Winter 12-5-2004

Application of Weighted Support Vector Machines to Network Intrusion Detection

Yinshan Jia

Chuanying Jia

Hongwei Qi

Application of Weighted Support Vector Machines to Network Intrusion Detection

Yinshan Jia [1,2], Chuanying Jia [2], Hongwei Qi [3]

[1]School of Information Technology, Liaoning University of Petroleum and Chemical Technology, Fushun 113001, China
[2]Dalian Maritime University, Dalian 116026, China
[3]Fushun Ethylene Chemical Co. Ltd., Fushun 113004, China
{jiayinshan, chuanyingjia, qi_hongwei}@sina.com

ABSTRACT

Support Vector Machines(SVMs) have succeeded in many classification fields. Some researchers have tried to apply SVMs to Intrusion Detection recently and got desirable results. By analyzing C-SVM theoretically and experimentally, we found that C-SVM had some properties which showed C-SVM was not most suitable for Network Intrusion Detection. First, C-SVM has different classification error rates on different classes if the sizes of training classes are uneven. Second, C-SVM is over-dependent on every training sample, even if the samples are duplicated. Third, C-SVM does not make a difference between training samples. According to these characteristics of C-SVM and the fact that the size of network normal data is always much larger than that of intrusion data and the fact that the importance of attack data is different from each other, an extended C-SVM, termed weighted C-SVM is proposed in this paper. Weighed C-SVM introduces two parameters, class weights and sample weights. Class weights are used to adjust false negative rate and false positive rate of each intrusion class. And sample weights are used to emphasize importance of some intrusion samples. Experiments showed that Weighted C-SVM was more effective than C-SVM in network intrusion detection systems.

Keywords: Network Intrusion Detection, Support Vector Machines, Weighted Support Vector Machines, Machine Learning

1. INTRODUCTION

With the rapidly increasing connectivity and accessibility to the Internet, network security has been pained more and more attention to. Intrusion detection systems are considered an effective measure against network attacks. Intrusion is generally defined as violating confidentiality, integrity and availability of computer or computer network system[1]. The kernel of detections is to extract the behavior model of networks. Currently, many methods have been applied to create detection model, including Neural Networks, Data Mining, and so on[2]. All these methods learn on training data sets first to create detection model, and then use the detection model created by learning process to monitor future behaviors of network. Intrusion detection is a classification problem in nature. It classifies the network behavior to normal class and attack class, or normal class and attack classes. The previous is a binary classification problem and the other is a multi-class classification problem.

Support vector machines are canonized by many researchers and have been applied to many classification fields successfully. The advantages of SVMs over conventional classification methods are its high generalization ability especially if the number of training data is small, its adaptability to various classification problems by changing kernel functions, and its global optimal solution. Recently, some researchers began to apply SVM technology to intrusion detection. Mukkamala, et al.[3] described approaches to intrusion detection using support vector machines and demonstrated that efficient and accurate classifiers can be built to detect intrusions using SVMs. They also compared the performance of neural networks based and SVM based systems for intrusion detection, and concluded that at the same level of accuracy, the training time for SVMs was significantly shorter, and running time of SVMs was also notably shorter. They also used SVMs in feature ranking and selection for intrusion detection systems[4] and identifying key variables for intrusion detection [5]. In these papers, they did only binary classification experiments, and got very desirable results. John S. Baras and Maben Rabi[6] used SVMs to detect intrusion on a host computer system. In order to handle variable-length strings, the approach combined the ability of an HMM generative model. Ming Luo, et al.[7] and Fugate, et al. [8]used unsupervised clustering and support vector machines to anomaly detection. Other works on applications of support vector machines on intrusion detection can be found in[9],[10],[11],[12].

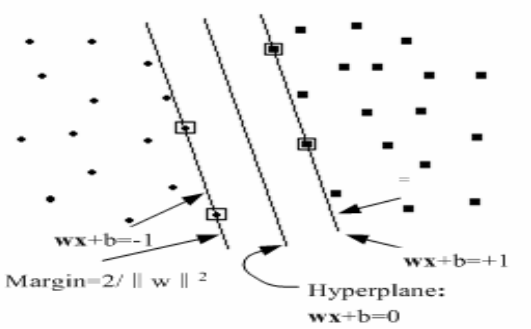

Figure 1. Principle of support vector machines

2. SUPPORT VECTOR MACHINES

Support Vector Machines, proposed by by Cortes and Vapnik[13][14], is a relative new machine learning methodology based on statistical learning theory. This support vector machine is considered as the standard support vector machine and C-SVM[15][16]. Its basic idea is to find the hyperplane which can separate data belonging to two classes with maximum margin. This hyperplane is called optimal hyperplane. Figure 1 shows the principle of SVMs.

