Network Intrusion Detection Using Various Deep Learning Approaches

Kamila Bekshentayeva, Matthew Canute, Young-min Kim, Donggu Lee, Adriena Wong
Department of Computing Science and Engineering Science
Simon Fraser University
Burnaby, BC V5A 1S6
{kdagilov, adriena_wong, mcanute, donggul, yka85}@sfu.ca

Abstract

This paper analyzes the potential degradation of machine learning models given the changing network security landscape, defined by a constant battle between adversaries and network intrusion detection systems, which are adapting to evolving cyber-threats. Various deep learning models, such as Feed Forward Neural Networks (FFNN), Long Short-Term Memory Networks (LSTM), and Echo State Networks (ESN), are tested on CSE-CICIDS2018 data set, where the most important features were extracted to enhance detection of denial of service (DoS) attacks. The performance of the models was evaluated based on accuracy and F1 score.

Additionally, a new data set was generated by employing a modified attack, previously unseen by our models' training and test sets, on our own web-server. While one of the models was unable to detect this new attack, the other two models, which relied on temporal information, were still able to identify the modified attack with comparable results.

1 Introduction

Network Intrusion Detection systems (NIDS), which are primary defence elements in the security of network technology, monitor network traffic in order to detect malicious activities. These systems rely on the identification of known attack signatures; however, the frequency of complex, diverse, and challenging-to-detect attacks increases the difficulty to identify current threats. Finding deviations from normal behaviour in the observed network traffic is known as anomaly-based intrusion detection.

To detect anomalies, many NIDS implement classification or clustering algorithms. Various machine learning models were proposed to identify events that do not conform to regular patterns. In order to learn the representation from multi-variate time-series data, an Echo State Network (ESN) was combined with a variational autoencoder (VAE) in (15). This model, ESN-CVAE, demonstrated competitive performance in anomaly detection. In (9), Long Short-Term Memory (LSTM), bi-directional LSTM, Gated Recurrent Unit (GRU), Broad Learning System (BLS), and Deep Learning Neural Network were applied to classify known network intrusions using the NSL-KDD data set. The results indicated that BLS performs comparably in terms of accuracy and F-Score, with shorter training time. In (8), the cascade-structured meta-specialists approach for neural network-based anomaly detection achieves a very high accuracy with a low false-positive rate, when tested on KDD Cup 99 and NSL-KDD. However, some machine learning methods might misclassify cyber-threats, for example when non-random perturbations are applied to a data set, which can drastically decrease accuracy, and illustrate potential vulnerabilities of NIDS (16).

In this paper, we compared the performance, based on the accuracy and F1 score, of three different machine learning models in their ability to detect network attacks: Feed Forward Neural Networks (FFNN), Long Short-Term Memory Networks (LSTM), and Echo State Networks (ESN). We used a

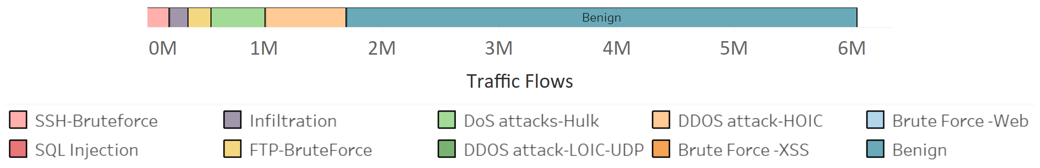

Figure 1: Data Set (76 Traffic Features of 6.1 Million Flows in total)

simple two-layer FFNN, which did not directly take into account the temporal aspect of the data, and compared it to an LSTM and ESN, which are different types of recurrent neural networks (RNNs) that process sequential data. These models were chosen because of their success with the similar KDD data set. The three models were trained and tested on CSE-CIC-IDS2018 data set. Additionally, we tested the models' performance on a modified attack to evaluate their adaptability.

