Robustness of Deep Autoencoder in Intrusion Detection under Adversarial Contamination

Pooria Madani
Department Electrical Engineering and Computer Science York University
Toronto, Ontario
madani@eecs.yorku.ca

Natalija Vlajic
Department Electrical Engineering and Computer Science York University
Toronto, Ontario
vlajic@eecs.yorku.ca

ABSTRACT

The existing state-of-the-art in the field of intrusion detection systems (IDSs) generally involves some use of machine learning algorithms. However, the computer security community is growing increasingly aware that a sophisticated adversary could target the learning module of these IDSs in order to circumvent future detections. Consequently, going forward, robustness of machine-learning based IDSs against adversarial manipulation (i.e., poisoning) will be the key factor for the overall success of these systems in the real world. In our work, we focus on adaptive IDSs that use anomaly-based detection to identify malicious activities in an information system. To be able to evaluate the susceptibility of these IDSs to deliberate adversarial poisoning, we have developed a novel framework for their performance testing under adversarial contamination. We have also studied the viability of using deep autoencoders in the detection of anomalies in adaptive IDSs, as well as their overall robustness against adversarial poisoning. Our experimental results show that our proposed autoencoder-based IDS outperforms a generic PCA-based counterpart by more than 15% in terms of detection accuracy. The obtained results concerning the detection ability of the deep autoencoder IDS under adversarial contamination, compared to that of the PCA-based IDS, are also encouraging, with the deep autoencoder IDS maintaining a more stable detection in parallel to limiting the contamination of its training dataset to just bellow 2%.

ACM Reference Format:
Pooria Madani and Natalija Vlajic. 2018. Robustness of Deep Autoencoder in Intrusion Detection under Adversarial Contamination. In *HoTSoS '18: Hot Topics in the Science of Security: Symposium and Bootcamp, April 10–11, 2018, Raleigh, NC, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3190619.3190637

1 INTRODUCTION

Cyber security can be thought as a game which is played against adversaries, and as for any game, it is important to have a strategy. In order to win the game, however, one should also consider reactions of the opponents to the adopted strategy [1]. In respect to this view, an Intrusion Detection System (IDS) is one kind of a strategy in the detection of malicious activities in information systems. Due to lack of stationarity in the world of cyber security (e.g., frequent changes in normal and malicious behaviors), strategies adopted by IDSs require frequent updates in order to compensate for these continual changes. However, these very circumstances/factors - lack of stationarity and attempts of adaptive IDSs to adjust to occurring changes - provide ample opportunities for the adversaries to poison the modelling process of these systems [2]. In other words, by the virtue of their operation, adaptive IDSs are particularly susceptible to contamination introduced by adversaries into their training datasets. For that very reason, achieving robustness under adversarial contamination is one of the most desired properties of detection models in adaptive IDSs.

Deep learning, owed to the use of cascaded layers of nonlinear processing units, has shown promising results in many hard computational tasks and has revolutionized many research fields such as computer vision. Recently, IDS research community have started to adopt deep learning models in the construction of anomaly-based IDSs, and Hodo *et al.* [3] have compiled a survey on the related researche works in this domain. By leveraging the abundance of training data, deep learning models are capable of carrying the task of representation (i.e., feature) learning in order to model underlying complexities in a given training dataset. Given the abundance of sample data in the field of computer security and the complexities of the latent variables that define normality in this field, it is appropriate to study the capabilities of deep generative models for the construction of anomaly-based IDSs.

The contributions of the work presented in this paper are twofold: (1) We have investigated the viability of deploying deep autoencoders - a deep neural network model - in the construction of an anomaly-based IDS. Moreover, we have compared the performance of such an autoencoder-based IDS to that of and an IDS based on the other well-known subspace analysis method - Principle Component Analysis (PCA) - in terms of their ability to capture the normality of the input data as well as to detect their anomalies. Although autoencoders have been used in the past for the purposes of anomaly detection, in this paper we have explored different techniques (e.g., visualization) in validating the constructed model. (2) We have proposed a novel framework for testing robustness of adaptive anomaly-based IDSs under adversarial contamination. Using the proposed framework, we have compared the robustness of the proposed autoencoder-based IDS to that of tis PCA-based counterpart under adversarial contamination.

HoTSoS '18, April 10–11, 2018, Raleigh, NC, USA

ACM ISBN 978-1-4503-6455-3/18/04…$15.00
https://doi.org/10.1145/3190619.3190637

2 BACKGROUND AND RELATED WORKS

Intrusion detection systems, depending on the point of deployment, are generally classified into two categories: (1) Network Intrusion Detection System (NIDS), and (2) Host-based Intrusion Detection System (HIDS). NIDSs are placed at points within a network to monitor traffics to and from all network devices. Given that they are deployed at the network level, they have a broader view of network activities are able to detect network-wide malicious activities. However, encrypted network packets reduce the ability of NIDSs to detect certain classes of intrusions. In contrast to NIDSs, HIDSs run on individual network devices and are able to monitor the inner states of devices in order to detect malicious activities. There are numerous survey papers [4–7] that review the advantages and disadvantages of the two intrusion detection system architectures.

