Performance Evaluation of PSVM Using Various Combination of Kernel Function for Intrusion Detection System

Rishabh Jain, Aprajita Pandey, Pramod Duraphe, Bhawna Nigam, and Suresh Jain

***Abstract*—The internet and computer networks are exposed to an increasing number of security threats. With new types of attacks appearing continuously, developing flexible and adaptive security oriented approaches is a severe challenge. In this context, intrusion detection technique is a valuable technology to protect target systems and networks against malicious activities. But this system doesn't provide the required accuracy. Thus to meet this requirement, this paper proposes an intrusion detection system as a model based on Proximal Support Vector Machines (PSVMs) implemented with various combination of basic kernel functions. PSVM is a light and simple modification of support vector machine. We have implemented PSVM for binary classification of intrusion detection data. For experimental training and testing NSL-KDD dataset is preprocessed using Principle Component Analysis technique. Using proposed classification model, we have achieved up to 79% classification accuracy.**

***Index Terms*—Intrusion detection system, kernel function, PCA, proximal support vector machine.**

I. Introduction

With the expansion of computer network, the threat to network system has increased as the various intrusions may cause significant attacks. So to defend from such attacks, lots of security techniques and products, such as firewalls, access control, intrusion detection systems (IDSs), etc., have been developed. Among these security techniques, intrusion detection plays a key role because it is a dynamic defense technique, which is different from earlier static defense techniques including firewalls and access control.

An Intrusion Detection System (IDS) is a device or software application that monitors network or system activities for malicious activities or policy violations and produces reports to a Management Station. Intrusion prevention is the process of performing intrusion detection and attempting to stop detected possible incidents. Thus, intrusion detection techniques based on data mining or machine learning [1], [2] have attracted much attention in recent years.

In data-mining-based IDSs, the process of data analysis and behavior modeling can be automatically carried out and there are also two different processing policies, i.e., misuse detection and anomaly detection. In misuse detection, various standard data mining algorithms, fuzzy logic models [3], and neural networks [4] have been used to classify network intrusions. In anomaly detection, data mining methods based on statistics [5], or clustering techniques [6] are employed to identify attacks as deviation from normal usage.

Despite many advances, existing IDSs still have some challenges in improving their performance to meet the requirements of detecting increasing attacks in fast growing networks. These problems have been looked into and Intrusion Detection system has been proposed using support vector machine [7]. Support Vector Machine (SVM) performs classification by constructing an *N*-dimensional hyperplane that optimally separates the data into two categories. But it also faced problem of handling huge network data with large number of attributes. Thus to reduce the data SVM is employed along with PCA to reduce the dimension of data [8]. Principal component analysis (PCA) involves a mathematical procedure that transforms a number of possibly correlated variables into a smaller number of uncorrelated variables called principal components [9]. Thus it reduces the dimension of network data without any significant loss of data.

This paper proposes an intrusion detection system using Proximal Support Vector machines (PSVMs) with various combinations of kernel functions. In the proposed method, PCA is used to reduce the feature dimension of network data records. PSVM is employed to construct intrusion detection model based on the processed training data. It is a simple classifier than SVM wherein each class of points is assigned to the closer of two parallel planes that are pushed apart as far as possible.

The outline of paper is as follow: section 2 describes Architecture with its components of the proposed model, section 3 describes the experimental evaluation in which the dataset used and process followed is explained, section 4 describes the conclusion derived and the section 5 quotes the references.

II. Architecture of Proposed Intrusion Detection System

This method can be applied in general to any network data on IDS, but here we have applied KDD-Cup99 dataset [10] on this method to illustrate our model. The overall structure and component of PSVM IDS is depicted in Fig. 1. Dataset is given for pre-processing where Z-Score and PCA

Manuscript received July 15, 2012; revised August 18, 2012.

Rishabh Jain is Working for RKD, Indore, Madhya Pradesh, India (e-mail: rkdrishabh@gmail.com).

Aprajita Pandey is working for Accenture India, Pune, India (e-mail: aprajitapandey@gmail.com).

Pramod Duraphe is working for Accenture India, Mumbai, India (e-mail: slasher316@gmail.com).

is applied [11]. This processed data is divided into training and testing file. Training file is used to construct the PSVM model. The accuracy of this model is tested using the testing file.

