International Journal of Computer Applications (0975 – 8887)
Volume 75 – No.13, August 2013

A Parallel Support Vector Machine for Network Intrusion Detection System

Preeti Yadav
Department of CSE
Barkatullah University Institute of Technology
Bhopal,India

Divakar Singh, PhD
Head of CSE Department
Barkatullah University Institute of Technology
Bhopal,India

ABSTRACT

The paper proposes a parallel SVM for detecting intrusions in computer network. The success of any Intrusion Detection System (IDS) is a complex problem due to its non-linearity and quantitative or qualitative traffic stream with irrelevant and unnecessary features. How to choose effective and key features of IDS is a very important topic in information security. Since the training data set size may be very large with a large number of parameters, which makes it difficult to handle single SVM therefore parallel LMM concept is proposed in this paper for distributing data files to n different sets of n different devices that reduce computational complexity, computational power and memory for each machine. The proposed method is simple but very reliable parallel operation SVM and can be used for large data files and unbalanced method also provides the flexibility to change depending on the size of the data file, the processor and the memory available on the various units. The proposed method is simulated using MATLAB and the result shows its superiority.

Keywords: Parallel Support Vector Machine, Binary Classification.

1. INTRODUCTION

Security is becoming a major problem because Internet applications grow. Current security technologies focus on Encryption, ID, and firewall and access control. But all these Technologies cannot guarantee flawless system may be increased Intrusion Detection. Ability to IDS include a wide variety of attacks in real time with accuracy Results are important. Patterns of user activities and audit Records are examined and attacks are located. IDS are classified based on their functionality, such as abuse detectors and detection of anomalies. Misuse detection system uses well defined patterns of attack, which are compared with the user behavior for intrusion detection. Typically, misuse detection easier than detecting anomalies, because it uses a rule-based or signature Comparison of methods.

Anomaly detection requires storage normal use behavior and acts based on audit data obtained operating system. Support Vector Machines (SVM) are classifiers which were originally designed for binary classification [5], [6] may be used for classify attacks. If binary SVMs are combined with a decision trees, we have Multi Class SVMs, which can classify four types of attacks, probing, DoS, U2R, R2L attacks and Normal data, and can be prepared in five classes anomaly detection. Our goal is to improve the training time, testing time and accuracy IDS using the hybrid approach.

The paper is organized as follows. Section 2 resents a brief review. Section 3 describes the basic principles of SVM. Section 4 deals with parallel SVM. In Section 5 we study the proposed method.