IDSs are further classified into two to distinct categories based on the employed detection techniques: (1) signature-based IDSs, and (2) anomaly-based IDSs. Signature-based IDSs identify attacks by looking for *known* malicious patterns in incoming data, aslo referred to as signatures. . However, new attacks, for which there are no identified signatures, can evade such detection systems. In contract to signature-based, anomaly-based IDSs are capable of capturing unknown malicious activities by measuring the degree of deviation of incoming data streams from what is considered to be the *normal data profile*. There are many machine learning approaches [8–10] that are developed to detect outliers or anomalies in data. Unfortunately, in intrusion detection systems, due to the inherent absence of stationarity, non-adaptive anomaly-based techniques often result in high rate of *false alarm* (i.e., false positive) during runtime. As a result, such techniques are not well adapted by the industry practitioners [11].

Shyu *et al.* [12] were one of the early adaptors of Principle Component Analysis (PCA) in development of anomaly-based IDSs. They assumed anomalies are qualitatively different from the normal instances and this difference can be detected by measuring the deviation from the established *normal* (i.e., non-malicious) datasets. Their proposed Principle Component Classifier (PCC) consists of two functions of principle component scores: (a) major components $\sum_{i=1}^{q} \frac{y_i^2}{\lambda_i}$, and (b) minor components $\sum_{i=p-r+1}^{p} \frac{y_i^2}{\lambda_i}$, where λ_i is sample variance of feature i in the dataset and the top q principle components form the major components. Due to the simplicity of PCA, many anomaly detection-based IDSs have adopted a variation of PCC in detecting malicious activities. However, PCC in its original form, suffer from lack of robustness against adversarial noise in the training dataset. In other words, the training data must be free of any noise which is an unrealistic assumption for building an *adaptive* IDS that requires frequent semi-supervised retraining. Ringberg *et al.* [13] have studied the lack of robustness in PCA models for detecting anomalous traffic in IP networks. Specifically, they have shown that the false alarm rate is very sensitive to small differences in the number of principle components and a realistic number of anomalies in the training data can easily pollute the normal subspace.

Rubinstein *et al.* [14] demonstrated an attack against a typical adaptive PCA-based IDS by injecting malicious points and perturbing the principle components gradually. Furthermore, they developed ANTIDOTE [15] as a framework for testing and enhancing the robustness of anomaly detection techniques in *flow-based* intrusion detectors. It should be noted that our work differs form [14] in terms of the nature of the attack initiated against the model. In ANTIDOTE, the PCA engine is being tested against flow-based attacks whereas in our proposed framework we study the robustness of anomaly detectors against label contamination.

Xiao *et al.* [16] have studied the robustness of *Support Vector Machine* (SVM) against adversarial label contamination. In their work, they theoretically analyzed and proposed an algorithm that can manipulate a set of labels to maximize the empirical loss of the original classifier on a trained dataset. Although our work is inspired by the work of Xiao *et al.* [16], the difference comes from the fact that our proposed simulation is closer to an actual deployment scenario of an adaptive IDS.

Niyaz *et al.* [17] have used deep *autoencoder* for feature extraction and feature reduction before using a feedforward neural network to perform the classification. Our approach is significantly different since we are utilizing the reconstruction error of the autoencoder as the measure for anomaly detection. Hawkins *et al.* [18] used reconstruction error of the trained autoencoder to detect outliers which provided the basis for our research. Although, Hawkins *et al.* used this method for outlier detection, the sensitivity of their method under adversarial setting (e.g., intrusion detection system) has been studied extensively. Moreover, their technique was used to detect generic outliers outside of the context of building an adaptive IDS.

3 APPROACH

3.1 Deep Autoencoders

An autoencoder is a neural network that is trained to reconstruct the provided input on its output nodes. Generally, the representation layer h, describes a *code* that represents the input distribution. An autoencoder is consisted of two parts: an encoder function $h = f(x)$ that encodes the input on to the representation layer h, and a decoder function $r = g(h)$ that reconstruct the *coded* representation h back into the original input. The main goal of this architecture, as depicted in Figure 1, is to learn an approximation of $g(f(x)) \approx x$. Due to the bottleneck in the representation layer h, the autoencoder is forced to encode only approximately the underlying concepts that resemble the input. As a result, the model often learns useful properties of the input data in order to achieve necessary reconstruction ability.

The original idea of autoencoders, initially introduced by LeCun [19], has been studied for decades. Traditionally, autoencoders were used for dimensionality reduction and/or feature learning. Recently, due to the existent of theoretical relations between autoencoders and latent variable models, autoencoders are considered as one of the compelling subspace analysis techniques.

