私立東海大學資訊工程與科學研究所

碩士論文

指導教授：林祝興 博士

Dr. Chu-Hsing Lin

基於 LibSVM 之異常偵測模擬分析

Simulation Analysis for Anomaly Detection Using LibSVM

研究生：何嘉瀚

(Chia-Han Ho)

NATIONAL CENTRAL LIBRARY
國家圖書館
ROC

中華民國九十七年六月

致 謝

本篇論文能順利完成，首先誠摯的感謝指導教授林祝興博士及劉榮春博士，兩位老師細心的教導以及不時的討論並指點我正確的方向，使我在這兩年中不論是做學問的方式或是待人接物皆獲益匪淺、受益良多。

本論文的完成另外亦得感謝蘇格蘭的紅色約翰先生大力協助。在我情緒最低落的時候，因為有你的體諒及扶持，使得本論文能夠更完整而嚴謹。

感謝 joker、coji、鎮宇、仁傑、菊人、佳穎學長、丸子學姐;同學猴子、swing、dipsy、懋樺;建廷、宗哲學弟、美君學妹們經常在課業及研究上互相提攜與協助。

最後，謹以此文獻給我摯愛的雙親。

Abstract

Intrusion detection is the means to identify the intrusive behaviors and provides useful information to intruded systems to respond fast and to avoid or reduce damages. In recent years, learning machine technology is often used as a detection method in anomaly detection. In this thesis, we use support vector machine as a learning method for anomaly detection, and use LibSVM as the support vector machine tool. By using this tool, we get rid of numerous and complex operation and do not have to use external tools for finding parameters as need by using other algorithms such as the genetic algorithm. Experimental results show that high average detection rates and low average false positive rates in anomaly detection are achieved by our proposed approach.

Keyword: Anomaly Detection, LibSVM, Intrusion Detection System, Support Vector Machine, One-class SVM

摘要

入侵偵測是一種對網路傳輸進行及時監視，在發現可疑行為時發出警報，或是採取主動反應措施以降低或避免系統傷害之網路安全技術。依照監測方式之不同，可以將入侵偵測技術分為異常偵測和誤用偵測二類。異常偵測則是對使用者或網路流量先建立一個「正常」的行為，再對通過的封包去做比對，假如超過正常行為的門檻值就是視為異常。

近年來，異常偵測時常運用到機器學習理論作為建立 「正常」行為的一個學習機制。本論文中，使用到了一個被廣泛運用的學習技術--支援向量機，作為本異常偵測模擬實驗之學習技術，而在支援向量機工具的使用則是選用臺灣大學林智仁教授所開發的 LibSVM。世界上有相當多關於這方面的研究都是使用到支援向量機，搭配額外的演算法如基因演算法或是類神經網路或是額外的核函數以達到高的偵測率。但在本論文的研究中，不需要搭配外來的幫助，即可以達到良好的偵測率，以及低的誤判率。也降低了額外所需的計算量。

關鍵詞：入侵偵測、異常偵測、支援向量機

Contents

List of Figures

List of Tables

Chapter 1
Introduction

Intrusion detection system (IDS) forms the second line of defense, and the intrusion detection technology has become critical to protect systems and users in the Internet age [1]. Intrusion detection is the means to identify and indicate the intrusive behaviors. Information of users is monitored and collected, and is analyzed to find the users' patterns of behavior. The gathered information is compared with known data to detect invasions, attacks and abnormal activities. Upon detection of intrusions, intruded systems respond to avoid or reduce further damages.

There are mainly two types of intrusion detection techniques: anomaly detection and misuse detection. We will focus on learning-based anomaly detection in this paper.

Anomaly detection uses statistical analysis methods to analyze normal users' behaviors on the Internet plus internal information flow statistics and records to build a profile. Then, this profile is used as a benchmark to classify activities of system operations. Abnormal activities are detected when events occur outside the scope of normal activities. The advantage of anomaly detection is that one needs not to worry about various possible attacks until the first occurrence of abnormal behaviors is recorded.

For anomaly detection, we train data by support vector machine (SVM) [2]. There are many researches with good results about learning-based IDS with SVM [3, 4] and anomaly detection [5, 6]. To improve the efficiency for anomaly detection, some researchers propose to combine SVM with other technologies, for example, neural networks [7,8,9], and genetic algorithm [10, 11, 12]. We study in this thesis the feasibility of using LibSVM for anomaly detection.

SVM is a statistical learning theory based on machine learning methods. A special property of SVM is that it simultaneously minimizes empirical classification errors and maximizes geometric margins. By training with lots of data, SVM learns to find the best compromise and give the best projection with limited information.

We use KDDCUP 1999 dataset as training and testing data [13]. Two forms of SVM: C-SVM and one-class SVM are used as classification technologies and LibSVM [14] is chosen as the SVM tool.

We find that we can get good results by using this tool without evolved procedures such as the selection of parameters, which is hard to decide when using SVM. There are many ways to try out best parameters, such as genetic algorithm (GA) which needs lots of computations and consumes much time.

The suitable SVM is found by observation of the experimental outcomes of anomaly detection by different types of SVM. The high average detection rates and low average false positive rates in anomaly detection show our proposed approach is feasible.

The rest of this thesis is organized as follows. In chapter 2, we will introduce briefly the two forms of SVM. In chapter 3, we will present our experiments and in chapter 4, the results. Conclusions will be given in chapter 5.

