CS536 Project Report

Mohit Gupta and Jawad Ahmed Raheel

December 2018

1 Abstract

With the exponential growth of devices that connect to some network like the internet have also come the commensurate increase in security threats. Detection of such threats is the first step toward defending against them. As many of these attacks are increasingly sophisticated, it is desirable to have an intelligent network defense system. Machine learning has been dutifully employed to this end and has proven effective in detecting such malicious intrusion into the network. In this study, we apply Decision Trees (DT), Support Vector Machine (SVM), and Artificial Neural Network (ANN) classification models from machine learning on the KDD Cup 1999 dataset from the UCI KDD archive and analyze their comparative performance.

2 Introduction

The days when devices connecting to internet were just workstations. Today hand-held mobile devices like cell phones and mobile computers, to even household devices like toasters with IoT and smart home-controllers like Amazon's Alexa connect to the internet, and home networks. These devices easily number in the billions. With the explosive growth in these internet devices, the threats to security and privacy have grown manifolds too, both in number and sophistication. Indeed just a single threat would take many human hours of analysis and testing before a solution could be implemented. This has led to *manual* defenses anything but practical. Detection and identification of such threats is the first step toward analysis and solution. Some of the common malicious attacks networks (internet or otherwise) face are Denial of Service (DoS), Remote to Local (R2L), User to Root (U2R), and Probing [Naseer et al. 2018, Ahmed et al. 2018, Mukkamala et al. 2002].

With the improvements in computing technology, along with improvements in data collection and storage, it has become feasible to apply artifical intelligence and machine learning to the problems of intrusion detection [Dennin et al. 1987, Naseer et al. 2018] to classify various users and requests-for-service as normal or malicious.

3 Background

Networks today consist of laptop and desktop computers, hand-held devices like smart phones, tablet computers, smart appliances like home controllers among other things. All these would henceforth be referred to as hosts. A host can request another host for a service or data which would generate traffic (called messages) over the network in accordance with communication protocols. An attack can be thought of a malicious host doing one or more of the following:

1. **eavesdrop**: a malicious host can intercept communication between other hosts and gain access to confidential information
2. **edit**: a malicious host can insert or delete or edit the communication between hosts leading to miscommunication, fraud, forgery, etc.
3. **hijacking & impersonation**: a malicious host can hijack a connection whereby they cut one or more of the communicating hosts out of the communication loop and take their place
4. **denial of service**: a malicious host can deny the service from being used by other hosts, usually by overwhelming the network.

An intrusion in a network is defined as any anomaly happening in the network. One way by which this can occur is through the above mentioned attacks from a malicious host.
In this work, we have data with 4 different kinds of attacks i.e. DOS (Denial of Service) attack, U2R (User to Root), R2L (Remote to Local) attack and Probing.

1. **DOS**: as defined above, in a DOS attack, a malicious host can deny the service from being used by other hosts, usually by overwhelming the network.
2. **U2R**: a malicious host tries to take root access of the service, usually by exploiting security vulnerabilities and loop holes in the network.
3. **R2L**: a malicious host sends packets to a machine over the internet, which it does not have access to in order to expose the machines vulnerabilities and exploit privileges which only a local user is supposed to have on the host.
4. **Probing**: a malicious host scans server machine or networking device in order to determine weaknesses or vulnerabilities that it can exploit.

In this work, intrusion detection (anomaly detection) is modelled as a classification problem in supervised learning. Supervised learning is one of the most popular machine learning methods. Supervised learning uses labeled data to train anomaly detection models. The goal of this type of training is to classify the test data as anomalous or normal on the basis of feature vectors. Unsupervised learning, on the other hand, uses unlabeled or untagged data to perform

the learning task. In this work, our primary focus is on supervised learning and we use such methods to detect an anomaly. Also, there may be examples, where there is no attack taking place and therefore, it is a normal network call. We tag such calls as **normal**.

An artificial neural network is a network of simple elements called artificial neurons, which receive input, change their internal state (activation) according to that input, and produce output depending on the input and activation. Artifical Neural Networks have been around for many decades and have been gaining and losing the favor of research community. Application of Deep Neural Networks for the solution of Information security problems is a relatively new area of research. We tested with a few settings of the number of hidden layers and neurons in each layer, and then settled for the simplest ANN that gave us high accuracy and F-1 scores - which for us consisted of two hidden layers, having fifty and thirty neurons respectively. The input layer had sixteen neurons, and the output layer was composed of four.