In Figure 1, the points marked with rectangles are Support Vectors (SVs), which determine the hyperplane and the margin. In the case of nonlinear separation, SVMs use a map $f : X \to H$ to transform each data point **x** to $f(\mathbf{x})$ in higher dimensional feature H so that the data points can be separate linearly more probably. X is called input space, and H is called feature space. If $f(\mathbf{x})$ can not separate linearly in H, then SVMs find a hyperplane in H that minimize the error cost. This kind of hyperplane is called optimal hyperplane.

2.1 C-Support Vector Machine

Given $Z = \{(x_i, y_i) \mid x_i \in R^n, y_i \in \{+1, -1\}, i = 1,2,...,m\}$ is a set of samples, where x_i is a data vector, y_i is the label of the class that x_i belongs to. In order to seek the hyperplane that best separates the two classes from each other with the widest margin, we need to solve the following optimization problem[14]:

$$\min \tau(w, x) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}\xi_i, \quad (1)$$

$$\text{s.t} \quad y_i(< w, x_i > + b) \geq 1 - \xi_i, \quad (2)$$

$$\text{and} \quad \xi_i \geq 0, i = 1,...,m. \quad (3)$$

Where w is a normal vector of the hyperplane $< w, x_i > + b = 0$, b is the bias of the hyperplane, C is a penalty factor that determines the tradeoff between the maximization of the margin and the minimization of error cost, $< w, x_i >$ denotes dot product of w and x_i, each ξ_i is a slack variable that denotes the distance from x_i to margin plane $< w, x > + b = y_i$. In order to separate data more precisely, we use a map $f : X \to H$, where H is a higher dimensional space. Taking into account the compute complexity, we select a kernel function $k(x_i, x_j) = < \phi(x_i), \phi(x_j) >$. Introducing Lagrange multiplier a_i and b_i, we have:

$$L(w, x_i, b, a_i, b_i) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m} x_i - \sum_{i=1}^{m} [a_i(y_i(< w, f(x_i) > + b) - 1 + x_i) + b_i x_i]. \quad (4)$$

This function has to be minimized with respect to the primal variables w, x_i, b, and maximized with respect to the dual variables a_i, b_i. To eliminate the former, we compute the corresponding partial derivatives and set them to 0, obtaining:

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{m} a_i y_i f(x_i) = 0, \quad (5)$$

$$\frac{\partial L}{\partial b} = -\sum_{i=1}^{m} a_i y_i = 0, \quad (6)$$

$$\frac{\partial L}{\partial x_i} = C - a_i - b_i = 0. \quad (7)$$

Substituting (5), (6) and (7) into L, using $a_i, b_i \geq 0$, and incorporating kernels for dot products, we obtain the dual Lagrangian:

$$\max W(a) = \sum_{i=1}^{m} a_i - \frac{1}{2}\sum_{i,j=1}^{m} a_i a_j y_i y_j k(x_i, x_j), \quad (8)$$

$$\text{s.t.} \quad 0 \leq a_i \leq C, i = 1,...,m, \quad (9)$$

$$\text{and} \quad \sum_{i=1}^{m} a_i y_i = 0. \quad (10)$$

where a_i ,i=1,…,m, are Lagrange multipliers. If $a_i > 0$, then x_i lies on or in margin, and x_i is called a standard support vector. If $0 < a_i < C$, then x_i lies on margin, and x_i is called a in-bound support vector. If $a_i = C$, then x_i lies in margin or misclassified, and x_i is called a bounded support vector[15]. By solving the above dual Lagrangian, we obtain:

$$w = \sum_{i=1}^{m} a_i y_i f(x_i). \quad (11)$$

To Compute b ,we take into account that due to KTT conditions, for in-bound support vector x_j for which ξ_i=0, constrain (2) becomes

$$\sum_{i=1}^{m} y_i a_i k(x_j, x_i) + b = y_j. \quad (12)$$

Thus, b can for instance be obtained by averaging

$$b = y_j - \sum_{i=1}^{m} y_i a_i k(x_j, x_i). \quad (13)$$

over all data with $a_j > 0$ [15]. The resulting decision function can be shown as

$$f(x) = \mathrm{sgn}(\sum_{i=1}^{m} y_i a_i k(x_i, x) + b). \quad (14)$$

From (14), we know that using SVMs to classify new data need not to compute w after training SVMs. In most cases, we don't know the map f .After training SVMs, we store kernel function k, a_i which is nonzero and corresponding y_i, x_i , or store kernel function k , $a_i y_i$ and corresponding x_i . When a new data point needs to be classified, we can use these stored values and decision function (14) to classify x_i .