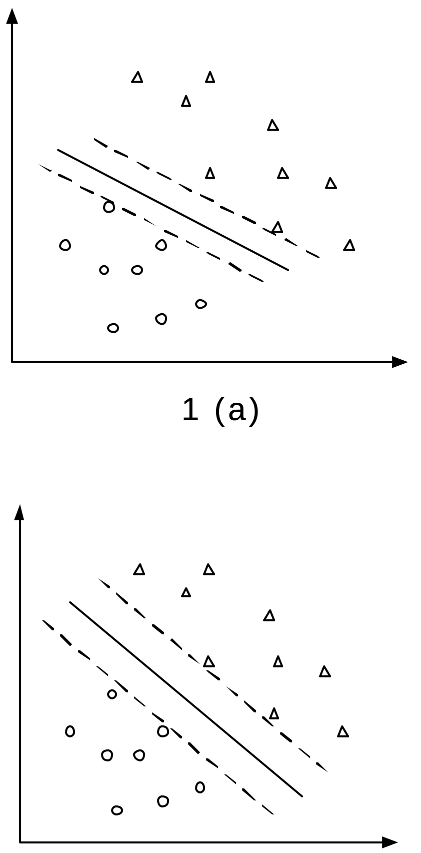
Chapter 2
Background

2.1 Support Vector Machine

Sometimes we want to catachrestically classify data into two groups. There exist a few good technologies for classification such as the naïve Bayes and neural networks. When applied correctly, these technologies give acceptable results. Most important advantages of SVM are simple to use and high precision.

SVM is a statistical learning theory based on machine learning methods. SVM is widely used in the respect of bioinformatics, data mining, image recognition, text categorization, hand-written digit recognition. The earlier SVM was designed to solve binary classification problems. It is important for SVM to solve multi-class classification in efficient ways. Some scholars propose related researches about multi-class SVM [15, 16].

The basic concept of SVM is to classify separable data in space R^d. We want to find a hyper-plane that separates these data into two groups, group A and group B in the R^d space. As shown in Figure 1, the data of group A are in the right and upper side of the hyper-plane, and the data of group B are in the other side of the hyper-plane. The margin between the two parallel hyper-planes in Figure 1 (a) is narrower than the gap between the two parallel hyper-planes in Figure 1 (b). Since hyper-planes with wider margin are preferred and so the hyper-plane in Figure 1 (b) is better.

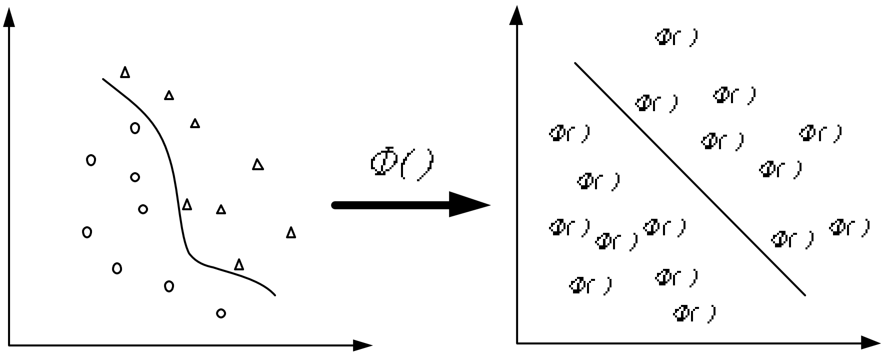


1 (a)

1 (b)

Figure1. Concept of SVM

In some non-linear cases, by transforming source data with a kernel function into high dimension space, one can solve non-linear data in original dimensions by separating into two parts with linear method in high dimensions to reduce the error [17]. The concept is shown in Figure 2.



2.1.1 C-SVM

The C-SVM is proposed by Cortes and Vapnik in 1995 [18] and Vapnik in 1998 [15]. The primal form is:

$$\min_{w,b,x,r} \quad \frac{1}{2} w^t w + \mathrm{C} \sum_{i=1}^{l} x_i$$

$$s.t. \;\; y_i \left(w^T \cdot f(x_i) + b \right) \geq 1 - x_i$$
$$, x_i \geq 0, i = 1 \ldots l$$

Where vectors $X_i \in R^n$, $i = 1, \ldots, l$ in two classes, and the vector $y_i \in R^l$ such that $y_i \in \{+1, -1\}$;

The dual is:

$$\min \quad \frac{1}{2} a^T Q a - e^T a$$
$$Q_{ij} = K(x_i, x_j) \equiv f(x_i)^T f(x_j)$$
$$s.t. \quad y^T a = 0$$
$$0 \leq a_i \leq C, \quad i = 1, \ldots, l$$

The decision function is:

$$\mathrm{sgn}(\sum_{i=1}^{l} y_i a_i K(x_i, x) + b)$$

e is the vector of all ones, C > 0 is the upper bound, Q is an l by l positive semidefinite matrix, $\mathrm{Q}_{\mathrm{ij}} \equiv \mathrm{y}_{\mathrm{i}} \mathrm{y}_{\mathrm{j}} \mathrm{K}(x_i, x_j)$,and $K(x_i, x_j) \equiv f(x_i)^T f(x_j)$ is the kernel. Here training vectors $\mathrm{x_i}$ are mapped into a higher (maybe infinite) dimensional space by the function $\boldsymbol{\Phi}$

The geometry interpretation of C-SVM is shown in Figure 3.