The learning process in autoencoders is described as minimizing a loss function

$$L(x, g(f(x))), \tag{1}$$

where L is the loss function (e.g., mean squared error) intended to penalize the dissimilarity between $g(f(x))$ and x. When the activation function used in *decoder* layers are linear and L is the mean squared error, the autoencoder learns to span the same subspace as *Principle Component Analysis* (PCA). Autoencoders with nonlinear

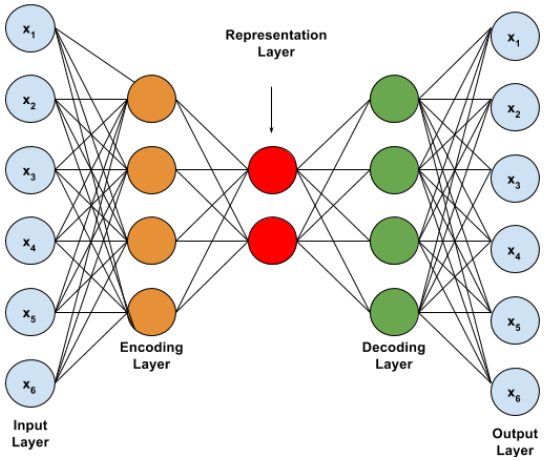

Figure 1: Anatomy of a generic autoencoder.

encoding function f and nonlinear decoding function g can learn a more powerful nonlinear generalization of PCA [20].

In terms of their *neural structure*, autoencoders are a special case of feedforward Artificial Neural Networks (ANN). Thus, they inherit all the properties and training procedures associated with ANNs. More importantly, given nontrivial depth in an autoencoder, universal approximation theorem [21] guarantees that the neural network can approximate any function to an arbitrary degree of accuracy.

3.2 Anomaly Detection using Autoencoders

Although autoencoders are mostly used for feature learning and dimensionality reduction, we are going to use its reconstruction *loss* (i.e., error) as a measure for anomaly detection. The underlying assumption (which is verified by our experiments) is that a deep autoencoder trained on normal (i.e., non-malicious) dataset will have a large reconstruction error when is used to reconstruct non-normal (i.e., attack) data samples. This is due to the fact that latent variables (such as environmental factors, learned by the representation h of the deep autoencoder) that exist in malicious data points are substantially different from those of non-malicious data points. As in Equation 2, *mean squared error* loss function L is used for both training and anomaly detection purposes.

$$L(x, g(f(x))) = \frac{1}{n}\sum(x_i - g(f(x_i)))^2 \tag{2}$$

$$\alpha = P(L(x, g(f(x))) > C \mid \text{x is normal instance}) \tag{3}$$

In order to be able to *approximate* a general function, lossless reconstruction is not the main objective of training of the autoencoder. To compensate for this, an allowance threshold can be introduced to help differentiate between a normal and faulty reconstruction - threshold C in Equation 3. Obviously, the value of this parameter will ultimately (also) control the rate of false-positives (i.e., normal instance mistakenly flagged as malicious) which is vital in the design and deployment of IDSs. To minimize the rate of false positives, the value of threshold C could be calculated empirically before the actual system deployment.

3.3 The Framework for Adversarial Drift Simulation

Although we are not the first to use autoencoders in anomaly detection, due to the recent advances in deep learning, we decided to test the robustness of deep autoencoders for anomaly detection purposes in adversarial settings. In the semi-supervised learning mode of anomaly detection models, normal data instance are provided in order to capture the underlying non-malicious (i.e., normal) data can be extremely large, it is typical to use the existing detection model to select candidate examples from the realtime data streams for retraining. To that end, the most naive strategy to system retraining is to consider new points identified as non-malicious (i.e., normal) to be added to the pool of new training instances in order to cope with the occurring concept-drift. Although many cleansing techniques are developed to remove outliers and abnormalities from such a constructed *training pool*, there are still ample opportunities for contamination and noise to get introduced into the training dataset by the adversaries [15, 16]. Therefore, it is important for detection models, in scenarios where frequent un- supervised retraining is necessary, to be robust to the ill effects of adversarial contamination of the training dataset. To date, there have been several research efforts to test and/or make classification techniques robust to the presence of noise [15, 16]. In this research work, we have decided to specifically test the robustness of an arbitrary deep autoencoder under adversarial contamination and compare it to that of *PCA* under same environmental settings.

Figure 2 depicts the proposed simulation framework within which an anomaly detection-based IDS will frequently retrain itself in order to cope with the occurring concept drifts. Under the proposed framework, training data is split into three smaller subsets – namely, *initial set* T which consist of the initial benign training samples, benign set B which contains the benign instances that are statistically drifted in distribution parameters in respect to T, and malicious set M that contains malicious instances. Initial detection model $D_{i=0}$ construction is done by training on the entirety of the training set T. Then, at each test iteration i, a batch of test points are created by *sampling* (with replacement) n_{benign} instances from set B and *sampling* (without replacement) $n_{malicious}$ instances from set M; the proportion of n_{benign} and $n_{malicious}$ should resemble the reality that the simulation tries to capture. At iteration i, the assembled test batch will be fed into to the detection model D_{i-1}, and those points that are labeled as *normal* are added to the training set T in order to be used during retraining and construction of model D_i. Since the assembled test batch contains both benign and malicious data points, it is possible for malicious points that are not detected (i.e., *false-negative* points) to get added and contaminate set T. At each iteration, before retraining of the IDS, the performance of detection model D_{i-1} is obtained on the assembled test batch in order to construct the performance trend at the end of the simulation. This process continues until $M = \{\emptyset\}$.

