**Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Security and Defense Applications (CISDA 2007)**

# Intrusion Detection Model Based On Particle Swarm Optimization and Support Vector Machine

Surat Srinoy, Student *Member, IEEE*

***Abstract*—Advance in information and communication technologies, force us to keep most of the information electronically, consequently, the security of information has become a fundamental issue. The traditional intrusion detection systems look for unusual or suspicious activity, such as patterns of network traffic that are likely indicators of unauthorized activity. However, normal operation often produces traffic that matches likely "attack signature", resulting in false alarms. One main drawback is the inability of detecting new attacks which do not have known signatures. In this paper particle swarm optimization (PSO) is used to implement a feature selection, and support vector machine (SVMs) with the one-versus-rest method serve as a fitness function of PSO for classification problems from the literature. Experimental result shows that our method allows us to recognize not only known attacks but also to detect suspicious activity that may be the result of a new, unknown attack. Our method simplifies features effectively and obtains a higher classification accuracy compared to other methods.**

## I. Introduction

INTRUSION detection is a problem of great significance to protecting information systems security; especially in view of the internetworking is a crucial aspect of daily life around the world increasing incidents of cyber attacks on the critical infrastructures. It includes attempting to destabilize the network, gaining unauthorized access to files with privileges, or mishandling and misusing of software. The intrusion detection is to automatically scan network activity and detection attacks. As defined in [1], intrusion detection is "the process of monitoring the events occurring in a computer system or network and analyzing them for signs of intrusions. It is also defined as attempts to compromise the confidentiality, integrity, availability, or to bypass the security mechanisms of a computer or network".

There have been many techniques for modeling anomalous and normal behaviors for intrusion detection. The signature-based and supervised anomaly detections are widely deployed and commercially available. The signature-based detection extracts features from the network data. It detects intrusions by comparing the feature values to a set of attack signatures provided by human experts. However, it can only detect previously known intrusions with a signature. The signature database has to be manually revised for each new type of discovered attacks. On the other hand, the supervised anomaly detection trains models on labeled data (i.e., data pre-classified as an attack or not) and checks how well new data fit into the model. Obviously, it cannot be quickly adapted to new types of intrusion and do not have enough labeled data available. In general, a very large amount of network data needs to be handled and classified. Hence, it is impractical to classify them manually.

Intrusion detection techniques can be categorized in misuse detection and anomaly detection [2]. Misuse detection systems find intrusions by matching sample data to known intrusive pattern. Anomaly detection systems find intrusion by analyzing the deviation from normal activities profiles that are retrieved from historical data. Intrusion detection is a critical component of secure information systems. Many approaches have been proposed which include statistical [3], machine learning [4], data mining [5] and immunological inspired techniques [6]. Statistical analysis techniques are widely used in anomaly detection [7]. Compared with machine learning methods, statistical analysis techniques have an advantage that they can run in real time without offline learning and relearning from training data, but its detection performance is not good enough [8].

Many SVMs have been successfully applied to gene expression data classification problems [9], [10], [11]. Since they are not negatively affected by high dimensionality; hence they can obtain a higher accuracy than a general classification methods, optimize the obtained support vector machine. This avoids a common disadvantage of general classification methods, namely the long operation time, and can reduce training errors of the SVMs.[12]

In this paper, PSO is used to implement a feature selection, and SVMs with the one-versus-rest method were used as evaluators for the PSO fitness function for five multi-class problems taken from the literature. The results reveal that our method elucidated a better accuracy than the classification methods they were compared to.

The rest of this paper is organized as follows. In section II, we discuss the related works and describe a brief introduction to particle swarm optimization algorithm and support vector machine; in section III, experimental design section IV, experimental results and comparison. Finally, section V presents our conclusion, some discussion and future research.

Manuscript received November 1, 2006. This work was supported in part by the Department of Computer Science at Suan Dusit Rajabhat University under Grant.

Surat Srinoy is with Faculty of Science and Technology at Suan Dusit Rajabhat University, Bangkok, 10300 Thailand. (phone: 662-244-5225; fax: 662-668-7136; e-mail: surat_sri@ dusit.ac.th).

**1-4244-0700-1/07/$20.00 ©2007 IEEE**

## II. RALEATED WORKS

In a classification problem, the number of features can be quite large, many of which can be irrelevant or redundant. Since the amount of audit data that an IDS needs to examine is very large even for a small network, classification by hand is impossible. Feature reduction and feature selection improves classification by searching for the subset of features, which best classifies the training data. Some of the important features an intrusion detection system should possess include refer in Srilatha et al. [13].

