UNWANTED BEHAVIOUR DETECTION AND CLASSIFICATION IN NETWORK TRAFFIC

İsmail Melih Önem
Department of Computer Science, Middle East Technical University Ankara, Ankara, Turkey

Keywords: Intrusion classifier, Intrusion detection, Category discovery, SVM, SVM kernel and parameter selection, SVM performance, SVM categorizing.

Abstract: An Intrusion Detection System classifies activities at an unwanted intention and can log or prevent activities that are marked as intrusions. Intrusions occur when malicious activity and unwanted behaviour gain access to or affect the usability of a computer resource. During the last years, anomaly discovery has attracted the attention of many researchers to overcome the disadvantage of signature-based IDSs in discovering novel attacks, and KDDCUP'99 is the mostly widely used data set for the evaluation of these systems. Difficulty is discovering unwanted behaviour in network traffic after they have been subject to machine learning methods and processes. The goal of this research is using the SVM machine learning model with different kernels and different kernel parameters for classification unwanted behaviour on the network with scalable performance. The SVM model enables flexible, flow-based method for detecting unwanted behaviour and illustrates its use in the context of an incident, and can forward the design and deployment of improved techniques for security scanning. Although scalability and performance are major considerations and results also are targeted at minimizing false positives and negatives. The classification matured in this paper is used for improving SVM computational efficiency to detect intrusions in each category, and enhanced model is presented experimental results based on an implementation of the model tested against real intrusions.

1 INTRODUCTION

With the immense growth of computer network usage and the huge rise in the number of applications running on top of it, network security is becoming more and more arrogant. Therefore, the role of Intrusion Detection System (IDSs), as special-purpose appliances to category anomalies in the network, is becoming further significant. The analysis in the intrusion detection and categorization field has been mostly focused on anomaly-based and misuse-based discovery techniques for a long time. While misuse-based discovery is generally preferred in commercial products due to its predictability and high accuracy, in academic research anomaly classification is typically formulated as a more powerful method due to its theoretical promising for turning to novel attacks.

Difficulty is discovering unwanted behaviour in network traffic after they have been subject to machine learning methods and processes. There is a great written works on various security methods to defend network objects from unauthorized use or disclosure of their private information and valuable assets. Even so, unconscious or automatic users find a way through much wiser means of get ridding of avoidance methods.

In usual methods located on port numbers and protocols have proven to be ineffective in terms of dynamic port allocation and packet encapsulation. The signature matching methods, on the other hand, require a known signature set and processing of packet payload, can only handle the signatures of a limited number of IP packets in real-time. A machine learning method based on SVM (supporting vector machine) is tendered in this paper for accurate classification and discovery unwanted behaviour with scalable performance. The method classifies the Internet traffic into broad application categories according to the network flow parameters obtained from the packet headers. An optimized feature set is acquired via various classifier selection methods.

Different SVM machine learning models are used for discovering unwanted behaviour on the network traffic. LIBSVM and Weka(Waikato

Önem Ãř..
UNWANTED BEHAVIOUR DETECTION AND CLASSIFICATION IN NETWORK TRAFFIC .
DOI: 10.5220/0003117501220128
In *Proceedings of the International Conference on Knowledge Discovery and Information Retrieval* (KDIR-2010), pages 122-128
ISBN: 978-989-8425-28-7

environment for knowledge analysis) is used in a Java environment for training and testing the learning algorithms. It provides several different SVM implementations along with multiple kernels. I examine three things, the relative importance of features in training the dataset, the choice of kernel algorithm and parameter selection of SVM classifiers. By understanding what features are the most relevant, the dataset can be trimmed to include only the most useful data. The choice of kernel results in different levels of errors when applied to the KDD Cup dataset (McHugh, 2000). Frameworks offer five different kernels: Sigmoid, Linear, Polynomial and RBF. Each kernel offers three parameters for tuning and optimization which values are "gamma, cost and nu".

The performance norm has also been the subject of mine research. Here, the best kernel should maximize a predictive performance criterion as well as a computational performance criterion. That is, I seek the best categorizers that are; good at discover unwanted behaviour, are efficient to compute over massive datasets of network traffic. I address the "predictive performance" criterion, what meaning by good, after describing the cost model for this domain.