A. Pre-Processing

Pre-processing involves normalization and dimension reduction of dataset.

1) *Normalization:* It is done as dataset includes values at extremes and the higher extreme values don't overshadow the lower extreme values. There are various methods available for normalization process; they are min-max normalization, z-score normalization and normalization by decimal scaling. We have used z-score normalization which is given by:

$$v' = \frac{v - mean_v}{stand\_dev_v}$$

where mean and standard deviation of each attribute is calculated.

2) *Dimension reduction using PCA:* Dimension reduction is done with the help of principal component analysis. In both neural network and statistics studies, PCA is one of the most fundamental tools of dimensionality reduction for extracting effective features from high-dimensional vectors of input data [12]. In the following section, the application of PCA for dimension reduction of network connection data and its combination with PSVM is discussed.

The network data records can be denoted as

$$x_t = [x_{t1}, x_{t2}, ...x_{tn}]^T \text{ (t=1, 2, …, N), n=41} \quad (1)$$

$$Let \quad x_f = \Phi^T x \qquad \mu = \frac{1}{N}\sum_{t=1}^{N} x_t \quad (2)$$

Then, the covariance matrix of data vectors is

$$C = \frac{1}{N}\sum_{t=1}^{N}(x_t - \mu)(x_t - \mu)^T \quad (3)$$

The principal components are computed by solving the Eigen value problem of

Covariance matrix C:

$$Cv_i = \lambda_i v_i \quad (4)$$

where λ_i (i = 1,2,...,n) are the eigenvalues and v_i (i = 1,2,...,n) are the corresponding eigenvectors.

To represent network data records with low dimensional vectors, we only need to compute the first m eigenvectors which correspond to the m largest eigenvalues.

Let

$$\Phi = [v_1, v_2, ..., v_m],$$

$$\Delta = diag[\lambda_1, \lambda_2, ..., \lambda_m] \quad (5)$$

Then we have

$$C\Phi = \Phi\Delta \quad (6)$$

In PCA , a parameter ν can be introduced to denote the approximation precision of the m largest eigenvectors so that the following relation holds.

$$\sum_{i-1}^{m}\lambda_i / \sum_{i-1}^{n}\lambda_i \geq v \quad (7)$$

Given a precision parameter ν, we can select the number of eigenvectors based on (6) and (7) and the low-dimensional feature vector of a new input data x is determined as follows:

$$x_f = \Phi^T x \quad (8)$$

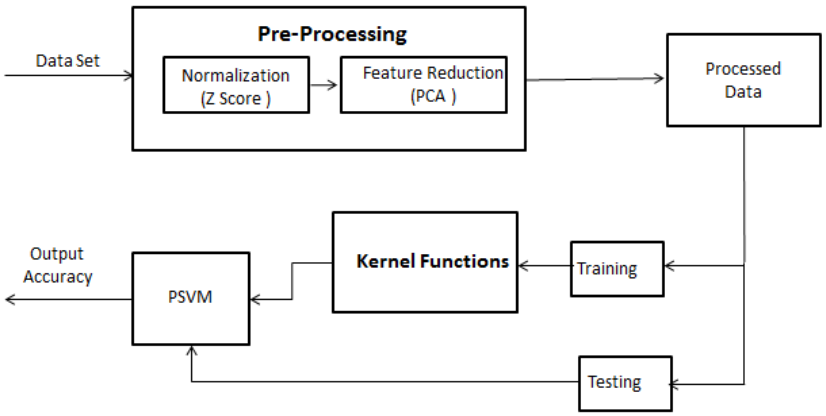

Fig 1. Structure of Intrusion Detection System based on PSVM

B. PSVM Model

The processed data set is divided into training and testing file. Training file is used for the construction of PSVM model using kernel trick. Kernel trick is a method for computing similarity function in the transformed space using the original attribute set. The dot product in transformed space can be expressed in terms of a similarity function in the original space:

$$K(u,v) = \Phi(u)\cdot\Phi(v) = (u\cdot v + 1)^2$$

The similarity function K is known as the kernel function. Thus it is a mathematical trick that allows the Support Vector Machine to perform a 'two-dimensional' - classification of a set of originally one-dimensional data. In general, a kernel function projects data from a low-dimensional space to a space of higher dimension (RBF).