In this work, we focused on comparing ANN based approach with traditional machine learning methods notably Support Vector Machine (SVM) model and Decision Tree Classifier. We opted to train all models on training dataset without ever exposing test dataset to the model during training and then tested/evaluated the models on testing datasets. This approach provided a fair estimate of capabilities of the above mentioned models by using unseen data instances at evaluation time.

For the purpose of evaluation, we use the metrics defined below by Joshi [Joshi. 2016]:

1. **True Positives (TP)** - These are the correctly predicted positive values which means that the value of actual class is yes and the value of predicted class is also yes.

2. **True Negatives (TN)** - These are the correctly predicted negative values which means that the value of actual class is no and value of predicted class is also no.

3. **False Positives (FP)** – When actual class is no and predicted class is yes.

4. **False Negatives (FN)** – When actual class is yes but predicted class in no.

5. **Accuracy** - Accuracy is a ratio of correctly predicted observation to the total observations. One may think that, if we have high accuracy then our model is best. Yes, accuracy is a great measure but only when you have symmetric datasets where values of false positive and false negatives are almost same. Therefore, you have to look at other parameters to evaluate the performance of your model.

   Accuracy = TP+TN/TP+FP+FN+TN

6. **Precision** - Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. High precision relates to the low false positive rate.

   Precision = TP/TP+FP

7. **Recall (Sensitivity)** - Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes.

   Recall = TP/TP+FN

8. **F1 score** - F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it's better to look at both Precision and Recall.

   F1 Score = 2*(Recall * Precision) / (Recall + Precision)

All experiments in this study are performed on NSLKDD dataset. NSLKDD is derived from KDDCUP99 which was generated in 1999 from the DARPA98 network traffic. Tavallaee et al. discovered some inherent flaws in original KDDCUP99 dataset which had adverse impacts on the performance of IDS models trained and evaluated on the Dataset. A statistically enhanced version of dataset called NSLKDD was proposed to counter discovered statistical problems. Some advantages of NSLKDD over KDDCUP99 dataset include removal of redundant records from training dataset for reducing complexity and bias towards frequent records and the introduction of non-duplicate records in testing datasets for unbiased evaluation.

4 Related work

Several supervised, semi-supervised and unsupervised models from machine learning have been applied to intrusion detection in networks since at least the 1980s [Denning, 1987]. Extensive coverage of both the models, and comparative analysis exists in literature [Laskov et al. 2002, Ghorbani et al. 2010, Bhattacharyya et al. 2013]. Recently Deep Learning models from machine learning have also been applied to the problem of intrusion detection in computer networks.Several machine learning models have been used for anomaly detection in computer networks, with varying degrees of success (medium to high). Supervised approaches such as k-nearest neighbors (k-NN), neural networks and support vector machine (SVM) have been studied extensively for this purpose [Liao et al, 2002, Mukakamala et al. 2002]. A comparative analysis of supervised and unsupervised learning techniques demonstrated that supervised models outperformed unsupervised models in both detection accuracy, as well as correctly identifying yet-unknown malicious activity [Laskov et al. 2005,

Ghorbani et al. 2010]. Recently, using denoising as well as vanilla deep Autoencoders on the NSLKDD dataset, yielded an accuracy of 88.3% [Aygun et al, 2017], which is quite good. Furthermore, an accuracy of 83.3% was claimed by authors when using Autoencoders as unsupervised latent feature generation mechanism on the NSLKDD datasets [Yousefzai-Azar et al, 2017]. Even more remarkable is the performance of Deep Belief Networks when using staked Restricted-Boltzmann Machines, which took the accuracy to a very high 97.5% on the NSLKDD dataset, however this is training and not testing accuracy [Alom et al. 2015]. A further improvement on the training accuracy to up to 98.8% for a 23-way classification is reported by authors when combining encoder layers of Sparse Autoencoder for encoding hidden features, with Softmax regression [Javaid et al. 2016].

An LSTM based anomaly detection system [Bontemps et al. 2016] was proposed. The authors reported variations of incorrect and false alarms of prediction errors via modifying a parameter, β of the proposed system but did not quantify results in terms of F1 score, precision and recall, accuracy of classification, mean average precision, so it is difficult to compare their results with the existing literature.

Application of machine learning to detect intrusion in networks has been shown to be very successful. Given the numerous models which can be applied to this problem, a comprehensive comparative study which documents the results of the most widely used (if not all of the) models in terms of the standard metrics is needed.