The paper is organized as follows: In this section, we introduced the topic and described the related work. The description of data set and features is provided in section 2. The summary of denial of service (DoS) and distributed denial of service (DDoS) attacks is given in 3. The FFNN, LSTM, and ESN algorithms are presented in 4. The experimental procedure and performance evaluation are given in sections 5 and 6, respectively. We conclude with section 7.

2 Data Set Description and Feature Selection

With changing network behaviours and intrusion patterns, it is becoming significantly more challenging to find a data set that reflects the current network trends. Moreover, most data sets are private, anonymized, and often missing certain statistical characteristics.

The Communications Security Establishment (CSE), in collaboration with the Canadian Institute for Cybersecurity (CIC), has devised a testbed framework (11; 12) to dynamically generate traffic flow data with the representative traffic composition of today. The testbed of this data set - CSE-CIC-IDS2018 (1) consists of victim and attacker networks implemented within Amazon Web Services: an attacker-network with 50 terminals, and a victim-network, implemented as a Local Area Network (LAN) with 420 terminals and 30 servers split into 5 subsets.

The benign (regular) data are collected as background traffic based on user behavior when employing the following protocols: HTTP, HTTPS, SMTP, POP3, IMAP, SSH, and FTP, in a non-malicious manner. Malicious (attack) data contain the most common attacks to date: brute force XSS and Web attacks, DoS and DDoS attacks, exploiting the heartbleed vulnerability, and many others. In this paper, we considered the GoldenEye, Slowloris, SlowHTTPTest, and Hulk DoS attacks collected on Thursday 15.02.2018 and Friday 16.02.2018. More details of the attack types in each day's data set can be found in (1).

The CSE-CIC-IDS2018 data set captures ten days between Wednesday 14.02.2018 and Friday 02.03.2018 (1). There are 6.1 million rows, each consisting of a bidirectional flow (biflow) representing the source-to-destination packet for nine different attack types as shown in fig. 1, and 76 features were extracted using the CIC open source tool CICFLOWMETER, mostly consisting of continuous values, such as the flow duration, but also including a one-hot encoding for the type of protocol, and an aggregate metric for the number of other bidirectional traffic flows associated with the same timestamp. Other features include maximum/minimum packet size and flow packets rate as shown in their website (1).

3 DoS and DDoS Attacks

DoS and DDoS attacks are employed by cyber criminals and hacktivists to overload a network's infrastructure causing disruptions and outages to small, medium, and large companies. Looking at the time series in fig. 2, DoS detection appears to be trivial: one could simply identify when there is a sudden upsurge in packet requests, and thus successfully detect whether the network is under attack using only one variable. However, the problem is not to determine whether the whole network is under attack, but rather to accurately classify whether a request for each source IP and port traffic flow is malignant or benign, since we do not want to block benign traffic in the midst of a DoS attack.