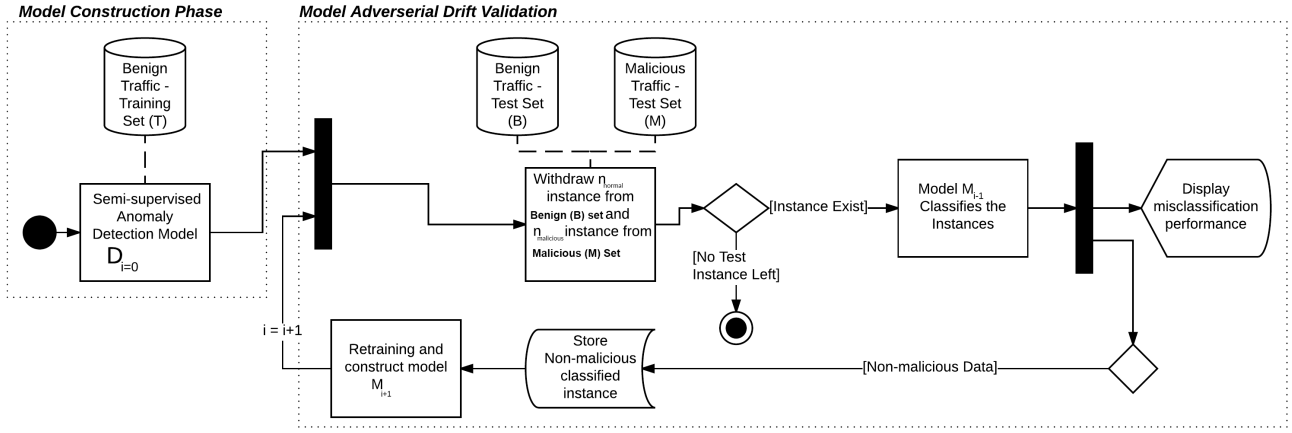

Figure 2: General overview of adversarial drift emulation

$$\text{Cont\%}(i) = \frac{\sum_{j=0}^{i} P\Big(L(x, D_j(x)) \leq C \mid \text{x is malicious}\Big) \times n_{malicious}}{|T|_i} \tag{4}$$

Since each detection model is treated as a black-box in our proposed evaluation framework, the main objective is to measure the trend of their performance by calculating *cumulative* contamination percentage. Equation 4 is the formula for calculating cumulative contamination at iteration i.

Although this method of contaminating training dataset is very simple, it is very realistic when considering deployment of an adaptive anomaly detection-based IDS in practice. After the initial training phase, the decision boundaries of the latest detection model will be used to select from the new incoming data points to enrich the existing training dataset T. Any false-negative classification of new data points can result in contamination of the subsequent retraining dataset, while any false-positive classification can result in loss of a valuable non-malicious training instance.

4 EXPERIMENTS

In this study, we have implemented a deep autoencoder with the proposed anomaly detection scheme in order to assess its robustness against adversarial label contamination. As a baseline for comparison, we have used a widely adapted anomaly detection scheme based on Principle Component Analysis proposed by Shyu *et al.* [12].

Initially, both models were trained with no adversarial drift emulation in order to test their comparative performance under stationarity assumption. Following that, we assessed the performance of the two models under a simulated adversarial drift

4.1 NSL-KDD Dataset

Initiated by DARPA in 1998, *KDDCUP'99* [22] became the mainstream dataset to test and evaluate research works in the field of intrusion detection by containing simulated normal and malicious traffics on a typical U.S. Air Force LAN. McHugh [23] and many other members of the community, however, heavily criticized the process of the creation of the dataset. Tavallaee *et al.* [24], attempted to resolve some of the statistical issues (e.g., data redundancies) that existed in the original dataset and NSL-KDD is the result of their effort. Although the dataset does not reflect some of the most sophisticated attacks that can be found in today's cyber landscape, it is still used to evaluate anomaly detection methods by the research community. For the in-depth explanation of the content of NSL-KDD, we refer readers to the paper published by Tavallaee *et al.* [24].

4.1.1 Preprocessing: NSL-KDD contains two subsets of data for training and testing purposes that are distributionally different. Since our attempt is to distinguish between malicious and normal traffics, we have replaced the label of all non-normal traffic instances with *'malicious'*. Moreover, similar to PCC, our proposed model of the autoencoder IDS uses only *'normal'* instances in the dataset for training. Thus, the normal instances from the training set are being extracted and used as the initial training set T. All the *'malicious'* instances from the training dataset are being merged with *'malicious'* instances from the test dataset in order to form the malicious set M for robustness test and poisoning simulations. *'Normal'* instances left in the test set constitute the instances in B that are continually sampled in order to build a simulation of normal traffic batches in each iteration.

NSL-KDD contains both nominal and numerical features, and for implementation convenience, many researchers exclude nominal-valued features. Moreover, some research works that are focused on model development for classification of malicious instances, tend to exclude some of the features based on their statistical relevance to their classification task. In our work, however, we could not make any argument about unseen normalities, and since we acknowledge that in the real-world ?normal profiles? are generally non-stationary, we have decided to retain all the features and rely on dynamic *representation learning* capability of the autoencoder

to discover and update the appropriate representations in different layers. As a result, we have transformed all the nominal features using *one-hot encoding* in order for these features to be used as inputs to the anomaly detection schemes.

