is the weight vector and b is the bias which translates hyper-plane from the origin. The hyper-plane:

$$\left\{\boldsymbol{x} : f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b = 0\right\} \tag{2}$$

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

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

$$f(\boldsymbol{x}) = \sum \alpha_i x_i{}^{\mathrm{T}}\boldsymbol{x} + b \,. \tag{3}$$

In the feature space, expression (3) takes the form:

$$f(\boldsymbol{x}) = \sum \alpha_i k(\boldsymbol{x}, \boldsymbol{x}_i) + b = \sum \alpha_i \Phi(\boldsymbol{x}_i)^{\mathrm{T}} \Phi(\boldsymbol{x}) + b \,. \tag{4}$$

As the feature space may be high dimensional, the kernel function $k(\boldsymbol{x},\boldsymbol{x}')$ must be computed efficiently.

The maximum margin classifier is the discriminant function that maximizes the geometric margin $1/\|\boldsymbol{w}\|$, where $\|\boldsymbol{w}\|$ is the norm of the weight vector. This is equivalent to minimizing $\|\boldsymbol{w}\|^2$, which leads to constrained optimization problem: minimize $\frac{1}{2}\|\boldsymbol{w}\|^2$, subject to

$$(\forall i = 1,\ldots,n) \quad y_i(\boldsymbol{w}^{\mathrm{T}} x_i + b) \geq 1 \,. \tag{5}$$

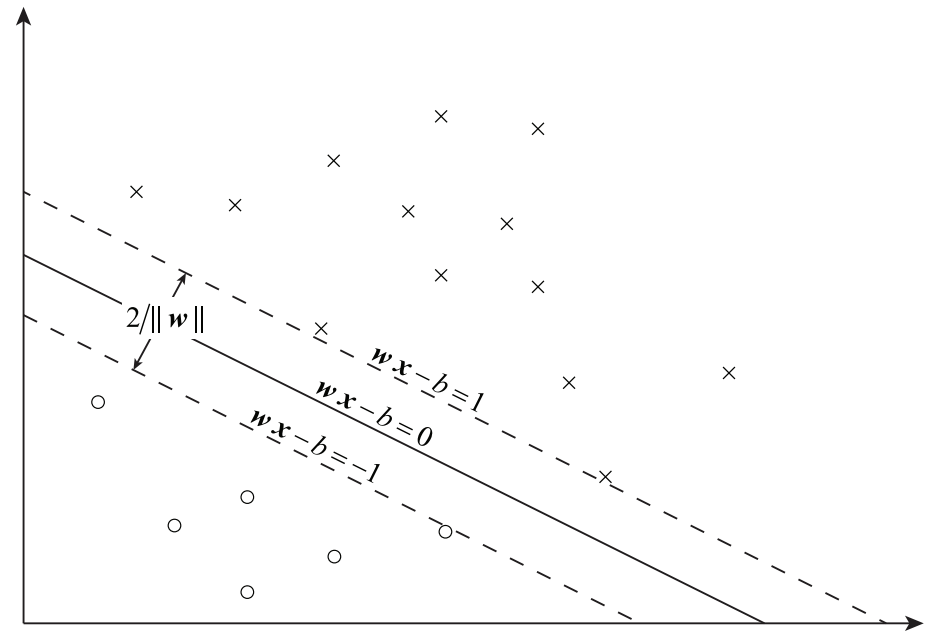

Fig. 1 – *Maximal margin hyper-plane division of the feature space for two class problem.*

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, equation (5) is modified with the slack variables:

$$(\forall i = 1,...,n) \quad y_i(\boldsymbol{w}^{\mathrm{T}} x_i + b) \geq 1 - \xi_i . \tag{6}$$

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 $\sum \xi_i$. 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 [19]. The optimization problem for soft-margin classifier becomes minimizing expression:

$$\tfrac{1}{2}\|\boldsymbol{w}\|^2 + C\sum \xi_i \tag{7}$$

subject to (6). Using the Lagrange multipliers (dual formulation), the optimization problems now becomes maximizing:

$$\sum \alpha_i - \tfrac{1}{2}\sum_i \sum_j y_i y_j \alpha_i \alpha_j \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j . \tag{8}$$

This formulation leads to an expansion of weight factor:

$$\boldsymbol{w} = \sum y_i \alpha_i \boldsymbol{x}_i = 0 . \tag{8}$$

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 [20]. he 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 [21].