Most intrusion occurs via network using the network protocols to attack their targets. Twycross [14] proposed a new paradigm in immunology, Danger Theory, to be applied in developing an intrusion detection system. Alves et al. [15] presents a classification-rule discovery algorithm integrating artificial immune systems (AIS) and fuzzy systems. For example, during a certain intrusion, a hacker follows fixed steps to achieve his intention, first sets up a connection between a source IP address to a target IP, and sends data to attack the target [16].

Generally, there are four categories of attacks [14]. They are: 1) DoS (denial-of-service), for example ping-of-death, teardrop, smurf, SYN flood, and the like. 2) R2L : unauthorized access from a remote machine, for example guessing password, 3) U2R : unauthorized access to local super user (root) privileges, for example, various "buffer overflow" attacks, 4) PROBING: surveillance and other probing, for example, port-scan, ping-sweep, etc. Some of the attacks (such as DoS, and PROBING) may use hundreds of network packets or connections, while on the other hand attacks like U2R and R2L typically use only one or a few connections.

### A. Particle Swarm Optimization

PSO is a new branch in evolutionary algorithms, which were inspired in group dynamics and its synergy and were originated from computer simulations of the coordinated motion in flocks of birds or schools of fish. As these animals wander through a three-dimensional space, searching for food or evading predators, these algorithms make use of particles moving in an *n*-dimension space to search for solutions for an n-variable function optimization problem. In PSO, individuals are called particles and the population is called a swarm. [17]

The initial swarm is generally created in such a way that the population of the particles is distributed randomly over the search space. At ever iteration, each particle is updated by following two "best" values, called *pbest* and *gbest.* Each particle keeps track of its coordinates in the problem space, which are associated with the best solution (fitness) the particle has achieved so far. This fitness value is stored, and called *pbest*. When a particle takes the whole population as its topological neighbor, the best value is a global "best" value and is called *gbest*. The pseudo code of the PSO procedure is given below.

Initialize population
While (number of generations, or the stopping criterion is not met)
  For p=1 to number of particles
  If the fitness of $x_p$ is greater than the fitness or $pbest_p$
  Then Update $pbest_p=X_k$
    For $k \in Neighborhood\ of\ X_p$
      If the fitness of $X_k$ is greater than that of *gbest* than
        Update $gbest=X_k$
      Next k
        For each dimension d

$$v_{pd}^{new} = w \times v_{pd}^{old} + c_1 \times rand_1 \times \left(pbest_{pd} - x_{pd}^{old}\right)$$
$$+ c_2 \times rand_2 \times \left(gbest_d - x_{pd}^{old}\right)$$
$$if\ v_{pd} \notin (v_{\min}, v_{\max})\ then$$
$$v_{pd} = \max\left(\min\left(v_{\max}, v_{pd}\right), v_{\min}\right)$$
$$x_{pd} = x_{pd} + v_{pd}$$

        Next d
    Next p
  Next generation until stopping criterion

Figure 1. The pseudo code of PSO algorithm [12]

$v_{pd}^{new}$ *and* $v_{pd}^{old}$ Are the particle velocities, $x_{pd}^{old}$ is the current particle position (solution), and $x_{pd}^{new}$ is updated particle position (solution). The values $pbest_{pd}$ and $gbest_d$ are defined as stated above. The two factors $rand_1$ and $rand_2$ are random numbers between (0, 1), where $c_1$ and $c_2$ are acceleration factors, usually $c_1=c_2=2$. Particle velocities of each dimension are tried to a maximum velocity $v_{max}$. If the sum of velocities causes the total velocity of that dimension to exceed $V_{max}$. $V_{max}$ is a user-specified parameter.

Based on the rules of particle swarm optimization, we set the required particle number first, and then the initial coding alphabetic string for each particle is randomly produced, in our case we coded each particle to imitate a chromosome in a genetic algorithm. Each particle was coded to a binary alphabetic string $S=F_1\ F_2\ K\ F_n$, $n$=1, 2, k, $m$; the bit value {1} represents a selected feature, whereas the bit value {0} represents a non-selected feature.