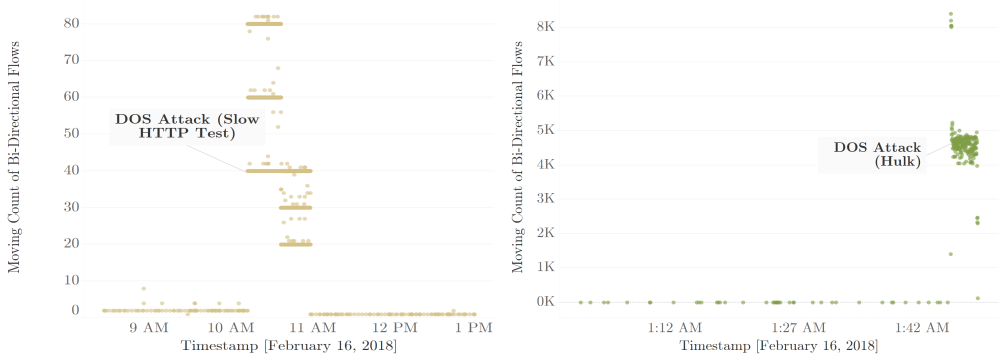

Figure 2: Attacks visualization

DDoS attacks may belong to one of the three categories: volumetric DDoS, TCP state exhaustion attacks, and application-layer attacks. Volumetric attacks or "floods" crash a target by a voluminous traffic. A botnet, a collection of devices infected with malware under a control of a botmaster, floods a victim by consuming its bandwidth with UDP or ICMP packets until victim fails. TCP state exhaustion attacks, also known as protocol attacks, usually target firewalls, load balancers, and servers by sending large IP packets or SYN, thus flooding a target system. The main focus of application-layer attacks is to monopolize SMTP, HTTP, and DNS services. These types of attacks are the most challenging to detect because the requests seem to be legitimate.

The following DoS and DDoS attacks are contained within the portions of data sets used to train the proposed models: Slowloris, GoldenEye, HULK, and SlowHTTPTest. *Slowloris* attack is described as slow rate and low volume traffic generation incoming from a single source, which makes it problematic to detect using traditional NIDS approaches. This type of attack occurs at the application-layer, and is characterized by sending fractional HTTP GET requests without termination code in order to keep connection open. Hence, the server is forced to wait indefinitely for the remaining data. *GoldenEye* is a Python-based application for security testing that may be also used for malicious activities. Attacks created using GoldenEye also belong to the class of application-layer attacks. Their main goal is to consume all of the available HTTP/HTTPS sockets by sending keep-alive requests in addition to cache control options. *HULK (HTTP Unbearable Load King)* is a tool for creating high volumes of obscure traffic that brings down web servers with voluminous requests from a number of random user agents. *SlowHTTPTest* is another tool for implementing most common application-layer DoS attacks, such as Slow HTTP Post, Slowloris, and Slow Read attacks, which rely on the fact that HTTP is designed to wait for complete requests before they are processed, which in turn drains the concurrent connections pool.

4 Algorithms

In order to compare the effectiveness of different approaches in detecting the above cyber-attacks, we evaluated the performance of three machine learning model architectures: Feed-Forward Neural Networks (FFNN), Long Short-Term Memory (LSTM), and Echo State Networks (ESN). They are described below, with more emphasis placed on the ESN design, as it is the lesser-known approach.

4.1 Feed-Forward Neural Networks

A simple FFNN is an artificial neural network with multiple hidden layers between the input and output layers. For this model, we did not use any information from the progression of the data set as a time-series, but rather trained on each row at a time. By referencing another paper that created a NIDS using the same data set, we started with a similar network architecture, although we were not able to achieve the same 100% accuracy that their paper claimed to achieve (6). As will be shown in a later table, however, we ended up with similar results using a much smaller number of features. We then experimented with two to four hidden layers along with different-sized hidden layers by factors of eight, and drop-out rates by factors of 0.2 in a grid-search of the best F1 score. As a result, two hidden layers were chosen with drop-out rates 0.2 for the first hidden layer and 0.4 for the second.

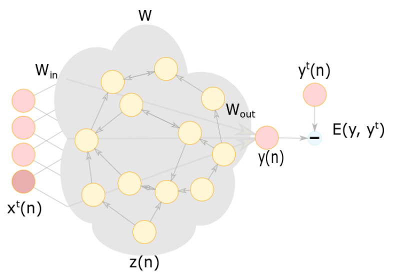

Figure 3: Echo State Network

FFNN was run for 18 epochs since the model's learning rate appears to converge shortly afterwards. The sizes of the layers, determined by trial and error, were 16 for the first layer and 8 for the second.

4.2 Recurrent Neural Networks

RNNs are a class of neural networks that process sequential data, allowing it to process much longer data sequences than other networks. RNNs can use their internal state to process sequences of inputs. This makes them applicable to tasks such as unsegmented, connected handwriting recognition or speech recognition (3). Unlike FFNNs, RNNs have feedback connections, where the output nodes form a cycle and is fed back into itself in future time steps.