2 CHALLENGES OF IMPROVING CATEGORIZING

The approach to this work is done in steps, with supplemental complexity being added to the model at each level. As a prelude to developing any models, the data must first be put into a usable format. I am using the KDDCup 99 dataset, delineated earlier, which includes of features that are either continuous (numerically) valued or discrete. The continuous features in the provided dataset are in the text format (i.e. tcp/udp) and must be transformed.

One of the primary challenges of intrusion discovery is gathering applicable data for training and testing of an algorithm. Lack of the KDD data set is the vast number of redundant records, which causes the learning algorithms to be biased towards the frequent records, and thus prevent them from learning rare records, which are usually more pernicious to networks. In addition, the existence of these repeated records in the test set will cause the evaluation results to be biased by the methods which have better categorizing rates on the frequent records.

One of the disadvantages of SVM-based and other supervised machine learning method is the requisite on a large number of labelled training samples (Yao, Zhao, and Fan, 2006). Furthermore, recognizing the traffic after the network flow is collected could be too late should security and interventions become necessary in the early stage of the traffic flow. My intend is using supervised machine learning methods, as well as using feature parameters obtainable in the traffic flow for fast and accurate Network traffic discovery.

Even though, the recommended data set still suffers from some of the problems in complex data set and may not be a perfect stand in of existing real networks, because of the lack of public data sets for network-based IDSs, at the same time it can be applied as an impressive benchmark data set to help researchers compare different machine learning methods.

3 PROBLEM DEFINITION AND SVM'S

Machine learning has large implications for intrusion discovery, because intrusions are becoming more complex and information systems are evenly become more intricate. By using machine learning techniques to analyze incoming network data, I can decide to determine malicious attacks before they compromise an information system. Research in the field of intrusion detection seems to focus on a variety of support vector machine method, neural networks and cluster algorithms.

Support vector machines are the correspondingly recent methods of machine learning based on structural risk minimization, and they are a powerful machine learning method for both arrested development and classification problems.

In this paper, I tried an effective approach to solve the two mentioned issues, resulting in new train and test sets, which consist of chosen records of the complete KDD data set. The provided data set does not suffer from a large number of tagged training samples. Besides, the numbers of records in the train and test sets are reasonable. This advantage makes it affordable to run the experiments which needed to randomly select a small portion. Inevitably, evaluation results of different research work will be consistent and comparable.

Through the use of correct kernel choice, feature selection and parameter selection, I have shown that it is possible to improve the accuracy and efficiency

of a Support Vector Machine applied to an Intrusion Detection Scenario.

Support vector machines (SVMs) are a set of related supervised learning methods used for classification and regression. In simple words, given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm intensifies a model that predicts whether a new example falls into one category or the other. Additionally, a support vector machine constructs a hyper plane or set of hyper planes in a high or infinite dimensional space, which can be used for discovery of unwanted behaviour in network traffic. SVMs use two key concepts to solve this problem: large-margin separation and kernel functions. Classification exercise usually involves separating data into training and testing sets. Each instance in the training set contains one "target value" and "several attributes".

Given a training set of instance-label pairs $(x_i, y_i), i = 1, \dots, l$ where $x_i \in 2\, R^n$ and $y \in \{1, -1\}^l$ the support vector machines (SVM) (Mahoney and Chan, 2003) require the solution of the following optimization problem:

$$\min_{w,b,\varepsilon} \quad \frac{1}{2}\mathrm{w}^{\mathrm{T}}\mathrm{w} + C\sum_{\mathrm{i}=1}^{\mathrm{l}} \varepsilon_{\mathrm{i}} \qquad (1)$$

$$\text{subject to} \quad y_i(w^t \Phi(x_i) + b) \geq 1 - \varepsilon_{\mathrm{i}}$$
$$\varepsilon_{\mathrm{i}} \geq 0$$

Training vectors x_i are mapped into a higher (maybe infinite) dimensional space by the function Φ. SVM finds a linear separating hyper plane with the maximal margin in this higher dimensional space. $C > 0$ is the penalty parameter of the error term. Furthermore, $\mathrm{K}(x_i, x_j) \equiv \Phi(x_i)^T \Phi(x_j)$s called the kernel function. γ , r, d is kernel parameters.