2. PREVIOUS WORK

Many schemes have been proposed in past for predicting disease and parallelization of SVM some of the techniques that helps in development of our concepts in writing this paper are discussed here. For disease prediction Wei Yu*, Tiebin Liu, Rodolfo Valdez, Marta Gwinn, Muin J Khoury [11] used data from the 1999-2004 National Health and Nutrition Examination Survey (NHANES) to develop and validate SVM models for two classification schemes: Classification Scheme I (diagnosed or undiagnosed diabetes vs. pre-diabetes or no diabetes) and Classification Scheme II (undiagnosed diabetes or pre-diabetes vs. no diabetes). The SVM models were used to select sets of variables that would yield the best classification of individuals into these diabetes categories. Mohammed Khalilia, Sounak Chakraborty and Mihail Popescu [12] employed the National Inpatient Sample (NIS) data, which is publicly available through Healthcare Cost and Utilization Project (HCUP), to train random forest classifiers for disease prediction. Since the HCUP data is highly imbalanced, we employed an ensemble learning approach based on repeated random sub-sampling. This technique divides the training data into multiple sub-samples, while ensuring that each sub-sample is fully balanced. We compared the performance of support vector machine (SVM), bagging, boosting and RF to predict the risk of eight chronic diseases. For parallel SVM the Yumao Lu and Vwani Roychowdhury [4] proposes A parallel support vector machine based on randomized sampling technique they modeled a new LP-type problem so that it works for general linear-nonseparable SVM training problems a unique priority based sampling mechanism is used so that we can prove an average convergence rate that is so far the fastest bounded convergence rate. Amit Maan et al.[5] introduce a distributed algorithm for solving large scale Support Vector Machines (SVM) problems. Their algorithm divides the training set into a number of processing nodes each running independently an SVM sub-problem associated with its subset of training data. The algorithm is a parallel (Jacobi) block-update scheme derived from the convex conjugate (Fenchel Duality) form of the original SVM problem. Each update step consists of a modified SVM solver running in parallel over the sub-problems followed by a simple global update. We derive bounds on the number of updates showing that the number of iterations (independent SVM applications on sub-problems) required to obtain a solution of accuracy ε is $O(\log(1/\varepsilon))$. The work proposed by Cheng-Tao Chu , Gary Bradski et el.[6] in their paper for a programming framework for processing with multicore processors in simple and unified way for machine learning to take advantage of the potential speed up. In paper, they develop a broadly applicable parallel programming method, one that is easily applied to many different learning algorithms. Our work is in distinct contrast to the tradition in machine learning of designing (often ingenious) ways to speed up a single algorithm at a time. Specifically, they show that algorithms that fit the Statistical Query model can be written in a certain "summation form," which allows them to be easily parallelized on multicore computers the proposed parallel speed up technique is tested on a variety of learning algorithms including locally weighted linear regression (LWLR), k-means, logistic regression (LR), naive Bayes (NB), SVM, ICA, PCA, gaussian discriminant analysis (GDA), EM, and backpropagation (NN) showing good results. To speed up the process of training SVM, another parallel methods have been proposed [7] by splitting the problem into smaller subsets and training a network to assign samples of different subsets. A parallel training algorithm on large-scale classification problems is proposed, in which multiple SVM classifiers are applied and may be trained in a distributed computer system. As an improvement algorithm of cascade SVM, the support vectors are obtained according to the data samples distance mean and the feedback is not the whole final output but alternating to avoid the problem that the learning results are subject to the distribution state of the data samples in different subsets. The experiment results on real-world text dataset show that this parallel SVM training algorithm is efficient and has more satisfying accuracy compared with standard cascade SVM algorithm in classification precision. The algorithm of Zanghirati and Zanni (2003) decomposes the SVM training problem into a sequence of smaller, though still dense, QP sub-problems. Zanghirati and Zanni implement the inner solver using a technique called variable projection method, which is able to work efficiently on relatively large dense inner problems, and is suitable for implementing in parallel. The performance of the inner QP solver was improved in Zanni et al. (2006). In the cascade algorithm introduced by Graf et al. (2005), the SVMs are layered. The support vectors given by the SVMs of one layer are combined to form the training sets of the next layer. The support vectors of the final layer are re-inserted into the training sets of the first layer at the next iteration, until the global KKT conditions are met. The authors show that this feedback loop corresponds to standard SVM training. The algorithm of Durdanovic et al. (2007), implemented in the Milde software, is a parallel implementation of the sequential minimal optimization.

3. PARALLEL SUPPORT VECTOR MACHINES

Penalization of any algorithm is a concept to arrange or partition the process of an algorithm such that it can be parallel processed on cluster of computers. In context of this particular paper we denoting Parallel SVM as the concept of partitioning a large training dataset into small data chunks and process each chunk in parallel utilizing the resources of a cluster of computers.

It's already clear from previous sections that training SVMs is computationally intensive and increases dramatically as the size of a training dataset increases. A SVM kernel usually involves an algorithmic complexity of O(m^2n), where n is the dimension of the input and m represents the training instances [3]. The computation time in SVM training is quadratic in terms of the number of training instances. Hence parallel approximate implementation to speed up SVM training on today's distributed computing infrastructures has proposed although the Parallel SVM is the sole solution to speed up SVMs. Algorithmic approaches such as (Lee & Mangasarian, 2001 [8]; Tsang et al., 2005; Joachims, 2006; Chu et al.,2006) [9], can be more effective when memory is not a constraint or kernels are not used.

SVM is based on creating a hyperplane as the decision plane, which separates the positive (+1) and negative (-1) classes with the largest margin. An optimal hyperplane is the one with the maximum margin of separation between the two classes, where the margin is the sum of the distances from the hyperplane to the closest data points of each of the two classes. These closest data points are called Support Vectors (SVs). Given a set of training data D, a set of points of the type

$$D = \{(x_i, c_i) \mid x_i \in R^p, c_i \in \{-1,1\}\}$$

Where c_i is either 1 or -1 indicative of the class to which the point x_i belongs, the aim is to give a maximum margin hyperplane which divide points having $c_i = 1$ from those having $c_i = -1$. Any hyperplane can be constructed as a set of point x satisfying w.x – b = 0.