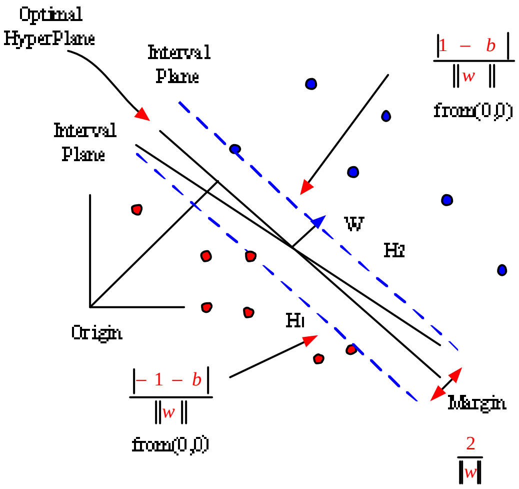


Figure 3. Geometry interpretation of C-SVM

As shown in Figure 3 the solid lines are the found hyper-planes. We call H_1 and H_2 the supporting hyper-planes. We want to find the best classification hyper-planes that have widest margin between the two supporting hyper-planes.

Definition of classification hyper-plane is:

$$w^T x = -b(w^T x + b = 0)$$

Therefore we can present supporting hyper-planes H_1 and H_2 as:

$$H_1 : w^T x + b + d$$

$$H_2 : w^T x + b - d$$

We scale H_1 and H_2 with constants *w, b,* and *d*:

$$H_1 : w^T x + b = 1$$

$$H_2 : w^T x + b = -1$$

The distance from H_1 to the origin is $\frac{|1-b|}{\|w\|}$. The distance from H2 to the origin is $\frac{|-1-b|}{\|w\|}$. The distance between H_1 and H_2 is $\frac{|2|}{\|w\|}$.

By above equation the data points should satisfy the following equations in R^d:

$$w^T x_i + b \geq 1 \quad for\, y_i = 1$$

$$w^T x_i + b \leq -1 \quad for\, y_i = -1$$

We can combine the two above inequality as:

$$y_i(\ w^T x_i + b\ ) \geq 1$$

And we get the widest margin between two Support Hyper-planes by:

$$Max\left(\frac{2}{\| w \|}\right), \text{or } \min\left(\frac{\| w \|}{2}\right)$$

2.1.2 One-class SVM

One-class SVM was proposed by Schölkopf et al. in 2001 for estimating the support of a high-dimensional distribution [19]. The base idea of one-class SVM is to separate data from the origin. Schölkopf et al. proposed a method to adapt the SVM one-class classification problem. After transforming the feature by the kernel function, the origin is seemed as the only member of the second class. Then the image of the one class is separated from the origin.

The algorithm can be summarized as mapping the data into a feature space H using a fit kernel function, and then trying to separate the mapped vectors from the origin with maximum margin:

Given training vectors $X_i \in R^n, \quad i = 1,...,l$, without any class information, the primal form is:

$$\min_{w,b,x,r} \quad \frac{1}{2} w^t w - r + \frac{1}{vl} \sum_{i=1}^{l} x_i$$

$$s.t.\ (w \cdot f(x_i)) \geq r - x_i, x_i \geq 0$$

The decision function is:

$$\operatorname{sgn}\left(\sum_{i=1}^{l} a_i K(x_i, x) - r \right)$$

The dual is:

$$\min \quad \frac{1}{2} a^T Q a$$

$$Q_{ij} = K(x_i, x_j) \equiv f(x_i)^T f(x_j)$$

$$s.t. \quad 0 \le a_i \le 1, \quad i = 1, \ldots, l$$

$$e^t a = vl$$

The geometry interpretation of one-class SVM is shown in Figure 4.

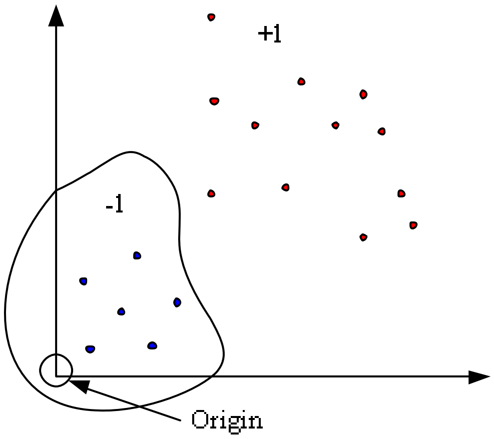


Figure 4. Geometry interpretation of one-class SVM

2.1.3 ?-Support Vector Classification

The ν-support vector classification (Schölkopf et al., 2000) uses a new parameter ν which controls the number of support vectors and training errors. The parameter $\nu \in$ (0; 1] is an upper bound on the fraction of training errors and a lower bound of the fraction of support vectors.

Given training vectors $X_i \in R^n, i = 1, \ldots, l$, in two classes, and a vector $y_i \in R^l$ such that $y_i \in \{1, -1\}$, the primal form considered is:

$$\min_{w,b,x,r} \quad \frac{1}{2} w^t w - vr + \frac{1}{l} \sum_{i=1}^{l} x_i$$

$$s.t. \ y_i \left(w^T \cdot f(x_i) + b \right) \geq r - x_i$$
$$, x_i \geq 0, i = 1 \ldots l, r \geq 0$$

The dual is:

$$\min \quad \frac{1}{2} a^T Q a$$
$$Q_{ij} \equiv y_i y_j K(x_i, x_j)$$

$$s.t. \quad e^T a \geq n, \quad y^T a = 0$$
$$0 \leq a_i \leq 1/l, \quad i = 1, \ldots, l$$

The decision function is:

$$\operatorname{sgn}(\sum_{i=1}^{l} y_i a_i K(x_i, x) + b)$$

In [23] [24], it shows that $e^T a \geq n$ can be replaced by $e^T a = n$,so in LibSVM

The dual is:

$$\min \ \frac{1}{2} a^T Q a$$

$$s.t. \quad e^T a = nl, \quad y^T a = 0$$
$$0 \leq a_i \leq 1 \quad i = 1, \ldots, l$$

The decision function is:

$$\operatorname{sgn}(\sum_{i=1}^{l} y_i (a_i / r)(K(x_i, x) + b))$$

The margins are $y_i \left(w^T \Phi(x_i) + b \right) = \pm 1$

2.1.4 Comparison of SVMs

There are several images show how SVM classify with 2 classes of data. The tool is offered in the website of LibSVM. Because of spreading of the data, different type of SVMs or different kernels there are different hyperplanes. Here we have 2 groups of pictures show how different SVMs work under the same data in Figure 5 and Figure 6.