A kernel function K can be expressed as $K(u,v) = \Phi(u)\cdot\Phi(v)$

where *u* and *v* are n-dimensional two vectors. If and only if, for any function *g(x)* such that $\int g(x)^2 dx$ is finite, then

$$\int K(x, y)g(x)g(y)dxdy \geq 0$$

This is called Mercer's Theorem. Kernel functions that satisfy this theorem are called positive definite kernel functions. Examples of such functions are given below:

- Polynomial: $K(x, x_i) = \left(1 + x \cdot x_i^T\right)^d$
- RBF: $K(x, x_i) = e^{-\gamma \| x - x_i \|^2}$
- Sigmoid: $K(x_i, x_j) = \tanh\left(\gamma x_i^T x_j + r\right)$

These are basic kernel functions that are employed in PSVM. It is known that integration is distributive over addition and subtraction. Thus various combination of kernel function can be generated by applying addition and subtraction to basic kernel function that satisfies Mercer's Theorem. Different experiments are done on combination and found that *polynomial-RBF, polynomial - sigmoid, sigmoid - rbf, polynomial + RBF, polynomial + sigmoid, sigmoid + RBF kernel* function performs much better than other basic kernel function.

C. Proximal Support Vector Machine

Standard support vector machines, classify points by assigning them to one of two disjoint half spaces. These half spaces are either in the original input space of the problem for linear classifiers, or in a higher dimensional feature space for nonlinear classifiers .Such SVMs require the solution of either a quadratic or a linear program which require specialized codes. In contrast, Proximal SVM (PSVM) [13] which classifies points depending on proximity to one of two parallel planes that are pushed as far apart as possible. This specific formulation leads to a strongly convex objective function. Strong convexity plays a key role in the simple proximal code. As a result very fast computational speed is obtained. A much simpler classifier (PSVM) is implemented wherein each class of points is assigned to the closer of two parallel planes that are pushed apart as far as possible. This formulation leads to an extremely fast and simple algorithm for generating a linear or nonlinear classifier that is obtained by solving a single system of linear equations.

The point of departure is that, the optimization problem given by

$$\min_{(w,\gamma,y)} \nu e^T y + \frac{1}{2} w^T w$$

Such that $D(Aw - e\gamma) + y \geq e$, $y \geq 0$ (1)

Is replaced by the following problem:

$$\min_{(w,\gamma,y)} \nu \frac{1}{2} \| y \|^2 + \frac{1}{2}(w^T w + \gamma^2)$$

Subject to $D(Aw - e\gamma) + y = e$ (2)

where *e* is a vector of ones

The geometrical interpretation of this formulation is given in Fig (1) [14].

As depicted in Fig 1, the normal to the proximal planes:

$$x^T w - 1 \cdot \gamma = +1$$

$$x^T w - 1 \cdot \gamma = -1 \quad (3)$$

which are proximal to points belonging to the sets *A*+ and *A*− respectively. The error variable *y* in (2) is a measure of the distance from the plane $x^T w - 1 \cdot \gamma = +1$ of points of class *A*+ points and from the plane $x^T w - 1 \cdot \gamma = -1$ of points of class *A*−. Consequently, the plane is:

$$x^T w - 1 \cdot \gamma = 0 \quad (4)$$

It is midway between and parallel to the proximal planes (3), is a separating plane that approximately separates *A*+ from *A*− as depicted in figure 1. The separation is only approximate, here and in general, because no plane can separate all points of *A*+ from those of *A*− when their convex hulls intersect. The second term in the quadratic objective function of (2), which is twice the reciprocal of the square of the 2-norm distance between the two proximal planes of (3) (see fig. 2), maximizes this distance, often called the "margin".