The adaptive functional values were data based on the particle features representing the feature dimension; this data was classified by a support vector machine (SVM) to obtain classification accuracy; the SVM serves as an evaluator of the PSO fitness function. For example, when a 10-dimensional data set (n=10) $S_n = (F_1F_2F_3F_4F_5F_6F_7F_8F_9F_{10})$ is analyzed using particle swarm optimization to select features, we can select any number of features smaller than n, i.e. we can chose a random 6 features, here $S_n = (F_1F_3F_5F_7F_9F_{10})$. When

the adaptive value is calculated, these 6 features in each data set represent the data dimension and are evaluated by the SVM. The fitness value for the SVM evolves according to the K-fold Cross-Validation Method [18] for small sample sizes, and according to the Holdout Method [18] for big sample sizes. Using the K-Fold Cross-Validation Method, we separated the data into 10 parts $\{D_1, D_2, \text{K}, D_{10}\}$, and carried out training and testing a total of 10 times. If ever part $D_n$, $n$=1, 2, K, 10 is processed as a test set, the other 9 parts will be training sets. Following 10 times of training and testing, 10 classification accuracies are produced, and the averages of these 10 accuracies are used as the classification accuracy for the data set. When the Holdout Method is used, the data can be divided into two parts, a training set part, which contains a larger amount of data, and a test set part, which contains relatively fewer data. We assumed that the obtained classification accuracy is an adaptive functional value.

Each particle renewal is based on its adaptive value. The best adaptive value for each particle renewal is *pbest*, and the best adaptive value within a group of *pbest* is *gbest*. Once *pbest* and *gbest* are obtained, we can keep track of the features of *pbest* and *gbest* particles with regard to their position and speed. In this study, a binary version of a PSO algorithm is used for particle swarm optimization [19]. The position of each particle is given in a binary string from that represents the feature selection situation. Each particle is updated according to the following equations.

$$v_{pd}^{new} = w \times v_{pd}^{old} + c_1 \times rand_1 \times \left(pbest_{pd} - x_{pd}^{old}\right) + c_2 \times rand_2 \times \left(gbest_d - x_{pd}^{old}\right) \quad (1)$$

$$S\left(v_{pd}^{new}\right) = \frac{1}{1+e^{-v_{pd}^{new}}} \quad (2)$$

$$if\ \left(rand < S\left(v_{pd}^{new}\right)\right) then\ x_{pd}^{new} = 1; else\ x_{pd}^{new} = 0 \quad (3)$$

The feature after renewal is calculated by the function $S\left(v_{id}^{new}\right)(Eq.2)$, in which the speed value is $v_{pd}^{new}$. *If* $S\left(v_{pd}^{new}\right)$ is larger than a randomly produced disorder number that is within (0, 1), then its position value $F_n$, $n$=1, 2, K, $m$ is represented as {1} (meaning this feature is selected as a required feature for the next renewal). If $S\left(v_{id}^{new}\right)$ is smaller than a randomly produced disorder number that is within {0~1}, then its position value $F_n$, $n$=1, 2, K, $m$ is represented as {0} (meaning this feature is not selected as a required feature for the next renewal).

### *B. Support Vector Machine*

Support vector machine (SVM) were originally introduced by Vapnik and co-workers [20] for classification tasks, and were subsequently extended to regression problems [21]. The idea behind SVMs is the following: input points are mapped to a high dimensional feature space, share a separating hyper-plane can be found. The algorithm is chosen in such a way as to maximize the distance from the closets patterns, a quantity which is called the margin. SVMs are learning systems designed to automatically trade-off accuracy and complexity by minimizing an upper bound on the generalization error provided by the Vapnik-Chervonenkis (VC) theory [22]. In a variety of classification problems, SVMs have showed a performance which can reduce training and testing errors, thereby obtaining higher recognition accuracy. SVMs can be applied to very high dimensional data without changing their formulation.

The hyper-plane of SVMs is usually found by using a quadratic programming routine, which is then solved with optimization routines form numerical libraries. These steps are non-trivial to implement and computationally intensive [20]. In this study, Kernel-Adatron (KA) algorithm [20], are used to emulate SVM training procedures, which combine the implementation simplicity of the Adatron with the capability of working in nonlinear feature spaces. The Adatron comes with the theoretical guarantee of converging exponentially fast in a given number of iterations, provided that a solution exists [23], [24]. By introducing Kernels into the algorithm it is possible to find a maximal margin hyper-plane in a high feature space, which is equivalent to nonlinear decision boundaries in the input space. The algorithm comes with all the theoretical guarantees given by the VC [22] theory for large margin classifiers [25], as well as the convergence properties detailed in the statistical mechanics literature.