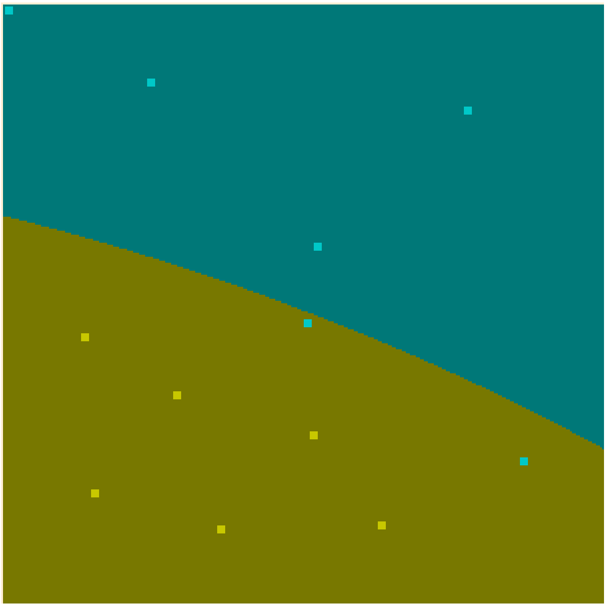


Figure 5.(a) C-SVM

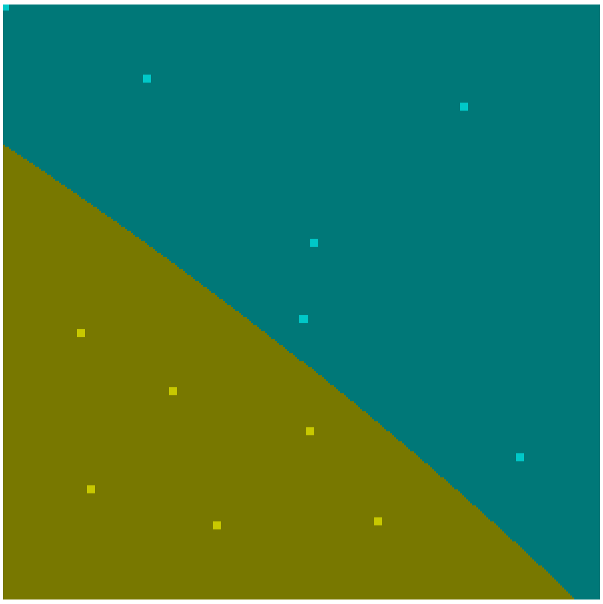


Figure 5.(b) ?-SVM

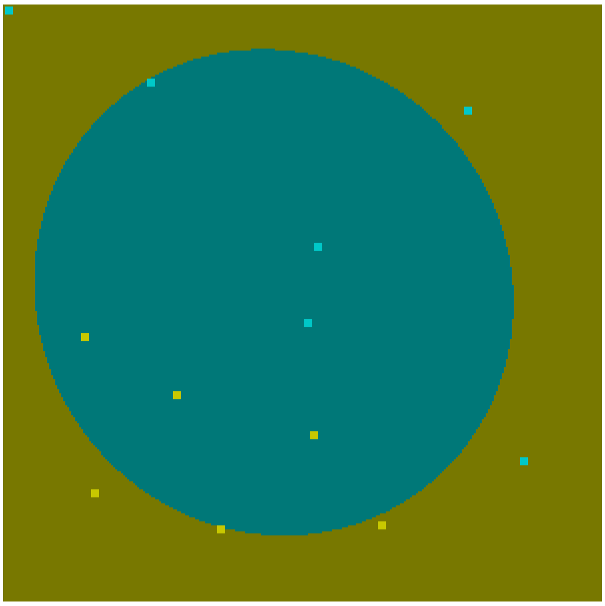


Figure 5.(c) One-class SVM

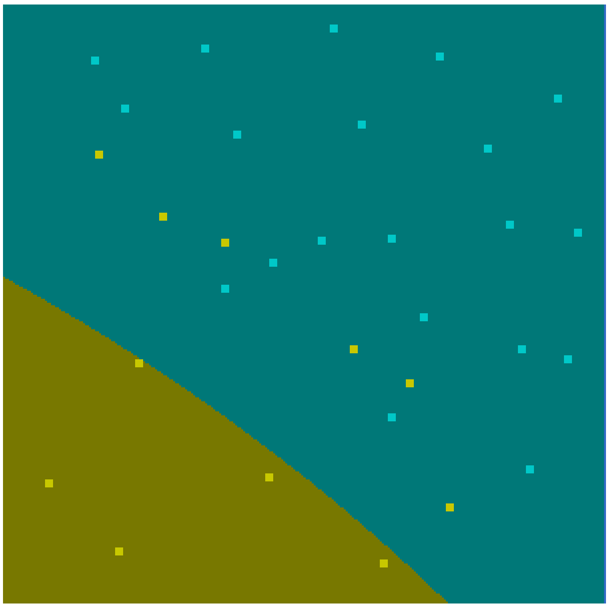


Figure 6.(a)C-SVM

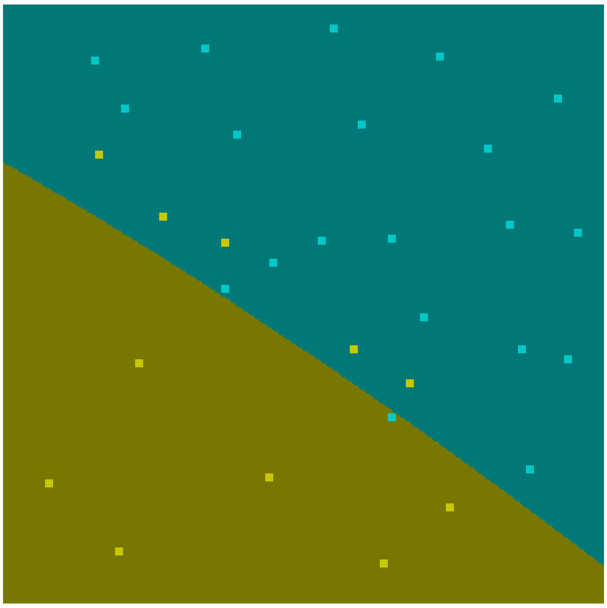


Figure 6.(b) ?-SVM

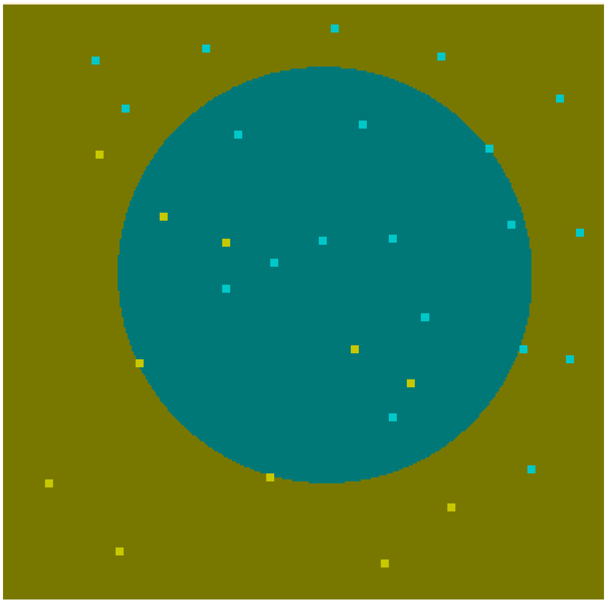


Figure 6.(c) one-class SVM

2.1.5 Kernels

Training vectors x_i are mapped into a higher (maybe infinite) dimensional space by the function *F*. Then SVM finds a linear separating hyperplane with the maximal margin in this higher dimensional space. C > 0 is the penalty parameter of the error term. $K(x_i, x_j) \equiv f(x_i)^T f(x_j)$ is called the kernel function. There are four kernels could be found in SVM books.

linear: $K(x_i, x_j) = x_i^T x_j$

polynomial: K(x_i; x_j) = $K(x_i, x_j) = (g\ x_i^T x_j + r)^d, g > 0$

radial basis function (RBF): $K(x_i, x_j) = \exp(-g \left\| x_i^T x_j \right\|^2), g > 0$

sigmoid: K(x_i; x_j) = $K(x_i, x_j) = \tanh(g\ x_i^T x_j + r)$

r, γ and *d* are kernel parameters.

2.1.5.1 RBF Kernel

The RBF kernel is suggested to use in this tool. This kernel nonlinearly maps vectors into higher dimensional space. It can handle the case when the label or attributes is not linear. It shows in [25] that the linear kernel is a special case of RBF kernel. The linear kernel with a parameter C has the same performance as the RBF kernel with some parameter *(C,* γ*)*. Also, the sigmoid kernel behaves like RBF for certain parameter [26].