3.2 Choosing the kernel

The RBF non-linearly maps samples into a higher dimensional space so it, unlike the linear kernel, can handle the case when the relation between class labels and attributes is nonlinear. If compared to polynomial, RBF 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 [8]. Their experiment proved that that SVM that uses RBF kernel given with

$$k(\boldsymbol{x}, \boldsymbol{x}') = \mathrm{e}^{-\gamma \|\boldsymbol{x} - \boldsymbol{x}'\|^2} \tag{9}$$

provides the best performance for an SVM based IDS system. Parameter γ controls the width of Gaussian and has a similar role as the degree of the polynomial kernel in controlling the classifier's flexibility. According to all mentioned above, RBF kernel is used in this research.

3.3 Expanding the model with feature weights

Feature reduction eliminates unnecessary features from the training set and reduces IDS computational cost. Systems based on feature weight calculation do not eliminate unnecessary features, but determine their importance to intrusion detection. Such systems have been simulated with the simple feature reduction that cuts off features with low weights, experimentally tested and provided high detection rate. Lack of proposed 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 set of instances: {[*label*] 1: $i_1 \cdot w_1$ 2: $i_2 \cdot w_2$ 3: $i_3 \cdot w_3$ ... 41: $i_{41} \cdot w_{41}$},

where i_1, i_2, …, i_{41} denote feature values and w_1, w_2, …, w_{41} weights that have been previously calculated and assigned to features.

There are various methods to calculate weight factors. One of them, proposed by authors, is based on elimination of a feature from training set and measuring classifier performance. Another method is based on rough set theory, as described in [9]. The method based on achieving weights directly from support vector decision function is presented in [10]. Although obtaining this information is possible only if a trained L2-loss linear model is used [22], authors do not provide sufficient information about the model needed for further discussion.

3.4 Calculating feature weights from classifier accuracy change

The proposed algorithm for feature weight calculation is derived from a feature reduction algorithm presented in [10]. Feature weights are calculated according to the accuracy change of a classifier trained with a set from which one feature was removed. Let a denote the accuracy of classifier trained with all features, and let a_i denote the accuracy of a classifier trained with all features except feature i. Accuracy change for that classifier is given with the expression:

$$\Delta a_i = a - a_i . \tag{10}$$

The smallest and the largest accuracy changes are defined with expressions (11) and (12):

$$\Delta a_{\min} = \min\left(\Delta a_i\right), \quad i = 1, \ldots, 41, \tag{11}$$

$$\Delta a_{\max} = \max\left(\Delta a_i\right), \quad i = 1, \ldots, 41 . \tag{12}$$

Feature weight w_i of the feature i is calculated with the equation:

$$w_i = \frac{\Delta a_i - \Delta a_{\min}}{\Delta a_{\max} - \Delta a_{\min}} . \tag{13}$$

All feature weights are scaled to a range [0, 1].

3.5 Calculating feature weights from F-score

F-score is a simple technique which measures the discrimination of two sets of real numbers. Feature's F-score is a ratio of discrimination between the positive and negative sets and discrimination within each of the two sets. The larger the F-score is, the more likely this feature is more discriminative. Let x_k training vectors have n_p and n_n positive and negative instances. Let $\dot{\boldsymbol{x}}_i$, $\dot{\boldsymbol{x}}_i^{(+)}$ and $\dot{\boldsymbol{x}}_i^{(-)}$ denote the average of the i-th feature of the whole, positive, and negative data sets, respectively. Let $x_{k,i}^{(+)}$ denote the i-th feature of the k-th positive instance, and $x_{k,i}^{(-)}$ denote the i-th feature of the k-th negative instance. The F-score of feature i is given as:

$$F(i) = \frac{(\dot{x}_i^{(+)} - \dot{x}_i)^2 + (\dot{x}_i^{(-)} - \dot{x}_i)^2}{\frac{1}{n_p - 1}\sum (x_{k,i}^{(+)} - \dot{x}_i^{(+)})^2 + \frac{1}{n_n - 1}\sum (x_{k,i}^{(-)} - \dot{x}_i^{(-)})^2}. \quad (14)$$

4 Experiments with U2R and R2L Classifiers

Although more recent SVM algorithms have been proposed that directly support multi-class learning, we have used two separate two-class SVM. For each SVM, the training data is divided into two classes; one represents the class itself and the other represents the remaining four classes. For the U2R classifier, positive class contains all U2R attack instances, while the negative contains all normal traffic and all other attack instances. For the R2L classifier, positive class contains all R2L instances, and the negative class all normal traffic and all other attack instances.