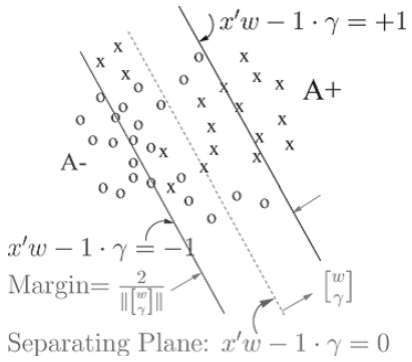

Fig. 2. Proximal support vector machine.

Maximizing *the* margin enhances the generalization capability of a support vector machine (Vapnik, 2000; Cherkassky & Mulier, 1998). The approximate separating plane (4) as depicted in figure 1, acts as a linear classifier as follows:

$$x^T w - \gamma \begin{cases} > 0, x \in A+ \\ < 0, x \in A-. \\ = 0, x \in A+ or A- \end{cases} \quad (5)$$

We note that the PSVM formulation (2) can be also interpreted as a *regularized* least squares solution of the system of linear *equations D*(*Aw* −*eγ*) = *e*, that is finding an approximate solution (*w, γ*) to *D*(*Aw* − *eγ*) = *e*, with least 2-norm.

Substituting for *y* in terms of *w* and *γ* from the linear constraint in the objective function of (2) gives the unconstrained minimization problem:,

$$\min_{(w,\gamma)} \frac{\nu}{2} \| D(Aw - e\gamma) - e \|^2 + \frac{1}{2} \left\| \begin{bmatrix} w \\ \gamma \end{bmatrix} \right\|^2 \quad (6)$$

Setting the gradient with respect to w and γ to zero and noting that $D2 = I$ give the necessary and sufficient optimality conditions for (6):

$$v A\_(Aw - e\gamma - De) + w = 0,$$
$$ve\_(-Aw + e\gamma + De) + \gamma = 0 \quad (7)$$

III. Experimental Evaluation

A. Dataset Used

The NSL-KDD dataset is used for experimentation purpose [14]. It is modified version of KDD99 dataset. KDD'99 dataset was based on the 1998 DARPA intrusion detection evaluation program, where an environment was setup to simulate a typical US Air Force LAN and raw tcpdump data was collected [15]. For each TCP/IP connection, 41 quantitative and qualitative features were extracted as a data record. The data records are all labelled as normal and abnormal. The NSL-KDD dataset has many advantages over KDD'99 like it doesn't include redundant records in training set as well as testing set. Thus our proposed model will not be biased towards more frequent records. NSL-KDD intrusion detection dataset is applied to the proposed model to demonstrate its effectiveness. The dataset contains a standard set of data to be audited, which includes a wide variety of intrusions simulated in a military network environment. It contains lakhs of records. So we have randomly selected 3740 records out of it.

The output class is classified as normal and abnormal. In experiment we have deleted those attributes which has similar values in every tuple.

B. Experimentation Criteria

NSL-KDD provides training as well as testing file. For the purpose of experimentation 2138 records having 29 attributes (excluding class attribute) is taken in training file and testing file contains 1970 records with same number of attributes. Testing file also has few intrusions that are not present in training data.

For calculating training accuracy, the training file containing 2138 records is divided into two parts viz. Training and testing. The first part i.e. training part is used for training the model i.e. for the construction of hyperplane. Then testing part is used to test the training model. Therefore the formulae used for calculating training accuracy is

Training accuracy= (correct classification of instance) / total number of instance in the second part of training dataset.

For calculating testing accuracy the NSL-KDD provides the test dataset which is used. This testing file is applied to the model generated and checked against the hyperplane created.

Testing accuracy= (correct classification of instance) / total number of instance in the test dataset.

C. Evaluation

The proposed PSVM model is tested. The network data records are normalized using z-score method. This normalized data is reduced using PCA. This pre-processed data is used to generate IDS model. For the purpose of training and testing we have used basic kernel functions and combination of these (Refer Part II section B). The performance of the classifiers includes training and testing accuracy. Experiments show that the combination of the kernel function gives better accuracy than basic kernel function. Accuracy of training and testing files depend on various parameters like gamma, degree etc. Accuracy of training and testing is given in TABLE I.(Training and testing accuracy of NSL-KDD dataset on PSVM) and the time to compute is given in TABLE II(Training and testing time for different kernel functions having same parameters[*].