The Kernel-Adatron algorithm theoretically converges in a finite number of steps to the maximal margin, provided that the linearly independent data points are linearly separable in the feature space with a margin $\lambda$>0. This result can be obtained for the following two reasons; all the fixed points of KA are Kuhn-Tucker points and, vice versa, KA always converges to a unique fixed point [26]. The KA procedure is described below.

1. initialize $\alpha_i = 1 \quad and\ \theta = 0.$
2. calculate
$$z_i = \sum_{j=1}^{n} \alpha_j y_j K\left(x_i, x_j\right) \quad (4)$$
3. calculate $\gamma_i = y_i\left(z_i - \theta\right)$
4. Let $\delta\alpha_i = \eta\left(1 - \gamma_i\right)$ be the proposed change to $\alpha_i$.
   if $\left(a_i + \delta\alpha_i\right) \le 0 \quad then\ \alpha_i \longrightarrow 0$
   if $\left(a_i + \delta\alpha_i\right) > 0 \quad then\ \alpha_i \longrightarrow \alpha_i + \delta\alpha_i$
5. Calculate the new threshold $\theta$
$$\theta = \frac{1}{2}\left(\min\left(z_i^+\right) + \max\left(z_i^-\right)\right) \quad (5)$$
where
$z_i^+$ *are those patterns i with class* $+1$ *and* $z_i^-$
this with class label -1.

6. if a maximum number of presentations of the pattern set has been exceeded or the margin $m = \frac{1}{2}\left(\min\left(z_i^+\right) - \max\left(z_i^-\right)\right)$ has approached 1 then stop. Otherwise return to step 2.

Figure 2. The Kernel-Adatron Algorithm[20]

The maximum number of iterations is 100, and the kernel function is the Radial Basis Function (RBF):

$$k\left(x_i, y_j\right) = \exp^{-r\left\|x_i - y_j\right\|}, i = 1, 2, k, n \qquad (6)$$

This algorithm is a gradient ascent routine that maximizes the margin in the feature space similar to perceptron-like algorithm, the Adatron, and was dubbed by Campbell and Christianini the Kernel-Adatron algorithm [20]. C and r are used to control the trade-off between training error and generalization ability. The decomposition techniques used for KA are one-versus-rest.

Initialize population
While (number of iterations, or the stopping criterion is not met)
For p=1 to number of particles
Segment training data and testing data
Initialize super parameter α

$$k\left(x_i, y_j\right) = \exp^{-r\left\|x_i - y_j\right\|x_p}$$

While (number of iterations,
or the stopping criterion is not met)
For i=1 to number of training data

$$z_i = \sum_{j=1}^{n} \alpha_j y_j k\left(x_i, y_j\right)$$

$$\delta\alpha_i = \eta\left(1 - z_i y_i\right)$$

If $\left(\alpha_i + \delta\alpha_i\right) \le 0 then\ \alpha_i = 0$

If $\left(\alpha_i + \delta\alpha_i\right) > 0 then\ \alpha_i = \left(\alpha_i + \delta\alpha_i\right)$

Next i
Next iteration until criterion
For i=1 to number of testing data

$$z_i = \sum_{j=1}^{n} \alpha_j y_j k\left(x_i, y_j\right)$$

If $z_i$>0 then $class_i$ =+1 else $class_i$ = -1
If $class_i$=*real class of testing data then*
*right=right+1*
next i
$fitness_p$=*right|number of testing data*
if the fitness of $x_p$ is greater than the fitness of $pbest_p$
then Update $pbest_p = x_p$
for k ∈ Neighborhood of $x_p$
if the fitness of $x_k$ is greater than that of *gbest* then
Update *gbest*=$x_k$
Next k
For each dimension d

$$v_{pd}^{new} = w \times v_{pd}^{old} + c_1 \times rand_1 \times \left(pbest_{pd} - x_{pd}^{old}\right)$$
$$+ c_2 \times rand_2 \times \left(gbest_d - x_{pd}^{old}\right)$$
$$S\left(v_{pd}^{new}\right) = \frac{1}{1 + e^{-v_{pd}^{new}}}$$
$$if\ \left(rand < S\left(v_{pd}^{new}\right)\right) then\ x_{pd}^{new} = 1; else\ x_{pd}^{new} = 0$$

Next d
Next p
Next generation until stopping criterion.

Figure 3 Pseudo code of the proposed method [12]

The two factors $rand_1$ and $rand_2$ are random numbers between (0, 1), where $c_1$ and $c_2$ are learning factors, usually $c_1$=$c_2$=2.