2.2 Analysis of C-SVM

In C-SVM, the KKT condition is:

$$a_i(y_i(< w, f(x_i) > + b) - 1 + x_i) = 0. \quad (15)$$

$$b_i x_i = (C - a_i) x_i = 0. \quad (16)$$

Therefore, there are tree cases as follows:

1)If $a_i = 0$, according to (16), then $x_i = 0$. x_i is correctly classified.

2)If $0 < a_i < C$, according to (15) and (16), then $y_i(< w, f(x_i) > + b) - 1 + x_i = 0$ and $x_i = 0$. x_i is a in-bound support vector.

3)If $a_i = C$, according to (15) and (16), then $y_i(< w, f(x_i) > + b) - 1 + x_i = 0$ and $x_i \geq 0$. x_i is a bounded support vector. If $0 \leq x_i < 1$, x_i is correctly classified. If $x_i \geq 1$, x_i is misclassified.

Suppose N_{BSV+} ,N_{BSV-} are the number of bounded support vectors in positive class and negative class respectively, and N_{SV+},N_{SV-} are the number of support vectors in positive class and negative class respectively, and m_+, m_- are the number of data points in positive class and negative class respectively. According to (10), we have:

$$\sum_{i=1}^{m} a_i = \sum_{i=1:y_i=+1}^{m} a_i + \sum_{i=1:y_i=-1}^{m} a_i. \quad (17)$$

$$\sum_{y_i=+1} a_i = \sum_{y_i=-1} a_i. \quad (18)$$

$$N_{BSV+} C \leq \sum_{y_i=+1} a_i. \quad (19)$$

$$N_{SV+} C \geq \sum_{y_i=+1} a_i. \quad (20)$$

Incorporating (19) and (20), we have:

$$N_{BSV+} C \leq \sum_{y_i=+1} a_i \leq N_{SV+} C. \quad (21)$$

Similarly, we can obtain:

$$N_{BSV-} C \leq \sum_{y_i=-1} a_i \leq N_{SV-} C. \quad (22)$$

Dividing (21), (22) by $C \times m_+$ and $C \times m_-$ respectively, and supposing $\sum_{y_i=+1} a_i = \sum_{y_i=-1} a_i = A$, we obtain:

$$\frac{N_{BSV+}}{m_+} \leq \frac{A}{C \times m_+} \leq \frac{N_{SV+}}{m_+}. \quad (23)$$

$$\frac{N_{BSV-}}{m_-} \leq \frac{A}{C \times m_-} \leq \frac{N_{SV-}}{m_-}. \quad (24)$$

From (23) and (24), we know that:

1)If $m_+ = m_-$, the fraction of bounded support vectors and fraction of support vectors in positive class is equal to that in negative class.

2)If $m_+ > m_-$, the fraction of bounded support vectors and fraction of support vectors in positive class is less than that in negative class.

3)If $m_+ < m_-$, the fraction of bounded support vectors and fraction of support vectors in positive class is greater than that in negative class.

4)Omitting any data point, even if it is a duplicated data point, will influence the result of classification.

Besides the four above characteristics, we can know intuitively C-SVM has other characteristics as follows:

5)If the training set has duplicated samples, there would be more duplicated support vectors. In this case, classification will be slower.

6)C-SVM does not take different importance of training samples into account. This would result in misclassification of important samples.

According to the characteristics analyzed above, we can conclude that C-SVM is not most suitable to intrusion detection. In intrusion detection, the training sets generally contain more normal samples than attack samples and contain many duplicated samples. New SVMs overcoming the shortcomings of C-SVM are needed.

3. WEIGHTED C-SUPPORT VECTOR MACHINE

In this section, we propose a novel C-SVM, termed weighted C-SVM, WC-SVM for short. The primal question in WC-SVM is

$$\min \tau(w, \xi, \rho) = \frac{1}{2}\|w\|^2 + \mu_{y_i} C \sum_{i=1}^{m} s_i \xi_i, \quad (28)$$

$$\text{s.t.} \quad y_i(< w, x_i > + b) \geq 1 - \xi_i, \quad (29)$$

$$\text{and} \quad \xi_i \geq 0, i = 1, ..., m. \quad (30)$$

Where $\mu_{y_i} \geq 1$ is the weight factor of class y_i, $s_i > 0$ is the important factor of data point x_i. The bigger s_i is, the more important x_i is. The primal question in WC-SVM is same as that of C-SVM if each s_i and μ_{y_i} is specified to 1. Comparing (1) and (28), we can understand the meaning of $\sum s_i \xi_i$ in (28). $\sum s_i \xi_i$ in (28) is practical cost of errors, and that $\sum \xi_i$ in (1) is geometric cost of errors.