4.2 Implementation

For the purposes of baseline comparison, we have implemented Principle Component Classifier (PCC) proposed by Shy *et al.* [12]. In their proposed anomaly detections scheme, top K principle components are considered to be the *major components* that can capture anomalies with extreme values. Additionally, the remaining principle components are used to detect anomalies that do not have same correlation structure similar to the majority of the normal instances. Choosing K is done experimentally, and since they have used the statistically-flawed KDDCUP'99 [22] dataset, we have redone their experiment on NSL-KDD in order to find the appropriate value for K.

As previously described, the desired *false-positive* rate dictates the selected value of the classification threshold C. The false-positive rate of 2%, frequently used as the target rate in the literature of IDS design, is selected to define the threshold values for both PCC and our proposed autoencoder-based IDS.

The architecture of the implemented autoencoder is similar to the one shown in Figure 1. The input and output layers each contain 122 sigmoid nodes, encoding and decoding layers each contain 50 sigmoid nodes, and the representation layer contains 10 sigmoid nodes.

$$S(t) = \frac{1}{1 + e^{-t}} \tag{5}$$

Sigmoid (Equation 5) activation function, in our experiments, outperformed other activation functions such as *ReLU* or *tanh*. Moreover, the lack of smoothness in the *error* surface (refer to Equation 2), had trapped many deterministic optimization methods such as *normal gradient descent* in different local minima's which resulted in a poor performance. In contrast, Adam Optimizer by Kingma and Ba [25] was used to stochastically compute the optimal *gradient* during the training which resulted in a acceptable detection performance.

Algorithm used to iteratively simulate the adversarial drift for each of the trained models (Algorithm 1 in the subsequent page) is the pseudocode realization of the process depicted in Figure 2. Note that in our experiments, at each iteration i, T_{test} contained $n_{benign} = 500$ benign samples and $n_{malicious} = 50$. Al- though in some attacks (e.g., distributed denial of service (DDoS) attacks) number of malicious samples could be exponentially larger than normal instances, we believe that 10% malicious instances of the total sample size (as used in our experimentation) is a realistic configuration for most other types of attacks including low-rate DDoS. Furthermore, this ratio is treated as a parameter in our proposed framework that can be tuned to emulate any attack scenario.

4.3 Result

The performance of an arbitrary deep autoencoder is measured by its ability to reconstruct presented inputs with a minimum loss. Figure 3 depicts a visualization of NSL-KDD datasets (malicious and non-malicious sets) and their corresponding reconstructions

Algorithm 1: Adversarial Label Contamination Robustness Test

Data: T : Initial benign traffic training sample
B : Benign traffic sample to be used for testing and re-training
M : Malicious traffic sample to be used for testing and poisoning
Result: $R = \{(I, C\%, D\%)\}$ is a set of tuples with I representing the iteration number, $C\%$ representing the contamination percentage, and $D\%$ representing the detection rate of the model interval order.

begin
- $M \longleftarrow$ Construct anomaly detection model using T
- $I \longleftarrow$ Initial iteration count to 0
- $C \longleftarrow$ Initial contamination count to 0
- **while** $M \neq \emptyset$ **do**
  - $I \longleftarrow I + 1$
  - $T_{test}[] \longleftarrow \emptyset$
  - **Add** n_{benign} random samples from B (by selection) to $T_{test}[]$
  - **Add** $n_{malicious}$ random samples from M (by removal) to $T_{test}[]$
  - $D \longleftarrow$ Initialize detection count to 0
  - **for** $x \in T_{test}$ **do**
    - **if** $M.classify(x)$ *as MALICIOUS* **and** x *is MALICIOUS* **then**
      - $D \longleftarrow D + 1$
    - **else if** $M.classify(x)$ *as BENIGN* **and** x *is MALICIOUS* **then**
      - False-negative has occurred, poisoning will take place
      - **Add** x to T
      - $C \longleftarrow C + 1$
    - **else if** $M.classify(x)$ *as MALICIOUS* **and** x *is BENIGN* **then**
      - Update false-positive statistics
    - **else**
      - Add true BENIGN classified traffic to training data to emulate *online* adaptation
      - **Add** x to T
    - **Add** $(I, (C/|T|)\%, (D/n_{malicious})\%)$ to R
  - $M \longleftarrow$ re-train anomaly detection mode using updated T

using the trained deep autoencoder. Figure 3(a) shows the heat-map visualization of the *normal training* instances used during the training of the deep autoencoder. It is important to note that all the feature values are normalized between 0 and 1. Each horizontal pixel-line represents an instance data and the main goal of this visualization is to represent an overall view of value distributions in the dataset.

Figure 3(b) is the corresponding visualization for *malicious* instances that are used during the robustness test. Through a visual inspection of these two heat-maps, we can observe a clear distributional difference between the normal and malicious instance datasets.