## III. EXPERIMENTAL DESIGN

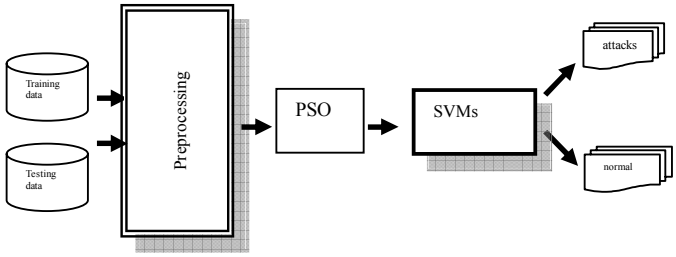


Fig 4. Overall Structure of Proposed Method

A detailed description of this method is shown in Fig 4. In first phase, we processed about preprocessing also handle missing and incomplete data. In second phase, feature selection using PSO and support vector machine clustering for detection group of data. In addition to this process, we manipulated the KDD'99 data set with importance attribute for processing. The preprocessing module performs the following tasks:

1. Identifies the attributes and their value.
2. Convert categorical to numerical data.
3. Data Normalization
4. Performs redundancy check and handle about null value.

## IV. EXPERIMENTAL RESULT

In this experiment, we use a standard dataset the raw data used by the KDD Cup 1999 intrusion detection contest [28]. This database includes a wide variety of intrusions simulated in a military network environment that is a common benchmark for evaluation of intrusion detection techniques. In general, the distribution of attacks is dominated by probes and denial-of-service attacks; the most interesting and

dangerous attacks, such as compromises, are grossly under-represented [29]. The data set has 41 attributes for each connection record plus one class label. There are 24 attack types, but we treat all of them as an attack group. A data set of size N is processed. The nominal attributes are converted into linear discrete values (integers). After eliminating labels, the data set is described as a matrix *X,* which has *N* rows and $m$=41 columns (attributes). There are $m_d$=8 discrete-value attributes and $m_c$ = 33 continuous-value attributes.

We ran our experiments on a system with a 1.6 GHz Pentium M processor and 512 MB DDR RAM running Windows XP. All the preprocessing was done using MATLAB®. MATLAB's Particle Swarm Optimization Toolbox [30] was used for Particle Swarm Optimization, whereas support vector machine operations were done in Support Vector Machine Toolbox [30, [31]. SVMs Toolbox is a software toolkit capable of performing all the operations for data processing and classification. In practice, the number of classes is not always known beforehand. There is no general theoretical solution to finding the optimal number of clusters for any given data set. We choose k = 5 for the study. We will compare five classifiers which have been also used in detecting these four types of attacks.

### *A. Data Preprocessing*

A considerable amount of data-preprocessing had to be undertaken before we could do any of our modeling experiments. It was necessary to ensure though, that the reduced dataset was as representative of the original set as possible. The test dataset that previously began with more than 300,000 records was reduced to approximately 18,216 records. Table 1 shows the dataset after balanced among category for attack distribution over modified the normal and other attack categories. Preprocessing consisted of two steps. The first step involved mapping symbolic-valued attributes to numeric-valued attributes and the second step implemented non-zero numerical features. We reduce the dimensionality of this data set (by using rough set) from 41 to 10 attributes are *duration, service, src_bytes, dst_byte, count, srv_count, serror_rate, dst_host_srv_count, dst_host_diff_srv_rate, and dst_host_same_src_port_rate.*

Table 1. Number and types of attacks in training data set

| Category | % Occurrence | Number of instances |
|---|---|---|
| Normal | 19.930 | 60,593 |
| Probe | 1.370 | 4,166 |
| DoS | 73.300 | 222,853 |
| U2R | 0.023 | 70 |
| R2L | 5.377 | 16,347 |
| **Total** | **100** | **304,029** |

### *B. Feature Selection*

Feature selection techniques aim at reducing the number of unnecessary features in classification rules. Particle Swarm Optimization theory has been used to define the necessity of features.