5 Results

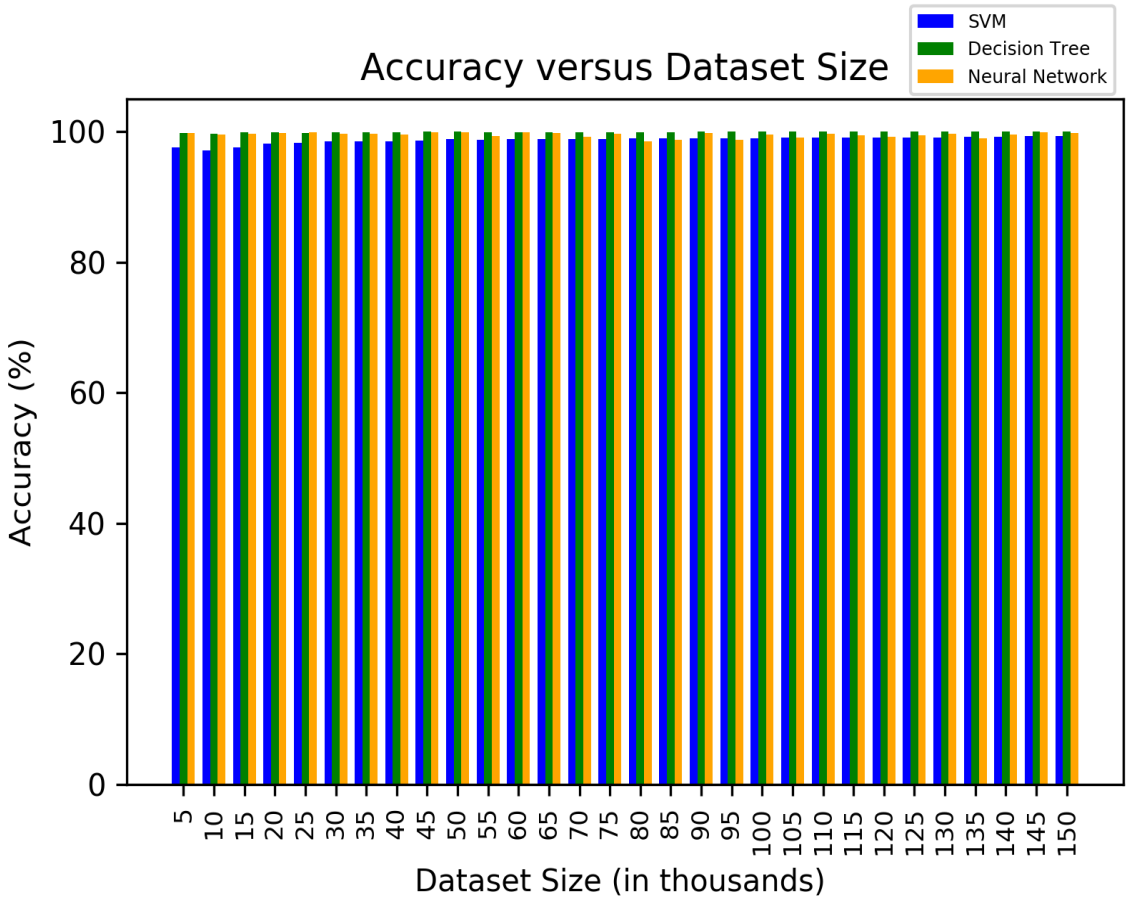

Figure 1: **Shows the accuracy of the three models for each of the data set size. All three models turned out to be more than 97% accurate in their predictions.**