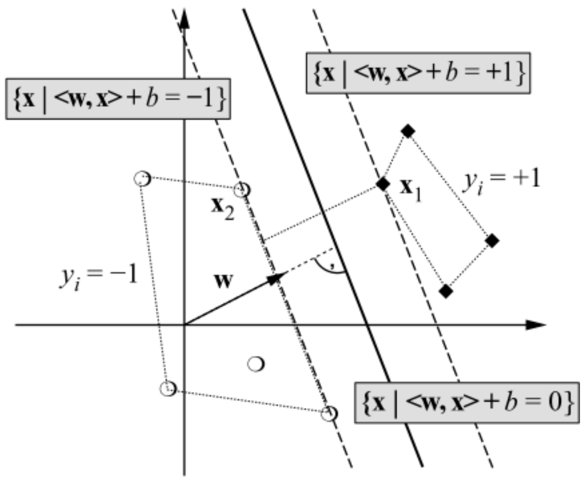

Figure 1. SVM, Visual description of separating hyperplane and support vectors.

The vector **w** is a normal vector. We want to choose **w** and b to maximize the margin. These hyperplanes can be described by the following equations:

$$w.x - b = 1$$

$$w.x - b = -1$$

The margin is given by

$$m = 1/\|w\|^2$$

The dual of the SVM is shown to be the following optimization problem:

Maximize (in α_i)

$$\sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j x_i x_j$$

Subject to

$$\alpha_i > 0 \; and \sum_{i=1}^{n} \alpha_i \, y_i = 0$$

y_i indicates the class of an instance, there is a one-to-one association between each Lagrange multiplier α_i and each training example x_i. Once the Lagrange multipliers are determined, the normal vector **w** and the threshold b can be derived from the Lagrange multipliers as follow:

$$\vec{w} = \sum_{i=1}^{n} y_i a_i \vec{x}_i$$

$$b = \vec{w}.\overrightarrow{x_k} - y_k$$

for some $a_k > 0$. Not all data sets are linearly separable. There may be no hyperplane exist that separates the positive (+1) and negative (-1) classes. SVMs can be further generalized to non-linear classifiers. The output of a non-linear SVM is computed from the Lagrange multipliers as follow:

$$u = \sum_{i=1}^{n} y_i a_i K(X_i, X) + b$$

Where K is a kernel function that measures the similarity or distance between the input vector X_i and the stored training vector X.

4. PROPOSED ALGORITHM

Penalization of SVM has been already discussed in section 4. The proposed algorithm also follows the concept developed in that section.

The proposed algorithm finds the minimum numbers of data points which represents the abstracts of large dataset this is performed by firstly partitioning the data points into n numbers of clusters the n depends upon the size of dataset, number of available processors, computational power of processors and available memory.

The first level partitioning is performed by linear division of input data. This process also helps in balancing the both class data. The complete step by step description of algorithm is given below

Algorithms steps

1. Let the positive and negative class datasets be D_P and D_N
2. Choose the set with minimum size let it is D_P
3. Calculate its centre point C_P
4. Calculate the distance of all vectors of (D_N) from C_P
5. Choose the n minimum distance vectors form D_N dataset, where n is the size of D_P
6. Name it D_{N1} now make a new data set D_{new} = (D_P + D_N)
7. Divide the D_{new} in N sections by K mans clustering
8. Calculate the Support Vectors of each sections S_{ij} with their Class L_{ij} , where i = {1,2,3…..N} and j = {P, N}
9. Train the final SVM using S_{ij} and L_{ij}.
10. Explanation of the algorithm
11. The data set for training having two classes only and having unequal sizes.

12. The steps 2, 3, 4 and 5 are used to eliminate the non useful vectors from dataset and also balance the classification dataset.

13. Divides the dataset in most similar N sets which are most difficult to classify when grouped together.

14. The calculation of support vectors from each section provides the abstracted information of all vector of that section with only a fewer vectors which reduces the load for final classifier.

Creation of final classifier for future classification.

5. RESULTS

The proposed algorithm is developed in MATLAB 7.5, R2007B and the simulation results are obtained by running it on intel P4 with 2 GB of RAM. The dataset is taken from “KDD99”.