4.2.1 Long Short Term Memory

Long short-term memory (LSTM) is an artificial RNN architecture used in the field of deep learning. Unlike standard feedforward neural networks, LSTMs have feedback connections. While the traditional LSTM is unidirectional, a bidirectional LSTM has two layers: one forward and one backward, connected to the output. We used a bidirectional LSTM, which may have limitations in how it is deployed within an intrusion detection system. However, it would be beneficial when used as a retrospective review the next day to determine which traffic biflow IP addresses were associated with a network attack.

Using the TensorFlow Keras package, the following values were chosen via trial and error for training the number of epochs: 20, bidirectional LSTM layer: 8 internal units, first layer activation: ReLU, second layer activation: sigmoid, loss function: binary cross entropy, and optimizer: Adam. An Adam optimizer was chosen due to its suitability for data sets with a large number of datapoints and/or parameters (7).

4.2.2 Echo State Networks

A practical and conceptually simple algorithm to implement, Echo State Network (ESN), belonging to Reservoir Computing approaches, is an alternative to gradient descent used for training RNNs (10). RNNs, due to their computational and memory capabilities, are one of the most powerful approaches in machine learning. However, training RNNs is a challenging task. In ESN, only the output from the reservoir, which is a randomly generated RNN, is trained. ESNs can be employed as a supervised machine learning approach for temporal data. The general structure of ESN is show in fig. 3.

$$E(y, y^{target}) = \frac{1}{N_y} \sum_{n=1}^{N_y} \sqrt{\frac{1}{T} \sum_{i=1}^{T} (y_i(n) - y_i^{target}(n))^2} \qquad (1)$$

According to the original reservoir computing approach introduced in (5), there are four steps associated with applying ESN:

- generation of random reservoir with the following parameters: W^{in}, W, α;
- obtaining the reservoir activation states $z(n)$ by using the training input $x(n)$;
- obtaining the output weights, while minimizing the error as in expression (1);

- applying collected output weights W^{out} with new input $x(n)$ to compute $y(n)$.

The reservoir in the ESN, providing a signal space for the input, allows us to obtain the desired y_i^{target}, and has two roles: it serves as a non-linear high dimensional expansion $z(n)$ of the input signal $x(n)$, and as a memory of the input. The reservoir is characterized by the tuple (W^{in}, W, α), where the input W^{in} and the recurrent connection W matrices generation is random. The global parameters of the ESN are: size of the reservoir N_z, reservoir sparsity, distribution of nonzero elements, spectral radius of W; W^{in}, and leaking rate α.

Due to the ESN's large memory requirements for its reservoir, some combinations of parameters produced memory errors and thus were unable to be considered. Following the practical guide to applying the ESN (10), and through conducting grid-search involving changing the spectral radius, leaking rate, and input scaling by 0.2 and the number of internal units by 5, we selected the following values for number of internal units: 20 for Thursday 15.02.2018 and 10 for Friday 16.02.2018, spectral radius: 0.9, leaking rate: 0.2, connectivity: 0.25, input scaling: 0.3, and noise level: 0.01.

5 Experimental Procedure

We performed a binary classification task on subsets of the CSE-CIC-IDS2018 Thursday 15.02.2018 and Friday 16.02.2018 data sets to distinguish between benign (0) and malignant (1) bi-directional flows. After trying different approaches such as z-score normalization, and trying different numbers of features, the resulting experimental procedure was as follows:

- grouping the data by timestamp and flow duration in order to determine the number of bi-flows per timestamp;
- sorting the data based on the timestamp of the flow's first packet;
- creating the training and test data sets by partitioning the data into 80% and 20% respectively;
- extracting the 15 most important features for the particular data set detailed in fig. 4;
- normalizing the data using a min-max scaling;
- replacing all NaN and infinity values with 0;
- training the model with appropriate parameters as described in the previous section;
- evaluating the performance using the accuracy and F1 metrics;
- oversampling 20% more observations of the underrepresented attack.

