A Cost-based Approach for Fast Intrusion Detection

Jiarui Gao[1]
[1]School of Computer Science, Shanghai Key Laboratory of Intelligent Information Processing,
Fudan University, China
jrgao14@fudan.edu.cn

ABSTRACT

The aim of Intrusion Detection System(IDS) is to maximize detection accuracy as well as minimize corresponding costs. In this paper, I present a cost-based approach utilizing neural network with GPU acceleration for fast intrusion detection. In this approach, both computational cost and time cost are considered, and models are trained with respect to their different protocol types and services. Empirical experiments are carried out on off-line benchmark dataset NSL-KDD.

KEYWORDS

Neural network; Intrusion detection; anomaly detection; cost analysis.

ACM Reference format:
Jiarui Gao[1]. 2017. A Cost-based Approach for Fast Intrusion Detection . In *Proceedings of* , , , 4 pages.
DOI: http://dx.doi.org/xxx

1 INTRODUCTION

With the development of computing and Internet technology, Internet has became an important part of our daily life. However, this popularity has also brought security issues. Intrusion Detection Systems(IDSs) are designed to detect attacks, which help discover, determine, and identify unauthorized use, duplication, alteration, and destruction of information systems[7]. Specifically, IDSs are able to monitor intrusions and alert network administrators if necessary.

According to the detectable attacks, there are mainly three types of IDSs: misuse-based, anomaly-based, and hybrid[1]. Misuse-based IDSs are designed to detect known attacks by comparing signatures of those attacks and those in the database. Anomaly-based IDSs have the ability for detecting zero-day attacks. The normal network are modeled and anomalies can be detected when happened.

Also with respect for using environment, IDSs can be used in real-time environment of off-line environment. Off-line IDSs have been studied extensively recently. However, these systems fail to provide real time information, which means they can only detect intrusions after they already happened. The challenge for real-time IDSs is the huge cost brought with the detection.

In this paper, the aim is to develop a IDS which can be utilized in real-time environment with high detection accuracy. I present a cost-based approach for anomaly intrusion detection. The traditional machine learning models are replaced by neural networks, which enable the computational cost to be minimized utilizing GPU acceleration. Also, the time cost is considered, and according to the time cost theory in [5], low level features are utilized with high priority. The benchmark dataset NSL-KDD is used for evaluation, which is a new version of dataset KDDCUP'99[3] and has significant improvements compared with KDDcup99[8].

The rest of the paper is organized as follows. Sec 2 gives a detailed review of current related work about intrusion detection. In Sec 3, the new approach for intrusion detection is introduced. The experimental results are reported in Sec 4. More future work is presented in Sec 5. Finally, a conclusion is drew in Sec 6.

2 RELATED WORK

2.1 Detection Methodology

2.2 Machine Learning for Intrusion Detection

2.3 Public Datasets

3 METHODOLOGY

As shown in Fig 1, a multi-model detector is presented here.

3.1 Cost Analysis

Computational Cost. One of the major challenges in real-time IDSs is the computational cost attached with the system. For example, the well-known K-nearest Neighbor Classifier[6] performs well on off-line datasets, but it is not an appropriate detector in real time. Because the KNNC calculates the similarity between the new process and each process in the training set to decide the label of the new process. So KNNC is computationally expensive when the incoming number of processes is huge.

Traditionally, less features and models with less parameters will lead to low computational cost. However, this may also result in lower detection accuracy or higher false alarm rate. Different from former approaches, utilizing deep neural networks, the computing tasks can be accelerated on GPU[2]. Also, the ability to represent any non-linear function in very few hidden layers and hidden neurons make the computational expense even less.

Time Cost. It is obvious that the time delay for retrieving features simply from a network packet and extracting information from all former similar connections within a time period are quite different. Different from[5], the connection based features are utilized according to[5] in this paper.

- Level 1 features are basic features of individual TCP connections.