Once the autoencoder is trained using normal training instances (shown in figure 3(a)), its reconstruction on malicious and normal training datasets is tested. . Our proposed use of autoencoders? reconstruction error as an anomaly detection measure is validated by means of the obtained results shown in Figure 4: the autoencoder trained on normal instance dataset can poorly reconstruct malicious instances.

As shown in Figure 3(c), normal training instances are almost perfectly reconstructed with the reconstruction error (i.e., square mean error) distribution shown in Figure 4(a). On the other hand, malicious instances shown in Figure 3(b), are poorly reconstructed (refer to Figure 3(d)) and their reconstruction error distribution is shown in Figure 4(b). In order to compute the threshold α in Equation 3, the error distribution in Figure 4(a) is consulted.

The *receiver operating characteristic* (ROC) curve depicted in Figure 5 shows superiority of the autoencoder in detecting network anomalies compared to PCC. It is important to note that the curve is representing a non-poisoning scenario. In other words, all the training instances used to train the PCC and the autoencoder for construction of the ROC curve were empty of any malicious data point. The proposed autoencoder-based IDS has outperformed PCC

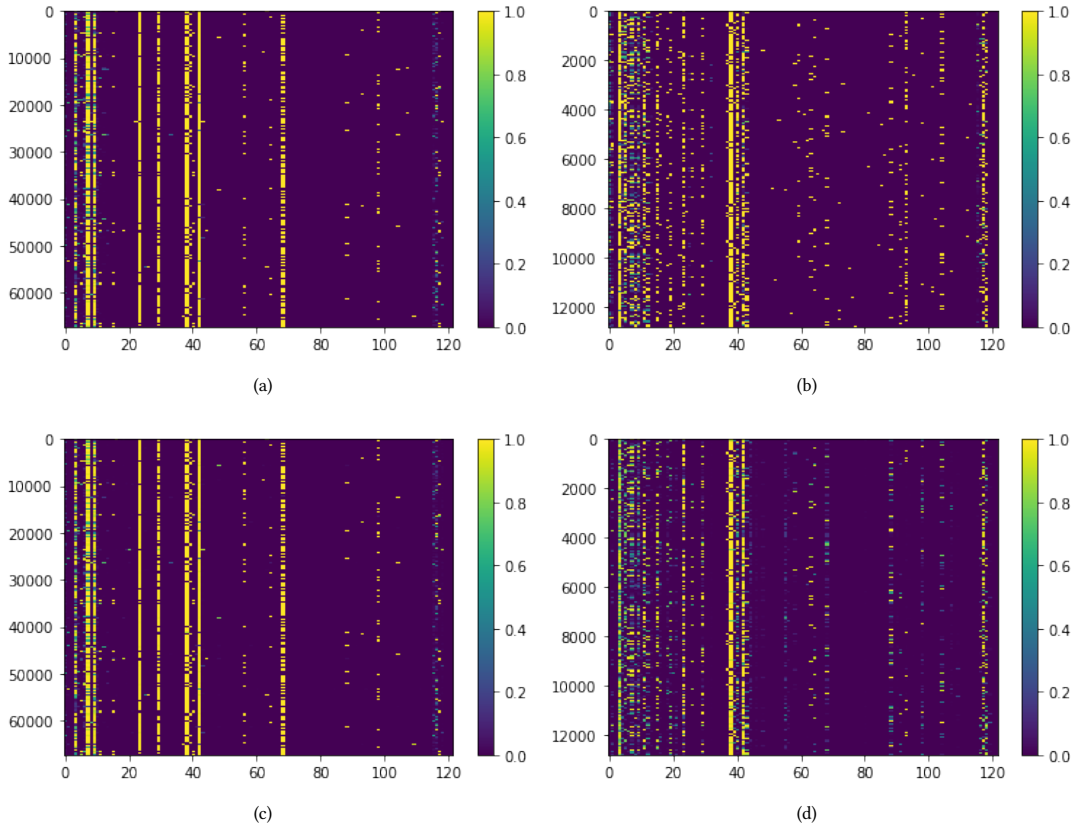

Figure 3: Dataset reconstruction visualization, x-axis represent the feature index while y-axis represent instance index - (a) normal training dataset, (b) anomaly test dataset, (c) reconstruction of normal training dataset using the trained autoencoder, and (d) reconstruction of malicious test dataset using the trained autoencoder.