With method same as that in C-SVM , we can obtain dual Lagrangian:

$$\max W(a) = \sum_{i=1}^{m} a_i - \frac{1}{2} \sum_{i,j=1}^{m} a_i a_j y_i y_j k(x_i, x_j), \quad (31)$$

$$\text{s.t.} \quad 0 \leq a_i \leq C m_{y_i} s_i, i = 1, ..., m, \quad (32)$$

and $\sum_{i=1}^{m} a_i y_i = 0$. (33)

The decision function is

$$f(x) = \mathrm{sgn}(\sum_{i=1}^{m} y_i a_i k(x_i, x) + b). \quad (34)$$

With analysis method same as in C-SVM, we have:

$$\frac{N_{BSV+}}{m_+} \# \frac{A}{C \cdot m_{y_i} \cdot s_i \cdot m_+} \quad \frac{N_{SV+}}{m_+}. \quad (35)$$

$$\frac{N_{BSV-}}{m_-} \# \frac{A}{C \cdot m_{y_i} \cdot s_i \cdot m_-} \quad \frac{N_{SV-}}{m_-}. \quad (36)$$

We can control the upper bound on the fraction of margin errors and fraction of SVs of each class by tuning the parameter m_{y_i} and s_i in (35) and (36). We think this is useful and important in network intrusion detections. First, attack data is always much less than normal data in sample set. If C-SVM is used, error rate on attack data will be bigger than that on normal data. This is not expected if the attack data contains some data whose corresponding behavior is very harmful to network systems. Second, detection is needed to be real time to avoid attack. We know from (14) and (34) that the number of SVs influences the detection speed. Although C-SVM provides a mean to tune the tradeoff between fraction of margin errors and fraction of SVs, the mean tunes the tradeoff of attack data and normal data simultaneously. In practice, we prefer tuning the tradeoff of each class independently to tuning the tradeoff of each class simultaneously. WC-SVM has advantages over C-SVM in these aspects apparently.

Moreover, attack behaviors do different harms to network systems. Some behaviors harm the systems seriously, and others do slightly. Some new behaviors are very harmful, and we expect IDS can detect them as long as the training sample set contains the data corresponding to them. If the data corresponding to the new harmful behaviors is very sparse in training set, C-SVM would misclassify them. This will result in false negatives in IDS. We introduce an important factor s_i for every train sample in WC-SVM in order to handle this case. If some data in training set is important, we can assign a big value to their corresponding s_is, otherwise, we can assign a small value to their corresponding s_is.

4. EXPERIMENTS

In this section, we use some experiment results to compare C-SVM and WC-SVM. Experiments on C-SVM were done with libsvm[17], and experiments on WC-SVM were done with a new program based on libsvm. Training samples and testing samples were selected randomly from KDD'99 data[18]. The kernel function used is RBF. Results of experiments are shown in Table 1, Table 2 and Table 3.

In Table 1, MR,NC,AC,NS denote Margin Errors, Normal Class, Attack Class and the Number of SVs respectively. In Table 2, FNR,FPR denote False Negative Rates and False Positive Rates respectively. In Table 3, NCAS, NMNST, IFNS denote the Number of Changed Attack Samples, the Number of Misclassified New Samples In Training, Important Factor of New Samples respectively.

Table 1. Results of training experiments

Experiment No.		1	2	3
Number of Attack Data		1000	100	500
Number of Normal Data		1000	2000	3000
C-SVM	C	1000	1000	1000
	Time(Seconds)	3.30	1.48	183.57
	ME of NC	0	14	0
	ME of AC	103	18	103
	NS in NC	196	115	232
	NS in AC	118	47	119
WC-SVM	C	1000	1000	1000
	μ_{+1}	1	1	1
	μ_{-1}	10	20	50
	s_i	1	1	1
	Time(Seconds)	65.51	3.645	233.57
	ME of NC	120	50	338
	ME of AC	0	0	0
	NS in NC	215	168	463
	NS in AC	43	21	36

Table 2. Results of test experiments

Experiment No. of Model		1	2	3
Number of Test Samples		2000	2000	2000
C-SVM	Time(Seconds)	2.53	1.16	2.27
	FNR(%)	100	75.60	100
	FPR(%)	0.19	8.34	0.20
WC-SVM	Time(Seconds)	1.64	1.39	3.45
	FNR(%)	93.40	25.32	37.1
	FPR(%)	13.04	56.60	36.54

Table 3. Results of test experiments on samples with important factors

Experiment No.		1	2	3
NCAS		10	15	20
Number of Test Samples		2000	2000	2000
C-SVM	NMNST	2	2	1
WC-SVM	IFNS	10	20	100
	NMNST	1	0	0

From results of experiments, we know that WC-SVM provide us means to adjust the correct rate and error rate of classification of each class and to emphasize the importance of each training sample.