The number of hyperparameters influences the complexity of model selection a lot. The polynomial kernel has more hyperparameters than the RBF kernels.

The RBF kernel has less numerical difficulties. One key point is $0 < K_{ij} \leqq 1$ in contrast to polynomial kernels of which kernel values may go to infinity $(g x_i^T x_j + g > 1)$ or zero $(g x_i^T x_j + g < 1)$ While the degree is large. The sigmoid kernel is not valid under some parameters [15].

2.1.6 LibSVM

LibSVM is a library for support vector machines. Its goal is to promote SVM as a convenient tool. It integrates C-SVM classification, ν-SVM classification, one-class

SVM, epsilon-SVM regression, and ν-SVM regression. It also provides an automatic model selection tool for C-SVM classification.

2.2 Intrusion Detection

Intrusion is defined by Heady et al. [27] “as any set of actions that attempt to compromise the integrity, confidentiality, or availability of a resource. “

Anderson define the concept of intrusion in [28]

- access information,
- manipulate information, or
- render a system unreliable or unusable.

An intrusion is a violation of the security policy of the system. The definitions above are general enough to encompass all the threats mentioned in the previous section. Any definition of intrusion is, of necessity, imprecise, as security policy requirements do not always translate into a well-defined set of actions. Whereas policy defines the goals that must be satisfied in a system, detecting breaches of policy requires knowledge of steps or actions that may result in its violation.