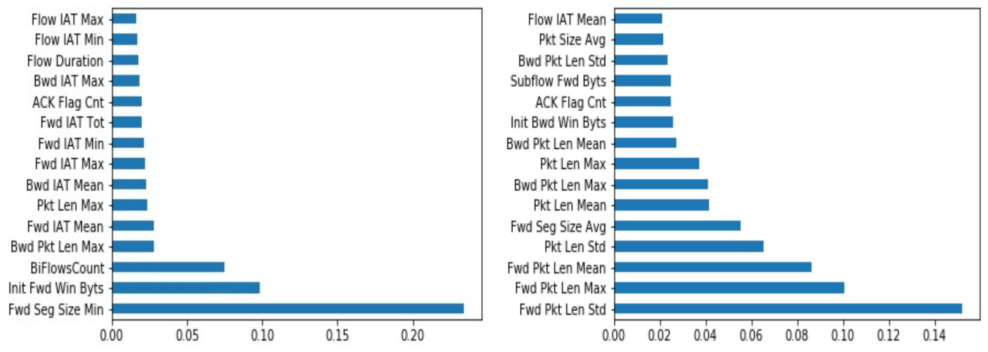

Figure 4: Feature importance for Thursday 15-02-2018 (left) and Friday 16-02-2018 (right)

After fitting the models and noting down the test accuracy, we found an updated 2019 version of the HULK attack script (13). Additionally, we modified it further by determining the feature importance of the FFNN by inspecting the separate Random Forest model's feature importance, considering the FFNN shap values (14), and observing weight coefficients of a separate logistic regression model. We used that information to attempt to trick the model by modifying the updated HULK attack script so that the HTTP request sizes were halved, and we inserted a timer to pause randomly between one to three seconds for the first 30 requests in an attempt to trick the LSTM and ESN as well. A simple webserver was then set up on a home network, and another computer was set up to use this modified DoS attack on the web server, along with other simulated web traffic randomly generated. A wireshark session was running in the background during the 30 seconds before the attack started, and during the 40 seconds that it took to shut down the web server. This generated a pcap file that we were able to convert into the same data set format by using the same CICFlow tool. Then we tested our trained models on this new data set.

6 Results and Discussion

The performance of the three tested models is shown in Table 1. The performance measures used are accuracy in expression (2) and F1 score in expression (3) where true positive (TP) are correctly identified intrusions, false negatives (FN) are correctly identified benign cases, false positives (FP) are incorrectly identified intrusions, and true negatives (TN) are incorrectly identified benign cases.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{2}$$

$$F1Score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{3}$$

where Precision $= \frac{TP}{TP+FP}$ and Recall $= \frac{TP}{TP+FN}$.

While all three models performed relatively well on both the Thursday 15-02-2018 and Friday 16-02-2018 attacks, it appears that the FFNN had the best performance in terms of its accuracy at 99.8% and 99.7%, respectively, and F1 scores at 98.6% and 99.9%, respectively. However, the FFNN failed to detect the 2019 updated HULK attack, while the ESN and the LSTM, maintained high accuracy and F1 scores. This is likely due to the fact that the FFNN does not consider the time series characteristics of the data, and, therefore, was unable to consider that the group count of bidirectional traffic flows, associated with the same timestamp, increased dramatically compared to the last few seconds.

We decided that the F1 score would be the most appropriate metric for model comparisons, since both the precision and the recall seemed to be equally important for detecting a DDoS attack. If our model were to have a high rate of false-positives and thus bad precision, then the DDoS attack would effectively fulfill its goal of shutting down our services because benign HTTP requests would be incorrectly denied. However, having a high rate of false-negatives and thus bad recall, could result in shutting down our service altogether.