Table 1: Results Comparison for Simple SVM and Parallel SVM

Data size	Training Ratio	Training Time(Sec.)	Matching Time(Sec.)	TPR	TNR	FPR	FNR	Acc.	Prec.	Recall	F-Measure	Technique
200	0.5	0.05	0.045	0.957	0.937	0.0627	0.043	0.947	0.938	0.957	0.947	SVM
200	0.5	0.01	0.031	0.949	0.943	0.0562	0.0508	0.946	0.945	0.9492	0.946	PSVM
400	0.5	0.1	0.049	0.967	0.938	0.0613	0.0325	0.953	0.940	0.9675	0.953	SVM
400	0.5	0.02	0.04	0.951	0.967	0.0327	0.0483	0.959	0.967	0.9517	0.959	PSVM
800	0.5	0.2	0.057	0.945	0.944	0.0555	0.0549	0.944	0.945	0.9451	0.944	SVM
800	0.5	0.04	0.046	0.964	0.952	0.0478	0.0356	0.958	0.954	0.9644	0.958	PSVM
1000	0.5	0.25	0.063	0.958	0.958	0.042	0.0417	0.958	0.958	0.9583	0.958	SVM
1000	0.5	0.05	0.051	0.965	0.959	0.0405	0.0347	0.962	0.960	0.9653	0.962	PSVM

6. CONCLUSION

The simulation results shows that proposed algorithm takes only 1/3 of time taken by normal SVM for training and this result is for single machine so expected results for multi machine case is dropped by n times where n is number of machines.
The proposed method also maintained the approximately same accuracy when compared with normal SVM, although it shows that dividing data into larger number of clusters decreases the accuracy but it could be controlled by selecting proper starting point and K-means clustering we leaved this work for future.

7. REFERENCES

[1] Sandya Peddabachigari, Ajith Abraham, Crina Grosan, Johanson Thomas. Modeling Intrusion Detection Systems Using Hybrid Intelligent Systems. Journal of Network and Computer Applications-2005.

[2] Jun GUO , Norikazu Takahashi, Wenxin Hu . An Efficient Algorithm for Multi-class Support Vector Machines. IEEE-2008.

[3] Latifur Khan, Mamoun Awad, Bhavani Thuraisingham. A new intrusion detection system using support vector machines and hierarchical clustering. The VLDB Journal DOI 10.1007/s00778-006-0002 , 2007.

[4] V. N. Vapnik. The nature of statistical learning theory. Springer-Verlag,New York. NY, 1995.

[5] Xiaodan Wang, Zhaohui Shi, Chongming Wu and Wei Wang. An Improved Algorithm for Decision-Tree-Based SVM. IEEE-2006.

[6] Pang-Ning Tan, Michael Steinbach, Vipin Kumar. Introduction to data mining. Pearson Education.

[7] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. Journal of Machine Learning Research, 2:265–292, 2001.

[8] YMahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani. A detailed analysis of KDD CUP'99 data set. IEEE-2009.

[9] ttp://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[10] C. W. Hsu, C. J. Lin. A comparison of methods for multiclass support vector machines. IEEE Trans. On Neural Networks, vol. 13, no. 2, pp.415-425, 2002.

[11] Snehal Mulay, P.R. Devale, G.V. Garje. Decision Tree based Support Vector Machine for Intrusion Detection. ICNIT-2010, unpublished.

[12] Lili Cheng, Jianpei Zhang, Jing Yang, Jun Ma. An improved Hierarchical Multi-Class Support Vector Machine with Binary Tree Architecture" 978-0-7695-3112- 0/08 2008 IEEE DOI 10.1109/ICICSE.2008

[13] Razieh Baradaran and Mahdieh HajiMohammadHosseini "Intrusion Detection System based on Support Vector Machine and BN-KDD Data Set", 7thSASTech 2013, Iran, Bandar-Abbas. 7-8 March, 2013.

[14] Sreeja M. S., Aarcha Anoop 'New Genetic Algorithm Based Intrusion Detection System for SCADA", International Journal of Engineering Innovation & Research Volume 2, Issue 2, ISSN: 2277 – 5668.

[15] Megha Bandgar, Komal dhurve, Sneha Jadhav,Vicky Kayastha,Prof. T.J Parvat "Intrusion Detection System using Hidden Markov Model (HMM)", IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 10, Issue 3 (Mar. - Apr. 2013), PP 66-70.

[16] Alma Cemerlic, Li Yang, Joseph M. Kizza "Network Intrusion Detection Based on Bayesian Networks", University of Tennessee at Chattanooga Chattanooga, TN 37403.

[17] S.A.Joshi, Varsha S.Pimprale "Network Intrusion Detection System (NIDS) based on Data Mining", International Journal of Engineering Science and Innovative Technology (IJESIT) Volume 2, Issue 1, January 2013.