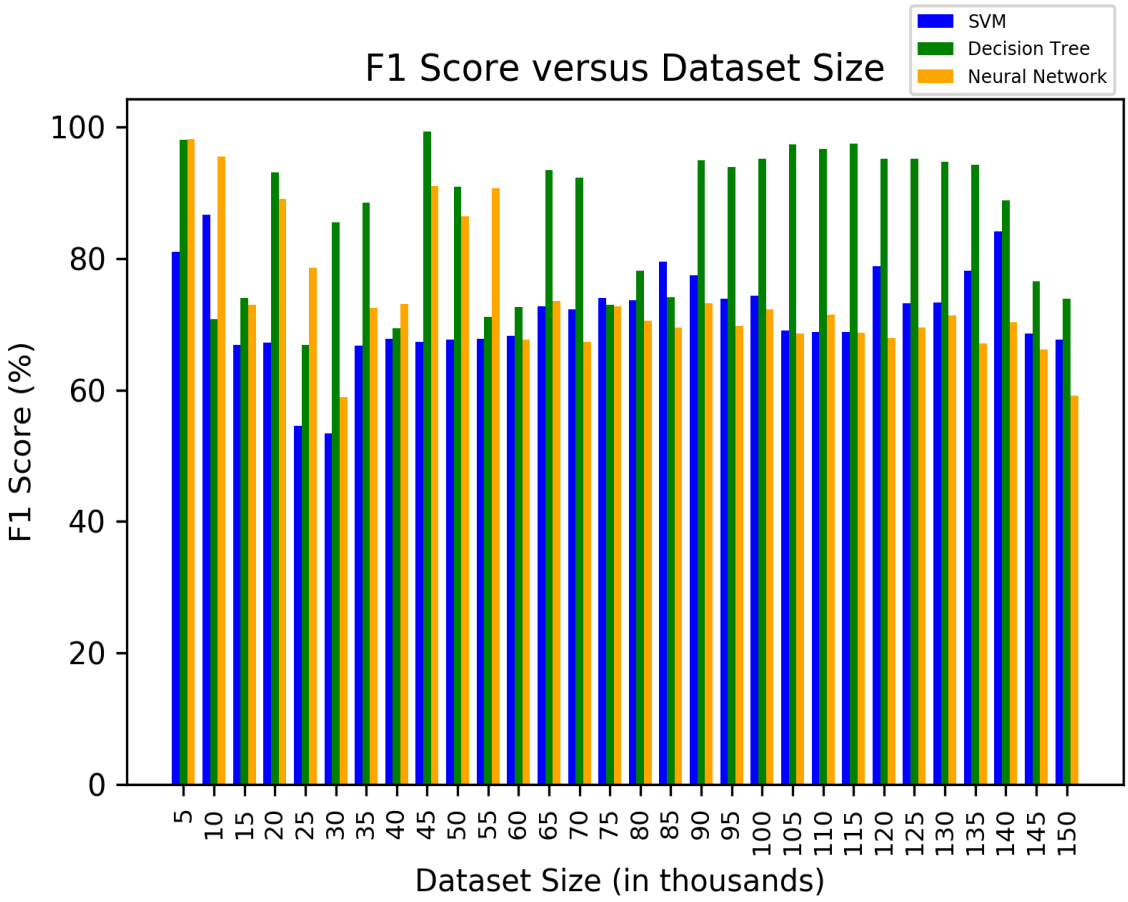

Figure 2: **Depicts the F-1 scores of the three models for each of the data set size. DT performs the best for virtually every data set. SVM and ANN at times fell below 60% was at least 10 percentage points better!**