Linear Kernel: The simpler kernel achieves to making a classification decision based on the value of a linear combination of the characteristics.

$$K(x_i, x_j) = x_i^T x_j. \qquad (2)$$

Polynomial Kernel: This kind of kernel represents the inner product of two vector(point) in a feature space of multi-dimension.

$$K(x_i, x_j) = (\gamma x_i^T, x_j + r)^d, \gamma > 0. \qquad (3)$$

Radial Basis Function (RBF) Kernel: Nonlinearly maps samples into a higher dimensional space so it, unlike the linear kernel, can handle the case when the relation between class labels and attributes is nonlinear.

$$K(x_i, x_j) = exp\left(-\gamma \|x_i - x_j\|^2\right), \gamma > 0. \qquad (4)$$

Sigmoid Kernel: A SVM model using a sigmoid kernel function is equivalent to a two-layer, perceptron neural network.

$$K(x_i, x_j) = tanh(\gamma x_i^T, x_j + r)^d \qquad (5)$$

4 APPROACH, CLASSIFICATION

4.1 Data Analysis and Data Partitioning

As I mentioned before, there are some difficulties in the KDD data set, which cause the estimation results on this data set to be deceptive. In this section I perform a set of tests to show the existing deficiencies in KDD.

First steps of executions are on partitioning because of using large volume of data. Before building a model, typically I separate the data using a partition utility. Partitioning produces mutually different datasets of attack types. The five traffic categories are "normal, probe, denial of service (DoS), user-to-root (U2R), remote-to-local (R2L)".

For this purpose, the data set is divided into five segments *(Attacks and Normal Traffic)*, where the observations which I perform a set of experiments. That shows the existing deficiencies on the portion of the data set and is then evaluated.

The *DoS* attack data set is divided into five segments too, where the observations which I perform a set of experiments to show the existing deficiencies on the portion of the data set and is then evaluated.

The consequential deficiency in the KDD data set is the vast number of redundant records, which causes the learning algorithms to be biased towards the frequent records, and thus prevent them from learning unfrequented records which are usually more harmful to networks such as U2R and R2L attacks. In addition, the existence of these repeated records in the test set will cause the evaluation results to be biased by the methods which have better classification rates on the frequent records.

The typical approach for performing anomaly discovery using the KDD data set is to employ a customized SVM machine learning algorithm to learn the general behavior of the data set in order to be able to differentiate between normal and malicious activities and randomly shuffles the order of all instances passed through it.

Table 1: Dataset Redundancy.

Class	Redundant	Invalids	Unique	Ratio %
Normal	159967	2	812814	83,56
DoS	3636103	0	247267	6,37
U2R	0	0	52	100,00
R2L	127	0	999	88,72
Probe	27242	0	13860	33,72

However, detailed classification are not of much interest in this paper since most of the anomaly detection systems work with binary labels, i.e., anomalous and normal, rather than identifying the detailed information of the attacks.

Table 2: Dataset subsampling.

Class	Inst. Count	Taken	Ratio %
Normal	812814	67943	8,36
U2R	52	52	100,00
R2L	999	999	100,00
Probe	13860	13860	100,00
DoS-Back	968	968	100,00
DoS-Land	19	19	100,00
DoS-Neptune	242149	13626	5,63
DoS-Pod	206	206	100,00
DoS-Smurf	3007	3007	100,00
DoS-TearDrop	918	918	100,00
Total:	1074992	101598	9,45

4.2 Feature Selection

There are several inciting influences behind limiting the feature set of the intrusion data for the SVM. A smaller feature set may result in considerably improved training and classification timing. Hanging on the anomaly discovery application, timing may be critical. Supplementally, some features may not truly relate to the intrusion classification results and should be excluded.