Due to a fact that features have different relevance factors towards different category detection, this technique is used in this research to examine the classifier's ability to detect minor categories. Feature weights and model parameters are calculated for each category separately.

Before the experiment with classifiers, the dataset was preprocessed. Preprocessing is done by:

1. conversion of categorical features with *m* values to *m* binary features and
2. linear scaling of feature values to range [0,1], which reduces the number of support vectors and computational resources needed for kernel transformation.

Experiments have been conducted for U2R and R2L categories separately with following steps:

1. reform the dataset by changing positive and negative class labels;
2. determine optimal hyper-parameters (soft margin constant and Gaussian kernel parameter) of the model using v-fold cross validation and grid search; optimal pair (C, γ) provides a classifier that can accurately predict unknown data;
3. train the SVM classifier with a 10% training set;
4. calculate feature weights w_i (scaled to [0,1]);
5. scale the training and test set with feature weights;
6. find the optimal hyper-parameters of the new model;
7. train the SVM classifier with scaled 10% training set;
8. test the new model with the test set.

Feature weights based on classifier accuracy change required 41 additional experiments for each classifier in which features were removed one at a time.

The experiment was conducted with LibSVM 3.16 (using factory provided RBF kernel) and additional Python scripts for F-score calculation on Intel® Core™2 Duo T5500 @ 1.66GHz processor with 8GB of RAM memory running Windows 7 Professional operating system.

4.1 U2R and R2L classifier's performance

Experimental results are given as follows: feature weights calculated using F-score and proposed accuracy change method are partially given in **Table 2**. Performance of the original model (SVM) and the models expanded with feature weights calculated from accuracy changes (AC+SVM) and F-score (FS+SVM) is given in **Table 3**. Difference in false negative rate (Δ FNR) is calculated comparing to the original, non-scaled model.

Table 2
Feature weights calculated for U2R *and* R2L *classifiers.*

	U2R classifier		R2L classifier	
Feature	Acc. change	F-score	Acc. change	F-score
1	0.41283	0.42391	0.42739	0.43104
2	0.18784	0.22891	0.23781	0.25006
3	0.85101	0.85716	0.90247	0.89692
4	0.52036	0.53009	0.49946	0.50829
…	…	…	…	…
39	0.40109	0.40817	0.39572	0.41295
40	0.20016	0.21904	0.23716	0.24297
41	0.15013	0.15773	0.19118	0.18963

Table 3
Performance of the original model and proposed U2R *and* R2L *classifiers.*

Category	Classifier	Accuracy (%)	FNR (%)	Δ FNR (%)
U2R	Original SVM	95.61 %	3.52 %	/
	AC + SVM;	98.55 %	0.84 %	2.68 %
	FS + SVM	98.41 %	0.82 %	2.70 %
R2L	Original SVM	96.02 %	2.86 %	/
	AC + SVM;	98.81 %	0.79 %	2.07 %
	FS + SVM	98.72 %	0.85 %	2.01 %

According to test results, usage of feature weights increases the detection accuracy and reduces false negative rates for minor categories. The average accuracy gain for the U2R classifier is 2.87% and for the R2L classifier is 2.74%. The graph on Fig. 2 illustrates the gain in classification accuracy.

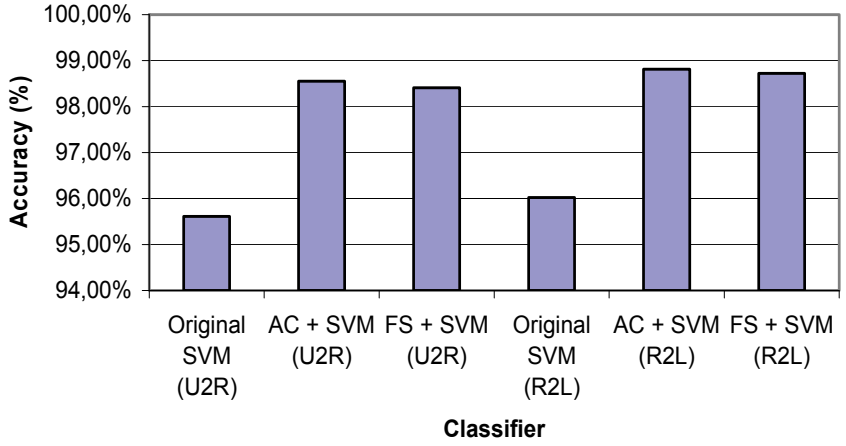

Fig. 2 – *Detection accuracy gain for proposed classifiers*.