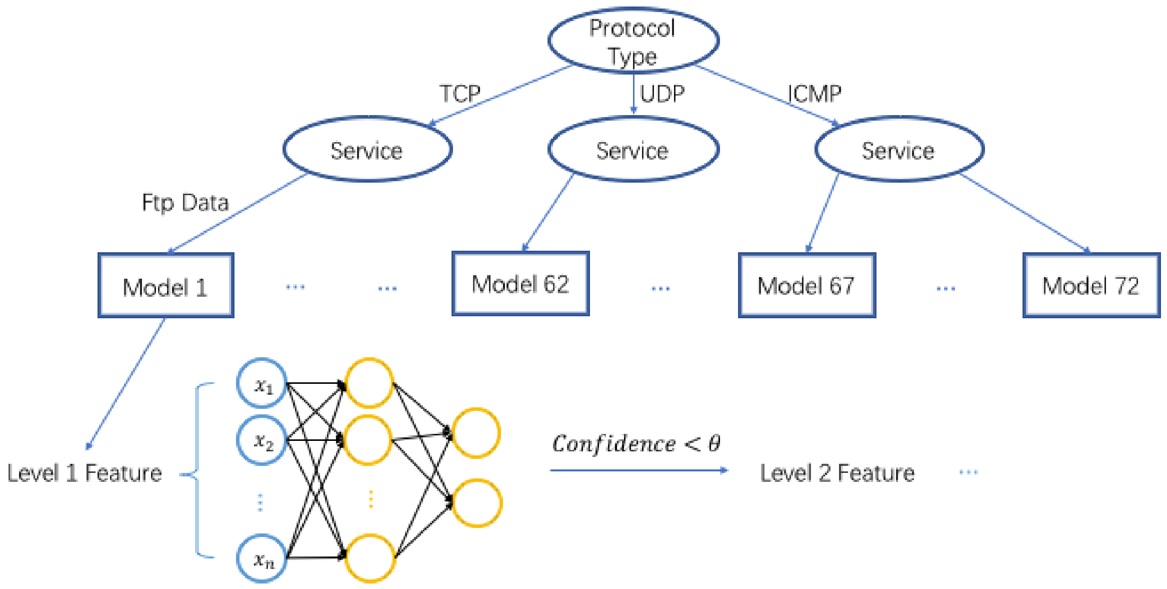

Figure 1: An overview of the detection framework including the decision tree(top part) and the neural networks(bottom part). The total number of models are calculated on NSL-KDD dataset.

- Level 2 features are content features within a connection suggested by domain knowledge.
- Level 3 features are traffic features computed using a two-second time window.

Notice that the Level 3 features require iteration from the entire former two-second packet set, the time cost is extremely high. Also, all the Level 3 features for connections coming at the same time can also be computed at the same time.

The lower level features are first used, and if the confidence of the classification is less than a threshold θ, higher level features are extracted and are used combined with other lower level features. For example, both Level 1 and Level 2 features are considered after extracting the Level 2 features.

3.2 Neural Network Detectors

Packets with different protocol types and different services tend to have different signatures to be malicious. So neural network models are trained for each protocol type and service respectively. Thus for each new connection, its protocol type and service decides which detector to use. This model selection framework can be modeled by a decision tree.

Neural network models with only one hidden layer is utilized in this paper, and the following experiments proves their capability for intrusion detection with such simple architectures.

4 EXPERIMENTS

4.1 Dataset

As mentioned in Sec 2.3, NSL-KDD dataset consists of the selected records of the KDDCUP'99 dataset, which is more challenging to detect compared with the original widely used benchmark KDD-CUP'99 dataset. The NLS-KDD dataset solves the issues of models trained on KDDCUP'99 dataset, which have high seemingly accuracy but actually poor capability of anomaly detection.

All the features in NSL-KDD dataset are considered in this paper except the additional feature: detection difficulty, which is generalized by statistical analysis for the KDDCUP'99 dataset[8]. Totally, there are 61 different TCP services, 5 different UDP services and 6 different ICMP services.

4.2 Parameter setting

All the code is implemented in Python utilizing deep learning framework Tensorflow, and the neural network models are trained on one GPU GeForce GTX TITAN X for 5000 epochs with early stop policy. All the neural networks consist of 1 hidden layer, and the number of hidden neurons are set specifically for every model. The initial learning rate is $\eta = 0.0001$ and dropout ratio is 0.75. Finally, a softmax layer with two output values is utilized to calculate loss and Adam[4] is used to accelerate learning.

4.3 Baseline

Support Vector Machine(SVM) is utilized as a baseline method for each model respectively. The code are implemented in Python on Scikit-learn Library. And the following baseline results are evaluated on the same server with the proposed approach without GPU support for SVM method.

4.4 Detection Results

Due to the coming deadline issue, I have only finished part of the experiments.

Detection Accuracy. Tab 1 demonstrates the detection accuracy for each level of features. And apart from the over all accuracy, I take three representative protocol type and service(TCP finger/UDP domain_u/ICMP eco_i) as examples.