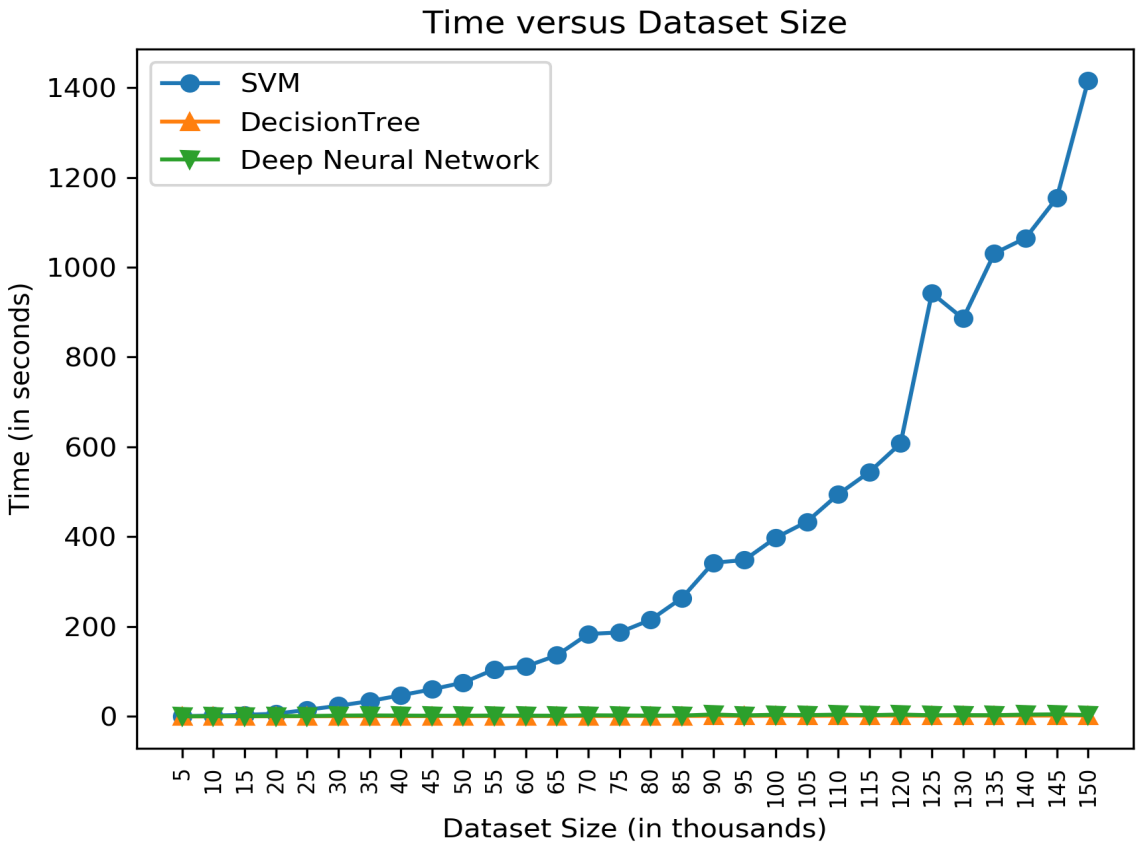

Figure 3: **Portrays the total running time for training and prediction for each model. The running time for SVM is quadratic with the data set size, which is expected as SVM solved a quadratic optimization problem. The running times for DT and ANN seem to be virtually independent of the data set size.**

Dataset Size	SVM (seconds)	Decision Tree (seconds)	Neural Network (seconds)
5000	0.47	0.05	0.15
10000	1.70	0.11	0.32
15000	3.50	0.17	0.43
20000	5.78	0.22	0.83
25000	14.60	0.28	0.72
30000	23.56	0.33	1.68
35000	33.91	0.39	2.10
40000	46.99	0.45	1.81
45000	60.16	0.51	1.45
50000	75.35	0.57	1.45
55000	104.67	0.64	1.87
60000	111.04	0.70	1.63
65000	135.96	0.76	1.46
70000	183.42	0.82	2.38
75000	186.95	0.89	2.06
80000	215.13	0.94	1.67
85000	263.60	1.00	1.79
90000	341.99	1.05	4.25
95000	348.16	1.11	2.15
100000	397.67	1.16	3.68
105000	433.39	1.23	2.74
110000	494.30	1.35	3.88
115000	544.08	1.74	2.65
120000	607.92	1.42	4.13
125000	942.45	1.49	2.45
130000	885.72	1.56	2.79
135000	1030.99	1.63	2.62
140000	1065.21	1.70	3.95
145000	1154.54	1.76	4.55
150000	1415.42	1.83	2.92

Figure 4: **Displays the running times for each model, which is graphically presented in Figure 3 above for visual comparison**

6 Conclusion

Figure 1 depicts the accuracy of classifying of the three models on the **test set**. We observe that while all three produced more than 95% accurate results, SVM consistently lagged behind the other two models, though not trailing by much. We conclude on the basis of our results here that if getting the highest possible accuracy is desirable, one should use DTs to classify connections as normal or one of the intrusions.

Given the dataset which had uneven distribution of the four *classes* (normal, DOS, U2R, R2L), the F-1 scores is a better measure of performance for the three models. We observe in Figure 2 DT again outperforms the other two models consistently, sometimes by a wide margin. ANN performed the worst, and a comparison between ANN and SVM is difficult to draw given the apparent lack of a pattern in Figure 2 for their respective F-1 scores.

While a high performing model is paramount, there usually is a trade-off between the time it takes for a model to train & predict and its performance. The analysis of the running times for the three models (see Figure 3) shows that SVM took the longest to run by a huge margin. It can be seen that there is a loose positive correlation between the running time of SVM and the data set size, while the other two models do not seem to take longer to run with increasing data set sizes.

Couple the running time with the accuracy and F-1 scores of the three models, we conclude that DT is the best overall model for network intrusion detection among the three. It has both the best performance, as well as the least running time. This comes as a surprise as it is breaking the run-time/performance trade off, but it could be the data set we have. Nonetheless gauging from our results, we are inclined toward strongly recommending DT.

7 Future Work

We have done a comparative analysis of three models from machine learning for the detection of intrusive connections in a network. Our work can be extended in various ways in any future studies. We outline a list of tasks that we would like to pursue:

1. **Larger data set**: We would like to perform this study on data set numbering in tens of millions of network calls, which is the typical data set size for machine learning tasks these days.