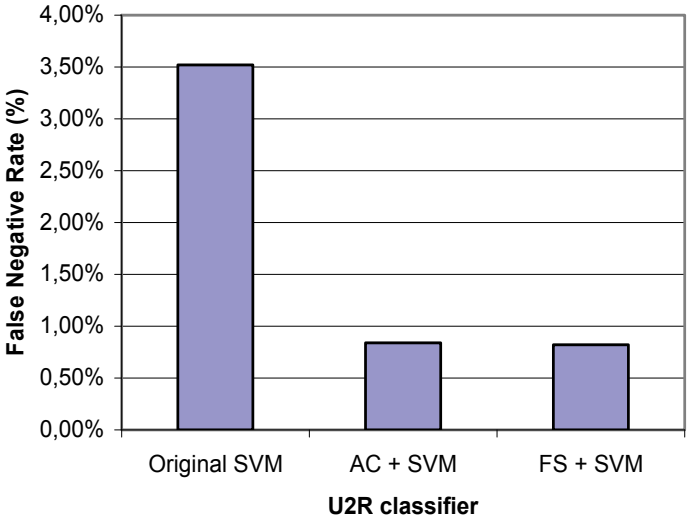

Fig. 3 – *False negative rate reduction for proposed* U2R *classifiers.*

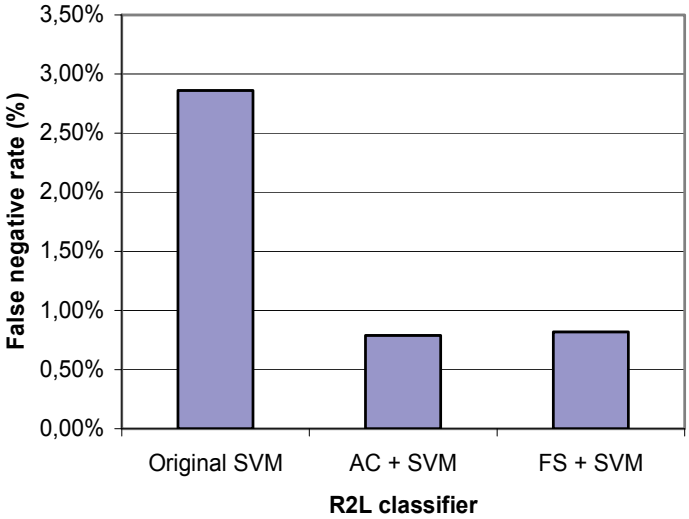

Fig. 4 – *False negative rate reduction for* R2L *classifier.*

The average false negative rate reduction (2.69% for the U2R and 2.06% for the R2L classifier) proves that proposed weight-scaled models are suitable for minor attack category detection. The graphs on Fig. 3 and Fig. 4 illustrate reduction in false negative rates achieved by proposed models.

5 Conclusion and Future Work

A new SVM based intrusion detection system than classifies unknown data instances according to the feature values and feature weights have been presented. Detection accuracy is increased comparing to original SVM classifiers with optimal hyper-parameters, and false negative rate is significantly decreased. System is capable to detect the minor attack categories (R2L and U2R) with high detection accuracy, as it is proven in the experiments. Although detection accuracy is not a major improvement, significantly reduced false negative rate provides an IDS system with high sensitivity. These systems are suitable for the environments where discrimination of normal activity can be easily corrected, but no intrusions should remain undetected.

Approximations of the model based on calculating feature weights using rough set theory [9] or support vector decision function [7] and eliminating features with low weights did not prove sufficiently good to be a minor category classifiers. The major downside of these models is that small number of features that are relevant for U2R and R2L categories was extracted from the set. This leads to a conclusion that IDS with high detection accuracy and low false

negative rate requires all features to be present in the set, and that their impact towards the classification must be calculated with a high precision.

There are two ways to implement the proposed model: to modify the data in the preprocessing phase and to use the modified kernel. The easier way of implementing the model is scaling feature values of unknown data instances in preprocessing phase. If the implementation is done via the kernel, the generic form of new kernel becomes $K(\boldsymbol{wx}_i, \boldsymbol{wx})$, where $\boldsymbol{w}$ is the feature weight vector. Implementation via kernel is more complex, and not as flexible as preprocessing phase implementation.

In the further researches we will analyze the multi-class SVM expanded by feature weight 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.

6 References

[1] J.P. Anderson: Computer Security Threat Monitoring and Surveillance, James P. Anderson Co., Fort Washington, MD, USA, 1980.