Observed from Tab 1, we can see that SVM models and neural network models perform nearly the same on most of the detections. But neural network models demonstrate a good property that as the level of features get higher, the accuracies never decay; while the SVM method present less stability.

Also, we can observe that the decision tree idea that to detect different protocol types and service with different models is quite reasonable. In Tab 1, huge accuracy improvement has been made thanks to this separation.

Cost Analysis. Tab 2 demonstrates the **computational cost** for each level of features. The same as former experiments, three representative protocol type and service are selected. Notice here I report the average time for computing single one packet feature item.

It can be noticed that significant improvement have been on computational cost utilizing neural network models, which is quite important in real time detection ability for the IDSs.

Also, the **time cost** is considered. Each new packet are first detected according to their Level 1 features. If the confidence of the classification, which is the larger value of the two output neurons is less than $\theta = 0.8$, than the higher features are extracted and higher level models are used. In the worst case, the packet will eventually get its label through Level 3 models. And more additional computational cost will also be spent.

The total cost is the sum of computational cost and time cost, which is much more efficient compared with simply using Level 3 features for all the packets.

5 FUTURE WORK

6 CONCLUSION

REFERENCES

[1] Anna L Buczak and Erhan Guven. 2016. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials* 18, 2 (2016), 1153–1176.

[2] Sharan Chetlur, Cliff Woolley, Philippe Vandermersch, Jonathan Cohen, John Tran, Bryan Catanzaro, and Evan Shelhamer. 2014. cudnn: Efficient primitives for deep learning. *arXiv preprint arXiv:1410.0759* (2014).

[3] KDD Cup. 2007. Available on: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. (2007).

[4] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[5] Wenke Lee, Salvatore J Stolfo, Philip K Chan, Eleazar Eskin, Wei Fan, Matthew Miller, Shlomo Hershkop, and Junxin Zhang. 2001. Real time data mining-based intrusion detection. In *DARPA Information Survivability Conference & Exposition II, 2001. DISCEX'01. Proceedings*, Vol. 1. IEEE, 89–100.

[6] Yihua Liao and V Rao Vemuri. 2002. Use of k-nearest neighbor classifier for intrusion detection. *Computers & security* 21, 5 (2002), 439–448.

[7] Srinivas Mukkamala, Andrew Sung, and Ajith Abraham. 2005. Cyber security challenges: Designing efficient intrusion detection systems and antivirus tools. *Vemuri, V. Rao, Enhancing Computer Security with Smart Technology.(Auerbach, 2006)* (2005), 125–163.

[8] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A Ghorbani. 2009. A detailed analysis of the KDD CUP 99 data set. In *Computational Intelligence for Security and Defense Applications, 2009. CISDA 2009. IEEE Symposium on.* IEEE, 1–6.

Table 1: Detection Accuracy Results.

Methods/Features	Over all	TCP finger	UDP domain_u	ICMP eco_i
Level 1 Features				
SVM	78.20%	99.26%	99.33%	94.27%
Ours	77.26%	99.26%	99.33%	94.27%
Level 2 Features				
SVM	81.39%	97.06%	99.55%	94.27%
Ours	80.67%	99.26%	99.78%	94.27%
Level 3 Features				
SVM	80.57%	99.26%	99.78%	98.09%
Ours	76.24%	99.26%	99.78%	98.89%

Table 2: Computational Cost Results.

Methods/Features	Over all	TCP finger	UDP domain_u	ICMP eco_i
Level 1 Features				
SVM	$4.2 * 10^{-4}s$	$4.4 * 10^{-6}s$	$9.2 * 10^{-7}s$	$5.7 * 10^{-7}s$
Ours	$5.4 * 10^{-8}s$	$2.7 * 10^{-6}s$	$4.3 * 10^{-7}s$	$1.4 * 10^{-6}s$
Level 2 Features				
SVM	$5.9 * 10^{-4}s$	$7.3 * 10^{-6}s$	$1.3 * 10^{-6}s$	$1.3 * 10^{-6}s$
Ours	$7.0 * 10^{-8}s$	$2.7 * 10^{-6}s$	$4.6 * 10^{-7}s$	$1.4 * 10^{-6}s$
Level 3 Features				
SVM	$2.3 * 10^{-4}s$	$7.2 * 10^{-6}s$	$4.1 * 10^{-6}s$	$6.2 * 10^{-6}s$
Ours	$1.0 * 10^{-7}s$	$2.8 * 10^{-6}s$	$5.2 * 10^{-7}s$	$1.5 * 10^{-6}s$