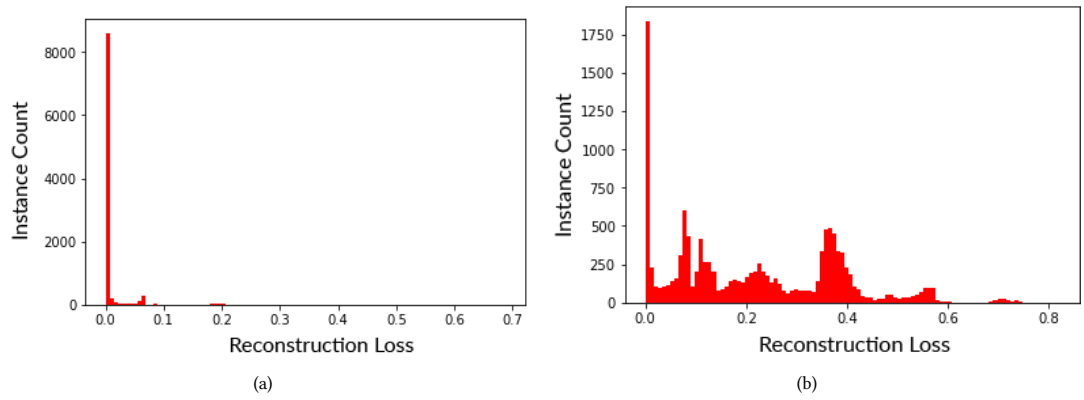

Figure 4: Autoencoder reconstruction error distribution of normal (a) and malicious (b) datasets.

by approximately 15% in detection rate while keeping the rate of reported false alarm at 1%.

The testing of both anomaly detection techniques for robustness under adversarial contamination was performed using the procedure described in Algorithm 1. Figure 6 shows the performance

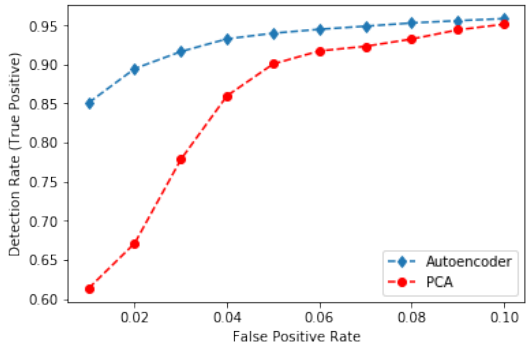

Figure 5: ROC comparison of PCA versus Autoencoder

of each of the methods at different contamination percentage. It should be noted and stressed that contamination selection is not influenced by the experimenters. Instead, recall, malicious samples that are added into the training datasets (at each iteration) are based on the falsenegative classifications of the model currently under test.

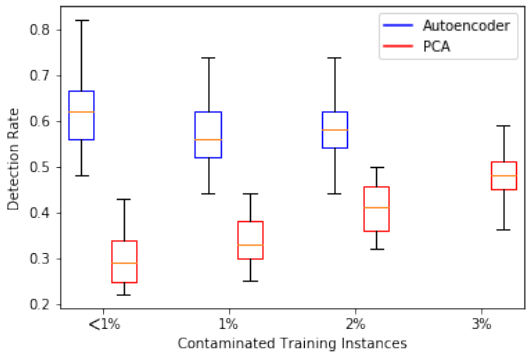

Figure 6: Detection rate of the models under different adapted contamination percentage of the training data.

As shown in Figure 6, the autoencoder IDS is generally more robust to contamination (compared to PCC), as the percentage of contaminated instances that make it into the training set never exceeds 2%. Moreover, under the comparable-conditions of input data contamination (i.e., under the same/comparable rates of false negatives), the autoencoder model?s detection rates are persistently better than those of of PCC. Thus, resulted in less possible contaminating iterations.

5 DISCUSSION AND CONCLUSION

In most anomaly-based IDSs novel attacks are detected at the expense of high rate of false alarm. Moreover, since most of detection alarms typically need to be investigated by human operators, high rate of false-positives (i.e., false alarm) can quickly overwhelm the operators and render the given detection scheme cumbersome. ., Luckily, adaptiveness to occurring concept-drifts has the potential to lower the number of false alarms in most detection systems. For that reason, self adaptiveness is one of the main requirements when develop- ing a modern IDSs. In our research, we have investigated a naive self-adaptive IDS scheme and demonstrated that, if this scheme is to be used in reality, the presence of contaminating points in retraining datasets would be inevitable. In the context of this proposed novel framework , we have furthered compared the robustness of the autoencoder-based IDS against the IDS based on PCA which is well adapted by the industry. Through our comparative study, the deep autoencoder IDS maintained a more stable rate of detection even in the presence of contaminations in its training dataset.

In this research work we have validated the reconstruction ability of deep autoencoders for data points drawn from different distributions. In cybersecurity where *normality* is a moving target, it is shown deep autoencoders can capture latent semantics to reconstruct such data points. Most anomaly-based IDSs detect novel attacks at the expense of high rate of false alarm. Since most of the detection alarm will be investigated by human operators, high rate of false-positive (i.e., false alarm) can quickly overwhelm the operators and render the detection scheme useless. Thus, adaptiveness to the occurring concept-drift can potentially lower such false alarms.

Self adaptiveness is one of the main requirements of developing a modern IDSs. In our research, we have investigated a naive self-adaptive scheme and demonstrated that in reality presence of contaminating points in retraining datasets are inevitable. Thus, through our proposed novel approach, we have compared the robustness of the autoencoder-based IDS with one of the popular PCA-based IDS that is well adapted by the industry. Through our comparative study, the deep autoencoder IDS maintained a more stable rate of detection even in the presence of contaminations in its training dataset (refer to Figure 6).