The value of accuracy depends on various factors involved during computation of hyperplane and PSVM. These depending factors are gamma (kernel parameter in RBF and Sigmoid), degree of polynomial (used in polynomial kernel function), threshold (ν, refer section 2, PCA equation 7) and noise value. In experiment we have kept the values of these depending factors constant.

Parameter values: Gamma=.1, Degree=2, Threshold=.98888 and Noise value=0.1

TABLE I: Training and Testing Accuracy of NSL-KDD Dataset on PSVM

Kernel Function	Training Accuracy (%)	Testing Accuracy (%)
Polynomial	99.4387	77.2866
RBF	99.2516	72.0020
Sigmoid	99.1581	78.2012
Polynomial+(10*RBF)	99.5323	78.4045
Polynomial+Sigmoid	98.6904	77.7949
Sigmoid+(10*RBF)	99.2516	75.7114
Polynomial-10*rbf_norm	99.3452	79.3699
Polynomial-10*Sigmoid	99.5323	77.6931
Sigmoid-RBF	99.0645	74.8476

TABLE II: Training and Testing Time for Various Kernel Functions

Kernel Function	Training Time(s)	Testing Time(s)
Polynomial	39.72	78.89
RBF	21.96	78.75
Sigmoid	50.55	85.83
Polynomial+(10*RBF)	70.1	79.65
Polynomial+Sigmoid	69.93	80.72
Sigmoid+(10*RBF)	62.98	80.08
Polynomial-10*rbf_norm	67.76	80.89
Polynomial-10*Sigmoid	64.68	81.31
Sigmoid-RBF	61.59	81.86

IV. Conclusion

This paper proposes performance evaluation of PSVM using various combination of kernel function for Intrusion Detection System. The testing file from NSL_KDD contains few records that are not present in the raining file. Then also the proposed method gives satisfactory training accuracy and testing accuracy. Also the composition of kernel functions performs better than basic kernel functions.

REFERENCES

[1] R. Lippmann and R. Cunningham, "Improving intrusion detection performance using keyword selection and neural networks," *Computer Networks*, vol. 34, no. 4, pp. 597—603, 2000.

[2] W. Lee and S. J. Stolfo, "Data mining approaches for intrusion detection," in *Proc. of the 1998NIX Security Symposium*, 1998.

[3] J. Gomez and D. Dasgupta, "Evolving fuzzy classifiers for intrusion detection," in *Proc. of the 2002 IEEE Workshop on Information Assurance United States Military Academy*, West Point, NY, June 2001.

[4] J. Cannady, "Artificial neural networks for misuse detection," *National Information Systems Security Conference*, 1998.

[5] E. Eskin, "Anomaly detection over noisy data using learned probability distribution," in *Proc. of the International Conference on Machine Learning*, 2000.

[6] L. Portnoy, E. Eskin, and S. Stolfo, "Intrusion Detection with unlabeled data using clustering," *ACM CSS Workshop*, 2001.

[7] D. S. Kim and J. S. Park, "Network-Based intrusion detection with Support Vector Machines," *Information Networking*, 2003.

[8] X. Xu and X. Wang, "An adaptive network intrusion detection method based on PCA and Support Vector Machines," *Advanced data Mining and Applications, First International Conference, ADMA*, 2005.

[9] Principal component analysis. [Online]. Available: http://en.wikipedia.org/wiki/Principal_component_analysis.

[10] KDD Cup 1999 Data. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[11] K. P. Soman, S. Diwakar, and V. Ajay, *Data Mining Theory and Practice*, PHI Learning Pvt. Ltd., 2006 pp. 238-270

[12] I. Lindsay Smith, *A Tutorial on Principal Components Analysis*, Cornell University, USA, 2002.

[13] G. Fung and O. L. Mangasarian, "Proximal Support Vector Machine Classifiers," *International Conference on Knowledge Discovery and Data Mining*, 2001.

[14] The NSL-KDD Data Set. [Online]. Available: http://www.iscx.ca/NSL-KDD/.

[15] KDD 99 cup Competition. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/task.html.

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

[17] MATLAB. User's Guide. *The Math Works*, Inc., Natick, MA01760, 1994-2001. http://www.mathworks.com.