There are also more methodical approaches. From a theoretical perspective, it can be shown that optimal feature selection for supervised learning problems requires an exhaustive search of all possible subsets of features of the chosen cardinality. If large numbers of features are available, this is impractical. For practical supervised learning algorithms, the search is for a satisfactory set of features instead of an optimal set.

Most methods for attribute selection involve searching the space of attributes for the subset that is most likely to predict the class best. For optimal set choosing, I combine three satisfactory methods which normalize the attribute ranking. One way to accelerate the search process is to stop evaluating a subset of attributes as soon as it becomes apparent that it is unlikely to lead to higher accuracy than another candidate subset. This is a job for a paired statistical significance test, performed between the classifier based on this subset and all the other candidate classifiers based on other subsets.

Gain Ratio Attribute, evaluates the worth of an attribute by measuring the gain ratio with respect to the class, ranks attributes by their individual evaluations.

Info Gain Attribute, evaluates the worth of an attribute by measuring the information gain with respect to the class, ranks attributes by their individual evaluations.

CFS (Boser, Guyon and Vapnik, 1992), is a simple filter algorithm that ranks feature subsets according to a correlation based heuristic evaluation function. Irrelevant features should be ignored because they will have low correlation with the class. Redundant features should be screened out as they will be highly correlated with one or more of the remaining features. The acceptance of a feature will depend on the extent to which it predicts classes in areas of the instance space not already predicted by other features. M_S is the heuristic "merit" of a feature subset S containing.

C is the class attributes and the indices j range over all attributes in the set. U is the gain value when gain ratios when selection done.

$$\sum_j U(a_j, C) \Big/ M_s \tag{6}$$

Table 3: Feature Selection Result.

Ga.Ra.A		Inf.Ga.A		Selection Result		CFS
Gn.	A.	Gn.	A.	Tot.	Att.	Att. Name
0,69	3	0,26	3	0,95	service	service
0,74	5	0,19	5	0,93	src_bytes	dst_bytes
0,44	12	0,45	12	0,88	logged_in	wrong_fr.
0,62	6	0,23	6	0,84	dst_bytes	logged_in
0,35	30	0,28	30	0,64	diff_srv_rt	srv_ser_rt
0,36	29	0,27	29	0,64	same_s._rt	same_s._rt
0,35	4	0,29	4	0,64	flag	
0,23	26	0,36	26	0,59	srv_ser._rt	
0,26	25	0,31	25	0,57	serror_rate	
0,42	33	0,14	33	0,56	dst_h._s.c.	
0,41	23	0,12	23	0,53	count	

Selected feature set is “service, src, bytes, logged_in, dst_bytes, diff_srv_rate, same_srv_rate, srv_serror_rate”.

4.3 Kernel Selection

Another crucial issue for support vector machines is choosing the kernel function. Kernels introduce different nonlinearities into the SVM problem by mapping input data X implicitly into hypothesis space via a function Φ where it may then be hyper plane separable.

However, searching for different kernels either via trial-and-error or other exhaustive means can be a computationally higher one.

Weka (Waikato environment for knowledge analysis) provides several different SVM implementations along with multiple kernels via standard parameters. I examine two things, the relative importance of features in training the dataset and the choice of kernel algorithm. By understanding what features are most relevant, the dataset can be trimmed to include only the most useful data.

Random sub-sampling, validation method randomly splits 66% of dataset into training and %33 for validation data. For each such split, the model is fit to the training data, and predictive accuracy is assessed using the validation data.

Table 4: Kernel Accuracies.

Kernels	Accuracy (%) with selected attributes	Accuracy (%)
Linear	86.77	73.35
Polynomial	58.57	33.23
RBF	99.67	97.71
Sigmoid	66.76	78.25

4.4 Parameter Selection

The first issue is deciding how to evaluate the parameter’s effectiveness, which is just the standard problem of evaluating machine learning method performance. Methods require that the machine learning engine be trained multiple times in order to obtain a single performance number for a single parameter setting.

The most common and reliable approach to parameter selection is to decide on parameter ranges, and to then do an exhaustive grid search over the parameter space to find the best setting. Unfortunately, even moderately high resolution searches can result in a large number of evaluations and unacceptably long run times.