Feature selection is an optimization process in which one tries to find the best feature subset, from the fixed set of the original features, according to a given processing goal and a feature selection criterion. A pattern's features, from the point of view of processing goal and type, may be irrelevant (having no effect on processing performance) or relevant (having an impact on processing performance). Features can be redundant (correlated, dependent) [32]. When we process volumes of data, it is necessary to reduce the large number of features to a smaller set of features. There are 42 fields in each data record and it is hard to determine which fields are useful or which fields are trivial. Jin et al [33] suggest correlation coefficients between fields by using SPSS. They propose that if the correlation coefficients of fields *i* and *j*, *R(i,j)*, is larger than 0.8, then there is a strong correlation between fields *i* and *j*, and will select either one of them to represent these two fields. PSO allow us to determine (for a discrete attribute data set) a set called a core, containing strongly relevant features, and reducts, containing core plus additional weakly relevant features, such that each reduct is satisfactory to determine concepts in the data set. Based on a set of reducts for a data set some criteria for feature selection can be formed, for example a selecting feature from a reduct containing the minimal set of attributes [32].

### C. *Performance Measure*

Standard measures for evaluating IDSs include *detection rate*, *false alarm rate*, *trade-off between detection rate and false alarm rate* [34], *performance* (Processing speed + propagation + reaction), and *Fault Tolerance* (resistance to attacks, recovery, and subversion). Detection rate is computed as the ratio between the number of correctly detected attacks and the total number of attacks, while false alarm (false positive) rate is computed as the ratio between the numbers of normal connections that are incorrectly misclassified as attacks [35]. These are good indicators of performance, since they measure what percentage of intrusions the system is able to detect and how many incorrect classifications are made in the process.

Table 2. Performance of Fuzzy *c*-Mean

| **Class type** | **# of record** | **Hit** | **Miss** | **Detection rate** | **False Alarm rate** |
|---|---|---|---|---|---|
| Normal | 5,763 | 5,749 | 14 | 99.76% | 0.24% |
| Probe | 2,164 | 2,164 | 0 | 100% | 0% |
| DoS | 3,530 | 2,897 | 633 | 82.07% | 17.93% |
| U2R | 70 | 67 | 3 | 95.71% | 4.29% |
| R2L | 6,689 | 6,145 | 544 | 91.87% | 8.13% |
| summary | 18,216 | 17,022 | 1,194 | 93.45% | 6.55% |

Table 3. Performance of Particle Swarm Optimization

| **Class type** | **# of record** | **Hit** | **Miss** | **Detection rate** | **False Alarm** |
|---|---|---|---|---|---|

| | | | | | rate |
|---|---|---|---|---|---|
| Normal | 5,763 | 5,748 | 15 | 99.74% | 0.26% |
| Probe | 2,164 | 2,090 | 74 | 96.58% | 3.42% |
| DoS | 3,530 | 3,470 | 60 | 98.30% | 1.70% |
| U2R | 70 | 46 | 24 | 65.71% | 34.29% |
| R2L | 6,689 | 3,489 | 3,200 | 52.16% | 47.84% |
| summary | 18,216 | 14,843 | 3,373 | 81.48% | 18.52% |

Table 4. Performance of Support Vector Machine

| Class type | # of record | Hit | Miss | Detection rate | False Alarm rate |
|---|---|---|---|---|---|
| Normal | 5,763 | 5,744 | 19 | 99.67% | 0.33% |
| Probe | 2,164 | 2,128 | 36 | 98.34% | 1.66% |
| DoS | 3,530 | 3,493 | 37 | 98.95% | 1.05% |
| U2R | 70 | 28 | 42 | 40.00% | 60.00% |
| R2L | 6,689 | 2,290 | 4,399 | 34.24% | 65.76% |
| summary | 18,216 | 13,683 | 4,533 | 75.12% | 24.88% |

Table 5. Performance of PSO-SVM

| Class type | # of record | Hit | Miss | Detection rate | False Alarm rate |
|---|---|---|---|---|---|
| Normal | 5,763 | 5,754 | 9 | 99.84% | 0.16% |
| Probe | 2,164 | 2,163 | 1 | 99.95% | 0.05% |
| DoS | 3,530 | 3,506 | 24 | 99.32% | 0.68% |
| U2R | 70 | 48 | 22 | 68.57% | 31.43% |
| R2L | 6,689 | 6,037 | 652 | 90.25% | 9.75% |
| summary | 18,216 | 17,508 | 708 | 96.11% | 3.89% |