We can divide the techniques of intrusion detection into two main types

Anomaly Detection:

Anomaly detection techniques make an assumption that all intrusive behaviors are anomalous. It means if we could build a "normal behavior profile" for a system, we could flag all different system states from the established profile by statistically significant amounts as intrusion attempts. However, there are two dangerous situations:

(1) Anomalous activities that are not intrusive are flagged as intrusive.

(2) Intrusive activities that are not anomalous result in false negatives (events are

not flagged intrusive, though they actually are).

These are dangerous problems, and more serious than the problem of false positives.

Misuse Detection:

The concept behind misuse detection schemes is that there are ways to represent attacks in the form of a pattern or a signature so that even variations of the same attack can be detected. This means that these systems are not unlike virus detection systems -- they can detect many or all known attack patterns, but they are of little use for as yet unknown attack methods. An interesting point to note is that anomaly detection systems try to detect the complement of "bad" behavior. Misuse detection systems try to recognize known "bad" behavior. The main issues in misuse detection systems are how to write a signature that encompasses all possible variations of the pertinent attack, and how to write signatures that do not also match non-intrusive activity.

Chapter 3
Experiment

There have been several major approaches to anomaly intrusion detection. In our scheme, we use statistical approach to build our system. At the beginning, behavior profiles need to be generated. We use SVMs here for the learn algorithm to build the profile.

3.1 Processing Procedure

The following steps show how we input the data and train the SVM to get the model.

- Transform source data to the format of LibSVM
- Conduct simple scaling on the data
- Choose the RBF kernel
- Find the best parameter C and γ
- Train source data to get the model
- Test data with the model

3.2 Data Source

In our experiment, we used 1999 KDD Cup data set. These data are prepared and managed by MIT Lincoln Labs. Lincoln Labs set up an environment to acquire nine weeks of raw TCP dump data for a local-area network (LAN) simulating a typical U.S. Air Force LAN. They operated the LAN as if it were a true Air Force environment, but peppered it with multiple attacks. KDD Cup [20] is the leading Data Mining and Knowledge Discovery competition in the world, organized by ACM SIGKDD - Special Interest Group on Knowledge Discovery and Data Mining, the leading professional organization of data miners. In recent years, this data set has been widely used as a benchmark for evaluation of the intrusion detection technology.

We separated the source data into normal part and abnormal parts. There were twenty four known types of attacks in the source data. A connection was established when a sequence of TCP packets starting and ending at some well defined time span, in which data flowed between a source IP address and a target IP address under some well defined protocol. Each connection was labeled as either normal for normal users, or abnormal for attacks with exact one specific attacking type. Each connection record consisted of about 100 bytes.

3.3 Experimental Environment

The experiment was performed in the following experimental environment:

CPU: Pentium Core 2 Duel E6750

RAM: DDR II 667 2GB

OS: Microsoft Windows XP SP2

SVM tool: LibSVM 2.86 (released on April 1, 2008).

3.4 Data Processing

SVM requires that each data instance is represented as a vector of real numbers. Non-numerical data items are needed to change into numerical data formats to make the data trainable by LibSVM.

For example the source data is as follow:

0,tcp,http,SF,181,5450,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,0.00,0.00,1.00,0.00,0.00,9,9,1.00,0.00,0.11,0.00,0.00,0.00,0.00,0.00,normal.

We want to transform it as follow:

1 1:0 2:1 3:1 4:2 5:181 6:5450 7:0 8:0 9:0 10:0 11:0 12:1 13:0 14:0 15:0 16:0 17:0 18:0 19:0 20:0 21:0 22:0 23:8 24:8 25:0.00 26:0.00 27:0.00 28:0.00 29:1.00 30:0.00 31:0.00 32:9 33:9 34:1.00 35:0.00 36:0.11 37:0.00 38:0.00 39:0.00 40:0.00 41:0.00

If the feature is zero, it would be ignored in LibSVM.

First we make tables of every feature which is not numerical data. Then we transfer to integer by orders in the table, and the last feature of source data would be the label flagged as "normal" or "abnormal" and swap in the front of each data and shown as "1" and "0". The full features are listed in 3.4.

The characteristic of this data set was that 80% of it belonged to abnormal behaviors. The data is not like the usual cases in real world, but because it contains lots of abnormal data, we can let our system learn well.

3.5 Data Feature

The KDD cup 1999 dataset contains total 41 features in Table 1, include basic features of individual TCP connections, content features within a connection suggested by domain knowledge, and traffic features computed using a two-second time window. The exactly feature descriptions are in [13]. We train SVM by using the large number of data contains this feature.

Table1. Features of KDD cup 1999 dataset

duration	is_guest_login
Protocol_type	Count
Service	serror_rate
src_byte	rerror_rate
dst_byte	same_srv_rate
flag	diff_srv_rate
land	srv_count
wrong_fragment	srv_serror_rate
urgent	srv_rerror_rate
hot	srv_diff_host_rate

num_failed_logins	dst_host_count
logged_in	dst_host_srv_count
num_compromise	dst_host_same_srv_rate
root_shell	dst_host_diff_srv_rate
su_attempted	dst_host_same_src_port_rate
num_root	dst_host_srv_diff_host_rate
num_file_creations	dst_host_serror_rate
num_shells	dst_host_srv_serror_rate
num_access_files	dst_host_rerror_rate
num_outbound_cmds	dst_host_srv_rerror_rate
is_hot_login	

Chapter 4
Results

We performed our experiment in two parts:

A. *Use C-SVM & ?-SVM for classifying technology.*

In this experiment we want to simulate the anomaly detection with 2 SVMs. We use two rates to evaluate the system, detection rate and false positive rate.