In the case of PCA-based adaptive IDSs, complete retraining of the entire model is necessary to cope with occurring concept drift. Given the training dataset is continually growing (while new training data are added to the existing training dataset), we have observed an increase in model re-training time unless a *forgetting* mechanism is introduced. In case of deep autoencoders, however, adaptation of existing detection model to occurring concept drift does not require retraining from scratch. Using mini-batch stochastic gradient descent we can update the latest detection model only using the recent acquired data examples. Thus, in a long run, deep autoencoders can exhibit superior online adaptive performance in respect to the re-training time required.

REFERENCES

[1] Kantchelian, A., Afroz, S., Huang, L., Islam, A.C., Miller, B., Tschantz, M.C., Greenstadt, R., Joseph, A.D., Tygar, J.: Approaches to adversarial drift. In: Proceedings of the 2013 ACM workshop on Artificial intelligence and security, ACM (2013) 99–110

[2] Huang, L., Joseph, A.D., Nelson, B., Rubinstein, B.I., Tygar, J.: Adversarial machine learning. In: Proceedings of the 4th ACM workshop on Security and artificial intelligence, ACM (2011) 43–58

[3] Hodo, E., Bellekens, X., Hamilton, A., Tachtatzis, C., Atkinson, R.: Shallow and deep networks intrusion detection system: A taxonomy and survey. arXiv preprint arXiv:1701.02145 (2017)

[4] Ghorbani, A.A., Lu, W., Tavallaee, M.: Network intrusion detection and prevention: concepts and techniques. Volume 47. Springer Science & Business Media (2009)

[5] Liao, H.J., Lin, C.H.R., Lin, Y.C., Tung, K.Y.: Intrusion detection system: A comprehensive review. Journal of Network and Computer Applications **36**(1) (2013) 16–24

[6] Kim, G., Yi, H., Lee, J., Paek, Y., Yoon, S.: Lstm-based system-call language modeling and robust ensemble method for designing host-based intrusion detection systems. arXiv preprint arXiv:1611.01726 (2016)

[7] Vasilomanolakis, E., Karuppayah, S., Mühlhäuser, M., Fischer, M.: Taxonomy and survey of collaborative intrusion detection. ACM Computing Surveys (CSUR) **47**(4) (2015) 55

[8] Garcia-Teodoro, P., Diaz-Verdejo, J., Maciá-Fernández, G., Vázquez, E.: Anomaly-based network intrusion detection: Techniques, systems and challenges. computers & security **28**(1) (2009) 18–28

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

[10] Kandhari, R., Chandola, V., Banerjee, A., Kumar, V., Kandhari, R.: Anomaly detection. Comput. Surveys **41**(3) (2009) 1–6

[11] Sommer, R., Paxson, V.: Outside the closed world: On using machine learning for network intrusion detection. In: Security and Privacy (SP), 2010 IEEE Symposium on, IEEE (2010) 305–316

[12] Shyu, M.L., Chen, S.C., Sarinnapakorn, K., Chang, L.: A novel anomaly detection scheme based on principal component classifier. Technical report, DTIC Document (2003)

[13] Ringberg, H., Soule, A., Rexford, J., Diot, C.: Sensitivity of pca for traffic anomaly detection. ACM SIGMETRICS Performance Evaluation Review **35**(1) (2007) 109–120

[14] Rubinstein, B.I., Nelson, B., Huang, L., Joseph, A.D., Lau, S.h., Rao, S., Taft, N., Tygar, J.: Stealthy poisoning attacks on pca-based anomaly detectors. ACM SIGMETRICS Performance Evaluation Review **37**(2) (2009) 73–74

[15] Rubinstein, B.I., Nelson, B., Huang, L., Joseph, A.D., Lau, S.h., Rao, S., Taft, N., Tygar, J.: Antidote: understanding and defending against poisoning of anomaly detectors. In: Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference, ACM (2009) 1–14

[16] Xiao, H., Biggio, B., Nelson, B., Xiao, H., Eckert, C., Roli, F.: Support vector machines under adversarial label contamination. Neurocomputing **160** (2015) 53–62

[17] Javaid, A., Niyaz, Q., Sun, W., Alam, M.: A deep learning approach for network intrusion detection system. In: Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS), New York, NY, USA. Volume 35. (2015) 2126

[18] Hawkins, S., He, H., Williams, G., Baxter, R.: Outlier detection using replicator neural networks. In: International Conference on Data Warehousing and Knowledge Discovery, Springer (2002) 170–180

[19] Soulié, F.F., Robert, Y., Tchuente, M.: Automata networks in computer science: Theory and applications. Manchester University Press (1987)

[20] Goodfellow, I., Bengio, Y., Courville, A.: Deep learning. MIT Press (2016)

[21] Csáji, B.C.: Approximation with artificial neural networks. Faculty of Sciences, Etvs Lornd University, Hungary **24** (2001) 48

[22] Cup, K.: Dataset. available at the following website http://kdd. ics. uci. edu/databases/kddcup99/kddcup99. html **72** (1999)

[23] McHugh, J.: 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 (TISSEC) **3**(4) (2000) 262–294

[24] Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A.: 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 (2009) 1–6

[25] Kingma, D., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)