2. **Balanced data set**: Our data set had a lot of network calls which were normal or were DOS, but very few other forms of intrusion. To better gauge the performance of all the models, it is highly desirable to have a balanced distribution of each *class* in the data. The KDD data set is freely available but has this limitation which we would like addressed by using other data sets.

3. **Granularity**: All of the intrusion types considered in this study are broad categories. Attacks such as rootkit, buffer-overflow, Perl, and load-module are well known instances of U2R category. DOS and R2L have at least six attacks each. We could do a fine-grained study to classify each attack individually, rather than the coarse-grained one done here.

4. **Models** With Deep Learning so popular currently, it would be a good study to apply a Deep Learning model on this task and compare its performance with the other models. Similarly, we could use a Recurrent Neural Network too, to test agains the Feed-Forward ANN we used in this study. Other models like K-Nearest Neighbors, Random-Forests, Regression, and Graphical Models all too can be used on this problem, which a future study can look at.

8 References

1. Hettich, S. and Bay, S. D. (1999). The UCI KDD Archive [http://kdd.ics.uci.edu]. Irvine, CA: University of California, Department of Information and Computer Science.

2. Ahmed M.A., Mohamed Y.A. (2018). Enhancing Intrusion Detection Using Statistical Functions. 2018 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE).

3. S, Naseer, Y. Saleem , S. Khalid , M.K. Bashir , J. Han, M. M. Iqbal, and K. Han. (2018). "Enhanced Network Anomaly Detection Based on Deep Neural Networks", IEEE Special Section on Cyber-threats and Countermeasures in the Healthcare Sector.

4. Y. Liao and V. Vemuri, "Use of K-nearest neighbor classifier for intrusion detection," Comput. Secur., vol. 21, no. 5, pp. 439–448, Oct. 2002.

5. S. Mukkamala, G. Janoski, and A. Sung, "Intrusion detection using neural networks and support vector machines," in Proc. Int. Joint Conf. Neural Netw. (IJCNN), 2002, pp. 1702–1707.

6. P. Laskov, P. Dssel, C. Schfer, and K. Rieck, "Learning intrusion detection: Supervised or unsupervised?" in Proc. 13th Int. Conf. Image Anal. Process. (ICIAP), Cagliari, Italy, F. Roli and S. Vitulano, Eds. Berlin, Germany: Springer, Sep. 2005, pp. 50–57.

7. A. A. Ghorbani, W. Lu, and M. Tavallaee, Network Intrusion Detection and Prevention (Advances in Information Security), vol. 47. Boston, MA, USA: Springer, 2010.

8. R.C.Aygun and A.G.Yavuz,"Networkanomalydetectionwithstochas- tically improved autoencoder based models," in Proc. IEEE 4th Int. Conf. Cyber Secur. Cloud Comput. (CSCloud), Jun. 2017

9. D. E. Denning, “An intrusion-detection model,” IEEE Trans. Softw. Eng., vol. SE-13, no. 2, pp. 222–232, Feb. 1987. [Online]. Available: http://ieeexplore.ieee
.org/abstract/document/1702202/

10. A. Solanas and A. Martinez-Balleste, Advances in Artificial Intelli- gence for Privacy Protection and Security (Intelligent Information Sys- tems). Hackensack, NJ, USA: World Scientific, 2010. [Online]. Available: http:// site.ebrary.com/id/10421991

11. D. K. Bhattacharyya and J. K. Kalita, Network Anomaly Detection: A Machine Learning Perspective. Boca Raton, FL, USA: CRC Press, 2013.

12. N. Gao, L. Gao, Q. Gao, and H. Wang, “An intrusion detection model based on deep belief networks,” in Proc. 2nd Int. Conf. Adv. Cloud Big Data, Nov. 2014, pp. 247–252. [Online]. Available: http://ieeexplore. ieee.org/document/7176101

13. L. Bontemps, V. L. Cao, J. McDermott, and N.-A. Le-Khac, “Collective anomaly detection based on long short term memory recurrent neural network,” in Proc. Int. Conf. Future Data Secur. Eng. Cham, Switzerland: Springer, 2016, pp. 141–152.

14. Joshi, Renuka. “Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures.” Exsilio Blog, 11 Nov. 2016, blog.exsilio.com/all/ accuracy-precision-recall-f1-score-interpretation-of-performance-measures/.