The detection rate is $\dfrac{True\ \ Positive + True\ \ Negative}{All\ \ Behaviors}$.

The false positive rate is $\dfrac{False\ \ \ Positive}{Number\ \ \ of\ \ \ "normal\ "\text{instances}}$.

	abnormal	normal
Judged “abnormal”	True positive	False positive
Judged “normal”	False Negative	True Negative

We processed the source data and made them fit with the format of SVM. First we took 20,000 as a unit, and tested 5 times, labeled as A1~A5, respectively. Second we took 50,000 as a unit, and tested 5 times, labeled as B1~B5, respectively. Third, we took 100,000 as a unit, and tested 5 times, labeled as C1~C5, respectively.

We drew randomly 60% of each test for training data. The left 40 % of data were validation data. The results are listed in Table 2. It shows good test results with very high average detection rates above 97%.

Table2. Detection rates by using C-SVM for classifying technology

	Avg. Detection Rate		Avg. False Positive rate	
	C-SVM	?-SVM	C-SVM	?-SVM
A	98.7%	90.3%	2.0%	6.5%
B	97.6%	87.5%	1.5%	7.3%
C	97.3%	92%	1.7%	5%

B. *Use one-class SVM for classifying technology.*

The main idea of this experiment is that we want to test (1) the efficiency of one-class SVM (2) to compare the efficiency of different algorithm. We duplicated the experiment environments in [21]. The p-kernel is a new kind of kernel described in [22] together with other techniques of detection. The normal data were divided into three parts, the train data, the test data and the validation data. Then, we drew 10,000 samples from source data, among them 6000 samples for training data, 2000 samples for test data, and 2000 samples for validation data. The results are listed in Table 3. A high average detection rate of 95% is shown in it when using LibSVM.

Table3. Detection rates by using one-class SVM for classifying technology.

Algorithm	Avg. Detection rate	Avg. False Positive rate
K-NN	91%	10%
Naïve Bayes	89%	8%
SVM(light)	91%	10%
P-kernel SVM	98%	6%
Our Scheme	**95%**	**7%**

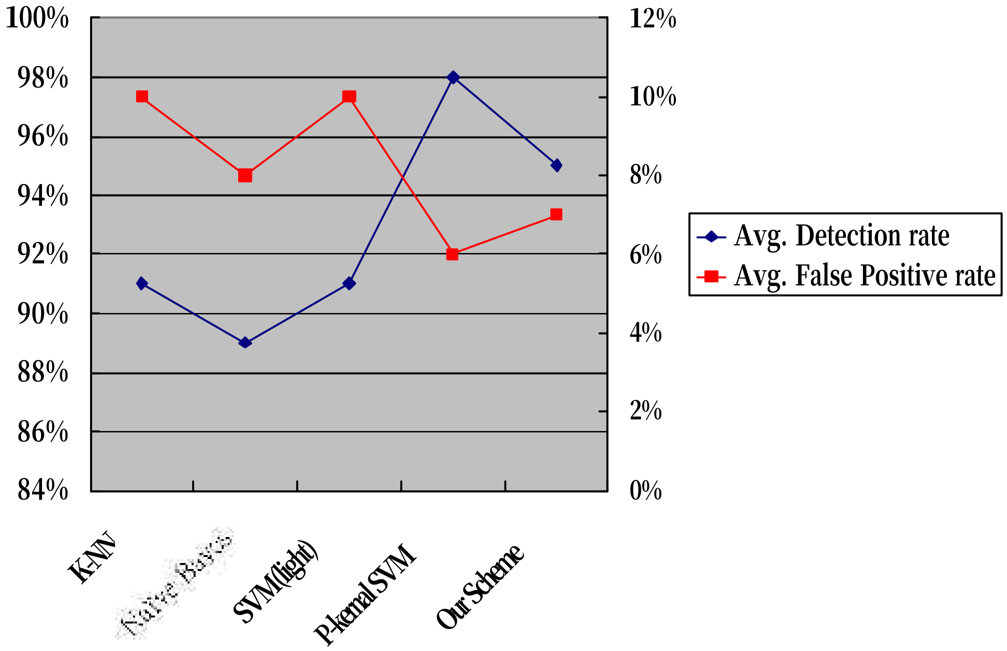


Figure 7. Result of Experiment Part B

Chapter 5
Conclusion

We use SVM to simulate this learning-based anomaly detection system. And in the choice of tools, we use LibSVM as a SVM tool. We compare the effectiveness of this SVM tool with other algorithms of unsupervised SVM based on p-kernels for anomaly detection. This research is referred to the use of p-kernel with SVM-light and gets nearly perfect results. We can easily get good result with average detection rate up to 95% using LibSVM only and its default parameters and kernel (RBF), without needless of other external kernels.

In the respect of C-SVM, by using default parameters, the result of detection rate and false positive are very good, but in test of ?-SVM the result is not always good as the test in C-SVM. By observing the experiment we can understand that the C-SVM may not be the latest technology of classification but if we can find the fit parameters, we can easily get nice results, in our simulations we all use default ones. This proves that the method we choose is a simple and effective way to achieve high detection rates.

We do obtain nice results by using LibSVM with the KDD Cup 1999 dataset and three forms of SVM. But since both the attacking technology and the detection technology are updating very fast, our future work is to design new methods and train them with latest data or some extreme cases, and to classify data efficiently with new forms of SVM.