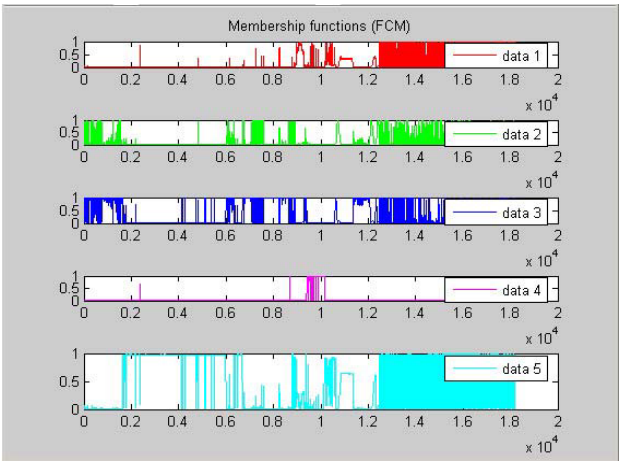


Fig 5: Membership functions of each cluster (Fuzzy set)

Anomaly detection amounts to training models for normal traffic behavior and then classifying as intrusions any network behavior that significantly deviates from the known normal patterns and to construct a set of clusters based on training data to classify test data instances. In table.2, table.3, table.4, table. 5 and fig. 5 are result from our experiments.

## V. CONCLUSION

In this paper we apply particle swarm optimization and support vector machine methods to intrusion detection to avoid a hard definition between normal class and certain intrusion class and could be considered to be in more than one category (or from another point of view it allows representation of overlapping categories). We introduce the current status of intrusion detection systems (IDS) and PSO-SVM based feature selection heuristics, and present some possible data mining based ways for solving problems. PSO-SVM based methods with data reduction for network security are discussed. Intrusion detection model is a composition model that needs various theories and techniques. One or two models can hardly offer satisfying results. We plan to apply other theories and techniques in intrusion detection in our future work.

## ACKNOWLEDGMENT

This work was implemented algorithm based on step that ref [12]. foremost, we would like to thank you Chung-Jui Tu, Li-Yeh Chuang, Jun-Yang Chang and Cheng-Hong Yang, from Department of Electronic Engineering, National Kaohsiung University of Applied Sciences, Kaohsiung, Taiwan for they algorithm in this paper.

## REFERENCES

[1] D.S Bauer, M.E Koblentz,. NIDX- an expert system for real-time network intrusion detection, *Proceedings of the Computer Networking Symposium*, 1988. pp. 98-106.

[2] Herv Debar, Marc Dacier, and Andreas Wespi, "Towards a Taxonomy of Intrusion Detection Systems", IBM Technical Paper, Computer Networks, Vol.39, Issue 9, pp. 805-822, April 1999.

[3] R. Bace and P. Mell, "Intrusion Detection Systems", NIST Special Publication on Intrusion Detection System, 31 November 2001.

[4] A.Sundaram, "An introduction to intrusion detection, Crossroads": The ACM student magazine, 2(4), April 1996.

[5] D. Denning, "An intrusion-detection model", In IEEE computer society symposium on research in security and privacy, 1986, pp. 118-131.

[6] T.Lane, "Machine Learning techniques for the computer Security", PhD thesis, Purdue University, 2000..

[7] Madori Asaka, Takefumi Onabuta, Tadashi Inoue, Shunji Okazawa, and Shigeki Goto, "A new intrusion detection method based on discriminant analysis", Oakland, pp.130-143, May 2001.

[8] Ming Tang, Song-Can Chen, Yi Zhuang and Jia Liu, "Using Statistical Analysis and Support Vector Machine Classification to Detection Complicated Attacks", Proc. Of the Third International Conference on Machine Learning and Cybernetics IEEE Computer Society, Shanghai, China, August 26-29, 2004.

[9] Furey, T.S., Cristianini, N., Duffy, N., Beddarski, K.W., Schummer, M. and Haussler, "Support vector machine classification and validation of cancer tissue samples using microarray expression data," *Bioinformatics,* vol. 16, pp.906-914, 2000.

[10] Guyon, I., Weston, J., Barnhill, S. and Vapnik, V., "Gene selection for cancer classification using support vector machines," *Machine Learning*, vol. 46, pp.389-422, 2002.

[11] Lee, Y. and Lee, C.-K., "Classification of multiple cancer types be multicategory support vector machines using gene expression data,"*Bioinformatics*, vol. 18, pp. 1132-1139, 2003.

[12] Chung-Jui Tu, Li-Yeh Chuang, Jun-Yang Chang, and Cheng-Hong Yang, "Fearture Selection using PSO-SVM", in *Proc. International multiconf. Engineers and computer scientist,* Hong Kong, 2006, pp.138-143.