Approach is to start with a very coarse grid covering the whole search space and keeping the number of samples at every iteration, constant. I compare the performance of the proposed search method by changing SVM type and gamma value for selected Kernel using LIBSVM and the RBF kernel, both in terms of the quality of the final result and the work required to obtain that result.

By storing the search parameter bounds in a list, the searching itself is independent of the number of parameters in the increasing space. This allows us to re-use the same parameters and code for all four kernels.

For both search methods, the parameter ranges are:

C-SVC

- $\log_\gamma\{0, \ldots., -8\}$
- $\log_C * 10\{0, \ldots.,70\}$

nu-SVC

- $\log_\gamma\{0, \ldots., -5\}$
- $\text{nu} * 100\ \{10, \ldots.,70\}$

Results of search are:

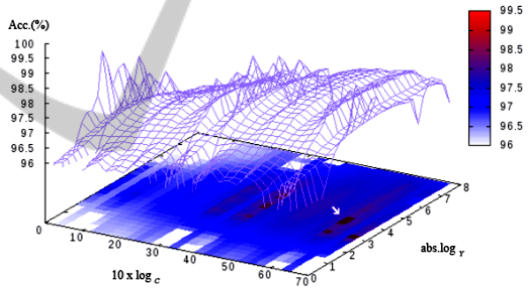

Figure 1: C-SVC Distribution of Parameter Search.

Table 5: C-SVC Parameter Selection Results.

C	Gamma	Accuracy (%)
1000000	5,00E-05	99,867
10000	5,00E-05	99,801
50000	5,00E-05	99,801

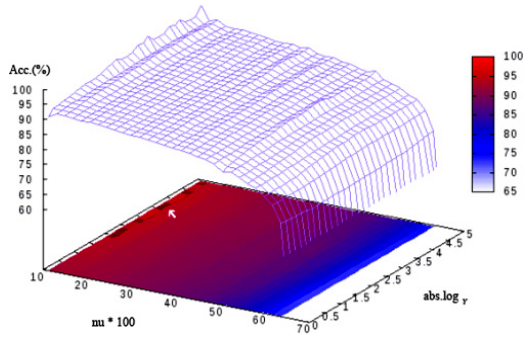

Figure 2: nu-SVC Distribution of Parameter Search.

Table 6: nu-SVC Parameter Selection Results.

nu	gamma	Accuracy(%)
0,11	0,001	98,740
0,11	5,00E-04	98,740
0,11	9,00E-04	98,740
0,11	1,00E-04	98,277

5 RESULT AND DISCUSSION

Firstly, final data set has the following advantages over the original KDD data set. It does not include redundant records in the train set, so the classifiers will not be biased towards more frequent records. There are no duplicate records in the proposed test sets; therefore, the performances of the learners are not biased by the methods which have better classification rates on the frequent records. The numbers of records in the training and test sets are reasonable, which makes it affordable to run the experiments on the complete set without the need to randomly select a small portion. Consequently, evaluation results of different research works will be consistent and comparable.

Table 7: RBF overall accuracy.

SVM Type	Accuracy (%) With selected params.	Classification
C-SVC	99.66	34594/34711
nu-SVC	99.00	34365/34711

Similarly, the best subset features to be trained on can be successfully identified using the parametric methods described above. By combining the kernel, feature and parameter selection, I arrive at an improved version of the algorithm. This more quickly and more accurately predicts the safety of network traffic.

Table 8: Performance measures after selections.

SVM Type	Running time of algorithm (sec.)	Running time of algorithm after improvements. (sec.)
C-SVC	12.631	1.203
nu-SVC	11.402	0.8902

After the finding ratios in these tables, I decided the way to faster detection for network intrusions while protecting a computer network from unauthorized users, including perhaps insiders etc. The learning task about classification is to build a predictive model (a categorizer) capable of distinguishing between *bad* connections, called intrusions or attacks, and *good* normal connections.