Bibliography

[1] G.Giacinto, F.Roli, L.Didaci, “Fusion of multiple classifiers for intrusion detection in computer networks,” Pattern Recognition Letters. Vol.24, pp. 1795–1803, 2003

[2] Cristianini N., and Taylor J.S., “An introduction to support vector machine,” Cambridge University Press, Cambridge, UK, 2000.

[3] B.J. Kim, “Kernel Based Intrusion Detection System,” International Conference on Information Systems archive Proceedings of the Fourth Annual ACIS International Conference on Computer and Information Science (ICIS'05) ,pp.13-18, 2005

[4] H.Li, XH.Guan, X.Zan, et al, “ Network intrusion detection based on support vector machine, ” Journal of Computer Research and Development, vol.40, no.6, pp.799-807, 2003

[5] M. Fugate and J.R. Gattiker, “Anomaly Detection Enhanced Classification in Computer Intrusion Detection,” In Pattern Recognition with Support Vector Machines, First International Workshop, Niagara Falls, Canada, August 10, 2002, Lecture Notes in Computer Science 2388, pp. 186-197.

[6] Wenjie Hu and Yihua Liao and V. Rao Vemuri, “Robust Support Vector Machines for Anomaly Detection in Computer Security,” In Proceedings of 2003 International Conference on Machine Learning and Applications, Los Angeles, CA, June 23-24, 2003.

[7] S. Mukkamala, G. Janoski, A H. Sung, "Intrusion Detection Using Neural Networks and Support Vector Machines," In Proceedings of IEEE International Joint Conference on Neural Networks, IEEE Computer Society Press, 2002, pp.1702-1707.

[8] S Mukkamala, A H. Sung, "Feature Selection for Intrusion Detection Using Neural Networks and Support Vector Machines," Proceedings of the 82nd Annual Meeting of the Transportation Research Board, National Academics.

[9] Li Yang, Guo Li, "An Efficient Network Anomaly Detection Scheme Based on TCM-KNN Algorithm and Data Reduction Mechanism" Information Assurance and Security Workshop, 2007. IAW '07. IEEE SMC20-22 Page(s):221 – 227 , June 2007

[10] Anh Tran Quang, Qianli Zhang, Xing Li, "Attack recall control in anomaly detection," In Proceedings of ICCT 2003, International Conference on Communication Technology, vol. 1, Page(s): 382 - 384, 9-11 April 2003

[11] Syng-Yup Ohn, Ha-Nam Nguyen, Dong Seong Kim, Jong Sou Park, "Determining Optimal Decision Model for Support Vector Machine by Genetic Algorithm," In International Symposium on Computational and Information Sciences, Shanghai, China, December 16-18, 2004, Lecture Notes in Computer Science, pp. 895-902.

[12] Dong Seong Kim, Ha-Nam Nguyen, Jong Sou Park, "Genetic algorithm to improve SVM based network intrusion detection system," Advanced Information Networking and Applications, 2005. AINA 2005, 19th

International Conference on Volume 2, 28-30 March 2005 Page(s):155 - 158 vol.2

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

[14] Chih-Chung Chang and Chih-Jen Lin, LIBSVM: a library for support vector machines, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

[15] Vapnik V., "Statistical learning theory," John Wiley and Sons, New York, 1998.

[16] J. Weston and C. Watkins, "Multi-class support vector machines," In Proceedings of ESANN99, Brussels, 1999.

[17] Boser, B. E., Guyon, I. M., Vapnik, V., "A Training Algorithm for Optimal Margin Classifiers," Fifth Annual Workshop on Computational Learning Theory, ACM, (1992).

[18] Cortes. C, and Vapnik. V., "Support vector networks," Machine Learning, vol.20, no2, pp.273-297, 1995

[19] Bernhard Schölkopf et al. "Estimating the support of a High-Dimensional Distribution," Technical Report, Department of Computer Science, University of Haifa, Haifa, 2001.

[20] http://www.sigkdd.org/kddcup/index.php

[21] K. Li and G. Teng, “Unsupervised SVM Based on p-kernels for Anomaly Detection,” ICICIC’06, 2006 International Conference on Innovative Computing, Information and Control.

[22] J.P. VERT, “Support Vector Machine Prediction of signal peptide cleavage site using a new class of kernels for strings,” Pacific Symposium on Biocomputing, vol 7:649-660, 2002.

[23]D. J. Crisp and C. J. C. Burges. A geometric interpretation of ?-SVM classifiers. In S. Solla, T. Leen, and K.-R. Muller, editors, Advances in Neural Information Processing Systems, volume 12, Cambridge, MA, 2000. MIT Press.

[24] C.-C. Chang and C.-J. Lin. Training ?-support vector classifiers: Theory and algorithms. *Neural Computation*, 13(9):2119-2147, 2001.

[25]Keerthi, S. S. and C.-J. Lin. Asymptotic behaviors of support vector machines with Gaussian kernel. Neural Computation 15 (7), 1667-1689, 2003.

[26]Lin, H.-T. and C.-J. Lin (2003). A study on sigmoid kernels for SVM and the training of non-PSD kernels by SMO-type methods. Technical report, Department of Computer Science, National Taiwan University.

[27]R. Heady, G. Luger, A. Maccabe, M. Servilla, “The architecture of a network level intrusion detection system,” Technical report, Computer Science Department, University of New Mexico, August 1990.

[28]J.P. Anderson. Computer security threat monitoring and surveillance. Technical Report Contact 79F26400, James P. Anderson Co., April 1980.

[29]C.H. Lin , J. C. Liu, C.H. Ho, “Anomaly Detection Using LibSVM Training Tools,” ISA’08; 2008 The 2nd International Conference on Information Security and Assurance