[13] S. Chebrolu, A. Abraham, J. P. Thomas, *Feature Deduction and Ensemble Design of Intrusion Detection Systems*, Computer & Security, 2004.

[14] J. Twycross , Immune Systems, "Danger Theory and Intrusion Detection", *presented at the AISB 2004 Symposium on Immune System and Cognition*, Leeds, U.K., March 2004.

[15] R.T. Alves, M.R.B.S. Delgado, H.S. Lopes, A.A. Freitas, "An artificial immune system for fuzzy-rule induction in data mining", *Lecture Notes in Computer Science, Berlin: Springer-Verlag*,Vol.3242, 2004, pp.1011-1020.

[16] W. Lee and S. Stolfo, Data Mining Approaches for Intrusion Detection, *Proceedings of the 7th USENIX security symposium*, 1998.

[17] J. Kennedy, R.C. Eberhart, Particle Swarm Optimisation, in: Proceedings of the IEEE, International Conference on Neural Networks, Piscataway, 1995.

[18] Stone, M., "Cross-Validation choice and assessment of statistical predictions," *journal of the Royal Statistical Society B*, vol. 36, pp. 111-147, 1974..

[19] Kennedy, J., Eberhart, R.C., "A discrete binary version of the particle swarm algorithm", Systems, Man, and Cyernetics, 1997. 'Computational Cybernetics and Simulation]., 1997 IEEE international Conference on Volume 5, 12-15 Oct. 1997. pp.4104-4108.

[20] Frieβ, T., N. Cristianini, and C. Campbell, "The Kernel-Adatron: a Fast and Simple Learning Procedure for Suuport Vector machines," Proc. Of the Fifteenth Internaltional Conference on Machine Learning, pp. 188-196, 1998.

[21] Drucker, H., Burges, C., Kaufman, L., Smola, A. and Vapnik, "Support Vector regression Machines, " in Moser, M., Jordan, M. and Petshe, T. (ed.) Neural Information Processing Systems, Vol. 9. MIT Press, Cambridge, Ma, pp. 155-161. 1997.

[22] Vapnik, V.N., The Nature of Statisticcal Learning Theory, Springer Verlag, 1995.

[23] Anlauf, J.K., and Biehl, M., "The Adatron-an adaptive perceptron algorihm," Europhysics letters, 10, pp. 687-692. 1989.

[24] Opper, M., "Learning Time of Neural Networks: Exact Solution for a Perceptron Algorithm," *physical review* A38:3824. 1988.

[25] Boser, B., Guyon, I., Vapnik, "A training algorithm for optimal margin classifiers," Fifth Annual Workshop on Computational Learning Theory, ACM Press. 1988.

[26] Colin, C. and Nello, C., "Simple Learning. Algorithms for Training Support Vector Machines," 1998.

[27] Scholkopf, B. and Smola, *Learning with Kernels: support Vector Machines, Regularization, Optimization and Geyond.* MIT Press, Cambridge, MA.

[28] KDD data set, 1999; http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[29] P. Laskov, K. Rieck, C. Schäfer, K.R. Müller, "Visualization of anomaly detection using prediction sensitivity", *Proc.of Sicherheit,* April 2005, 197- 208.

[30] Math Works, *Statistical Toolbox for User's Guide, Math Works*, 2001.

[31] Steven R. Gunn, "Support Vector Machines for Classification and Regression," Technical report, 1998.

[32] W. Chimphlee, Abdul Hanan Abdullah, Mohd Noor Md Sap and S. Chimphlee, "Unsupervised Anomaly Detection with Unlabeled Data Using Clustering", *Proc. Int. Conf. on ICT-Mercu Buana ICT2005*. 42-49.

[33] H. Jin, J. Sun, H. Chen, and Z. Han, A Fuzzy Data Mining Based Intrusion Detection System, *Proc. of 10thInternational Workshop on future Trends in Distributed Computing Systems (FTDCS04) IEEE Computer Society*, Suzhou, China, May 26-28, 2004, 191-197.

[34] A. Lazarevic, A. Ozgur, L. Ertoz, J. Srivastava, and V. Kumar, A comparative study of anomaly detection schemes in network intrusion detection. *In SIAM International Conference on Data Mining*, 2003.

[35] T. Wakaki, H. Itakura, and M.Tamura, Rough Set-Aided Feature Selection for Automatic Web-Page Classification, *Proc. of the IEEE/WIC/ACM International Conference on Web Intelligence (WI'04)*