6 CONCLUSIONS

Through the use of correct kernel choice and feature selection, I think that I have shown that it is possible to improve the accuracy and efficiency of a Support Vector Machine applied to an Intrusion categorizing scenario. The choice of kernel should be made for correct the superior results. Similarly, the best subset features to be trained on can be successfully identified using the parametric methods. By combining the kernel, feature and parameter selection, I arrived at an improved version of the algorithm. This is more quickly and more accurately predict the safety of network traffic.

There are some critiques of attack taxonomies and performance measures. However, attack taxonomies are not of much interest in this paper since most of the anomaly categorizing systems work with binary labels, i.e., anomalous and normal, rather than identifying the detailed information of the attacks.

The number of records in the train and test sets is reasonable, which makes it affordable to run the experiments on the complete set. Consequently, evaluation results of different research on different subsets works are consistent and comparable. Based on my approach, I gained performance results that indicate that our approximation of using SVM to represent and detect computer intrusions is workable. Intrusion detection systems have gained not popular acceptance, mainly because of their space requirements and the performance impact that is suffered while running them with regular system activity. I have displayed that it is workable to run an intrusion detection system based on improved SVM method, concurrently with other user activities on multi-user networks, without superfluous degradation in performance.

The proposed methods can be applied to encrypted network traffic, since it does not rely on the application payload for classification. Furthermore, as all the feature parameters are computable without the storage of multiple packets, the method lends itself well for real-time traffic identification.

Currently this work is being extended to work across each of the SVM kernels supported by LIBSVM, but some care must be taken to ensure

that the search is robust in the face of infeasible solutions which are more likely with some of the other kernels. In addition, a kernel-searching is being added on top of the per-kernel parameter search so the system can automatically identify the best kernel and its parameter settings.

REFERENCES

McHugh, J., 2000. *Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory.* ACM Transactions on Information and System Security, vol. 3, no. 4, pp. 262–294

Yao, J. T., Zhao, S., Fan, L., 2006. *Advanced Support Vector Machine Model for Intrusion Detection. Lecture Notes in Computer Science.* Springer-Berlin. 538-543

Mahoney, M., Chan, P., 2003. *An Analysis of the 1999 DARPA/Lincoln Laboratory Evaluation Data for Network Anomaly Detection.* LECTURE NOTES IN COMPUTER SCIENCE. 220–238

Boser, E., Guyon I., Vapnik, V., 1992. *A training algorithm for optimal margin classifiers. In Proceedings of the Fifth Annual Workshop on Computational Learning Theory.* ACM Press. 144-152

APPENDIX

Attack Hierarchy.

back	dos	perl	u2r
buffer_overflow	u2r	phf	r2l
ftp_write	r2l	pod	dos
guess_passwd	r2l	portsweep	probe
imap	r2l	rootkit	u2r
ipsweep	probe	satan	probe
land	dos	smurf	dos
loadmodule	u2r	spy	r2l
multihop	r2l	teardrop	dos
neptune	dos	warezclient	r2l
Nmap	probe	warezmaster	r2l

Traffic features of individual TCP connection.

feature name	description	type
duration	length (number of seconds) of the connection	continuous
protocol_type	type of the protocol, e.g. tcp, udp, etc.	discrete
service	network service on the destination, e.g., http, telnet, etc.	discrete
src_bytes	number of data bytes from source to destination	continuous
dst_bytes	number of data bytes from destination to source	continuous
flag	normal or error status of the connection	discrete
land	1 if connection is from/to the same host/port; 0 otherwise	discrete
wrong_fragment	number of ``wrong'' fragments	continuous
urgent	number of urgent packets	continuous
count	number of connections to the same host as the current connection in the past two seconds	continuous
serror_rate	% of connections that have ``SYN'' errors	continuous
rerror_rate	% of connections that have ``REJ'' errors	continuous
same_srv_rate	% of connections to the same service	continuous
diff_srv_rate	% of connections to different services	continuous
srv_count	number of connections to the same service as the current connection in the past two seconds	continuous
srv_serror_rate	% of connections that have ``SYN'' errors	continuous
srv_rerror_rate	% of connections that have ``REJ'' errors	continuous
srv_diff_host_rate	% of connections to different hosts	continuous