[2] P.E. Proctor: The Practical Intrusion Detection Handbook, Prentice Hall, Upper Saddle River, NJ, USA, 2001.

[3] B.E. Boser, I.M, Guyon, V.N. Vapnik: A Training Algorithm for Optimal Margin Classifiers, 5th Annual Workshop on Computational Learning Theory, Pittsburgh, PA, USA, 27 – 29 July 1992, pp. 144 – 152.

[4] T. Joachims: Making Large-scale SVM Learning Practical, LS8-Report, University of Dortmund, Dortmund, Germany, June 1998.

[5] S. Mukkamala, G. Janoski, A. Sung: Intrusion Detection using Neural Networks and Support Vector Machines, International Joint Conference on Neural Networks, Honolulu, HI, USA, 12 – 17 May 2002, Vol. 2, pp. 1702 – 1707.

[6] K.B. Duan, S.S. Keerthi: Which is the Best Multiclass SVM Method? An Empirical Study, Multiple Classifier Systems – Lecture Notes in Computer Science, Vol. 3541, 2005, pp. 278 – 285.

[7] S. Mukkamala, A.H. Sung: Artificial Intelligent Techniques for Intrusion Detection, IEEE International Conference Systems, Man and Cybernetics, Washington, DC, USA, 05 – 08 Oct. 2003, Vol. 2, pp. 1266 – 1271.

[8] V. Das, V. Pathak, S. Sharma, R. Sreevathsan, M.V.V.N.S. Srikanth, G. Kumar: Network Intrusion Detection System based on Machine Learning Algorithms, International Journal of Computer Science and Information Technology, Vol. 2, No. 6, Dec. 2010, pp 138 – 151.

[9] J.T. Yao, S. Zhao, L. Fan: An Enhanced Support Vector Machine Model for Intrusion Detection, Rough Sets and Knowledge Technology – Lecture Notes in Computer Science, Vol. 4062, 2006, pp. 538 – 543.

[10] S. Mukkamala, A.H. Sung: Feature Selection for Intrusion Detection with Neural Networks and Support Vector Machines, Transportation Research Record: Journal of the Transportation Research Board, Vol. 1822, 2003, pp. 33 – 39.

[11] http://kdd.ics.uci.edu/databases/kddcup99/.

[12] M. Sabhnani, G. Serpen: Why Machine Learning Algorithms Fail in Misuse Detection on KDD Intrusion Detection Data Set, Intelligent Data Analysis, Vol. 8, No. 4, Sept. 2004, pp. 403 – 415.

[13] Y. Bouzida, F. Cuppens: Neural Networks vs. Decision Trees for Intrusion Detection, IEEE/IST Workshop on Monitoring, Attack Detection and Mitigation, Tubingen, Germany, 28 – 29 Sept. 2006.

[14] I. Gadaras, L. Mikhailov, S. Lekkas: Generation of Fuzzy Classification Rules Directly from Overlapping Input Partitioning, IEEE International Fuzzy Systems Conference, London, UK, 23 – 27 July 2007, pp. 1 – 6.

[15] C.L. Liu: Partial Discriminative Training for Classification of Overlapping Classes in Document Analysis, International Journal on Document Analysis and Recognition, Vol. 11, No. 2, Nov. 2008, pp. 53 – 65.

[16] S. Raudys: Multiple Classification Systems in the Context of Feature Extraction and Selection, Multiple Classifier Systems – Lecture Notes in Computer Science Vol. 2364, 2002, pp. 27 – 41.

[17] Z.S. Pan, S.C. Chen, G.B. Hu, D.Q. Zhang: Hybrid Neural Network and C4.5 for Misuse Detection, 2nd International Conference on Machine Learning and Cybernetics, Xi'an, China, 02 – 05 Nov. 2003, pp. 2463 – 2467.

[18] M. Burgess: Computer Immunology, 12th USENIX Conference on System Administration, Boston, MA, USA, 06 – 11 Dec. 1998, pp. 283 – 298.

[19] C. Cortes, V. Vapnik: Support-vector Networks, Machine Learning, Vol. 20, No. 3, Sept. 2005, pp. 273 – 297.

[20] B. Scholkopf, A.J. Smola: Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond, MIT Press, Cambridge, MA, USA, 2002.

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

[22] I. Guyon, J. Weston, S. Barnhill, V. Vapnik: Gene Selection for Cancer Classification using Support Vector Machines, Machine Learning, Vol. 46, No. 1-3, Jan. 2002, pp. 389 – 422.