5. CONCLUSIONS

We have presented a new SVM termed WC-SVM. From the point of application, WC-SVM has advantages over C-SVM. We can not only specify a weight factor for each training set but also specify an importance factor for each data point. If a data point is misclassified, the penalty error in WC-SVM equals the product of the penalty error in C-SVM and the importance factor of this point and the weight factor of the class which this point belongs to. This results in lower misclassification rates of training classes which have bigger weight factors and results in less probability of misclassification of training samples which have bigger important factors. In our opinion, this idea is useful to

intrusion detection. This is because we always need different classification performance, and we need to run intrusion detection systems before we collect much attack data.

REFERENCES

[1]Bace, R. G., *Intrusion Detection*, Macmillan Technical Publishing, 2000

[2]David,J.M.,*Computer Intrusion Detection and Network Monitoring:A Statistical Viewpoint*, Springer-Verlag, 2001.

[3]Srinivas Mukkamala, Guandalupe Janoski, Adrew Sung, "Intrusion Detection Using Neural Networks and Support Vector Machines", *Proceedings of IEEE International Joint Conference on Neural Networks 2002*, pp1702-1707, Hawaii, May 12-17, 2002

[4]Srinivas Mukkamala and Andrew, H.Sung, "Feature Ranking and Selection for Intrusion Detection Systems Using Support Vector Machines", *Proceedings of the 2002 IEEE International Conference on Information and Knowledge Engineering*, pp503-509, Las Vegas, Nevada, June 24-27,2002

[5]Srinivas Mukkamala, Andrew,H.Sung and Ajith Abraham, "Identifying Key Variables for Intrusion Detection Using Soft Computing Paradigms", *Proceedings of the 2003 IEEE International Conference on Fuzzy Systems*, St. Louis, Missouri, USA, May 25-28,2003

[6]John, S. Baras and Maben Rabi. "Intrusion Detection with Support Vector Machines and Generative Models", Research Report, University of Maryland, 2002

[7]Luo Ming,Wang Lina,Zhang Huanguo and Chen Jin, "A Research on Intrusion Detection Based on Unsupervised Clustering and Support Vector Machines", *ICICS2003*,*LNCS 2836*, pp325-336, Springer-Verlag , 2003

[8]Mike Fugate and James, R.G., "Normally Detection Enhanced Classification in computer Intrusion Detection", *SVM 2002*, *LNCS 2388*, pp.186-197, Springer-Verlag , 2002

[9] Kim Dong Seong and Park Jong Sou, "Network-Based Intrusion Detection with Support Vector Machines", *ICOIN 2003*, *LNCS 2662*, pp 747-756, Springer-Verlag , 2003

[10]Rao Xian, Dong Chunxi and YANG Shaoquan, "An Intrusion Detection System Based on Support Vector Machine"[In Chinese], *Journal of Software*, Vol. 14, No. 4, pp798-803, 2003

[11]Li Hui, Guan Xiaohong, Zan Xin and Han Chongzhao. "Network Intrusion Detection Based on Support Vector Machine" [In Chinese]. *Journal of Computer Research and Development*, Vol. 40,No. 6,pp799-807, 2003

[12]Tran Quang-anh, Zhang Qianli and Li Xing. "SVM Classification-based Intrusion Detection System"[In Chinese], *Journal of China Institute of Communications*, Vol. 23, No. 5,pp51-55,2002

[13]Cortes, C. and Vapnik, V., "Support vector networks", *Machine learning*, Vol. 20,No. 3, pp273-297,1995

[14]Vapnik, V., *The nature of statistical learning theory*, Springer-Verlag, 1995

[15]Schölkopf, B. and Smola, A. J., *Learning with kernels*, MIT Press, 2002

[16]Schölkopf, B., Smola, A. J., Williamson, R. C., et al., "New support vector algorithms", *Neural Computation*, Vol. 12,No.5, pp1207-1245, 2000

[17]Chang Chih-Chung and Lin Chih-Jen, "LIBSVM: a library for support vector machine, 2001", *http://www.csie.ntu.tw /~cjlin/libsvm*

[18]*http://kdd.ics.uci.edu/databases/kddcup99/task.htm*