Table 1: Model Comparisons

	2018-02-15 GoldenEye/Slowloris		2018-02-16 HULK/SlowHTTPTest		2019 Modified HULK Attack	
Models	ACC	F1	ACC	F1	ACC	F1
LSTM	95.0%	N/A*	99.5%	99.6%	97.1%	98.5%
ESN	98.8%	99.6%	99.6%	99.6%	97.2%	98.6%
FFNN	99.8%	98.6%	99.7%	99.9%	3.2%	0.4%

* The F1 Score was not applicable in the cases where no malignant traffic flows were predicted.

7 Conclusion

In this project three deep learning models: FFNN, LSTM, and ESN were tested using a novel data set, CSE-CICIDS2018, which reflects up-to-date network traffic with regular flows and current network threats. The models demonstrated competitive performance, when evaluated based on the accuracy and F1 score. Although only four DoS attacks out of the wide range of possible network attacks were considered, we believe that this experiment demonstrates, how it may be beneficial to use a collection of different classifier model architectures. When an adversarial attack was created to generate less suspicious-looking data from the perspective of a classifier, the FFNN model failed to identify attacks, while the LSTM and ESN models were still able to detect the majority of the malignant traffic flows with accuracy and F1 scores above 97%.

This project also demonstrates the potential vulnerabilities of anomaly detection systems, i.e. if the representative network traffic data set, used to train a particular machine learning model, were to be released to the public, then the model might become ineffective. An attacker could potentially use that same public data set to build a reasonably similar model, and adjust the attack script until their model doesn't detect the attack anymore, which could then be utilized to bypass the model of interest.

8 Contributions

Our team consists of 5 members: Kamila Bekshentayeva, Matthew Canute, Young-min Kim, Donggu Lee, and Adriena Wong. The following are their contributions to this project: Kamila - General domain expertise, ESN research, poster creation, report writing; Matthew - FFNN exploration and code, adversarial attack implementation, report writing, poster creation; Young-min - LSTM exploration and code, features importance analysis, poster creation; Donggu - data preparation, poster visualisation creation; Adriena - ESN exploration and code, report writing.

References

[1] A Realistic Cyber Defense data set (CSE-CIC-IDS2018) [Online]. Available: https://registry.opendata.aws/cse-cic-ids2018/. Accessed: Oct. 28, 2019.

[2] CICFlowMeter [Online]. Available: http://netflowmeter.ca/ netflowmeter.html. Accessed: Oct. 28, 2019.

[3] I. GoodFellow,"Deep Learning,"in *The MIT Press* November, 2012, pp. 373.

[4] Intrusion Detection Evaluation data set (CICIDS2017) [Online]. Available: https://www.unb.ca/cic/data sets/ids-2017.html. Accessed: Oct. 28, 2019.

[5] H. Jaeger, "The 'echo state' approach to analysing and training recurrent neural networks," Tech. Rep. GMD Rep. 148, German Nat. Res. Center for Inf. Technol., 2001.

[6] V.KanimozhiT. PremJacob, "Artificial Intelligence based Network Intrusion Detection with hyper-parameter optimization tuning on the realistic cyber data set CSE-CIC-IDS2018 using cloud computing," in ICT Express Volume 5, Issue 3, Pages 211-214, September 2019.

[7] D. P. Kingma, J. Ba, "Adam: a method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Representations*, San Diega, USA, 2015, pp 1.

[8] M. Labonne, A. Olivereau, B. Polvé, and D. Zeghlache, "A cascade-structured meta-specialists approach for neural network-based intrusion detection," in *Proc. 2019 16th IEEE Annu. Consumer Commun. Netw. Conf.*, Las Vegas, NV, USA, Jan. 2019, pp. 1–6.

[9] Z. Li, P. Batta, and Lj. Trajkovic, "Comparison of machine learning algorithms for detection of network intrusions," in *Proc. IEEE International Conference on Systems, Man, and Cybernetics (SMC 2018)*, Miyazaki, Japan, Oct. 2018, pp. 4248–4253

[10] M. Lukoševičius,"A Practical Guide to Applying Echo State Networks,"in *Neural Networks: Tricks of the Trade.* Springer, 2012, pp. 659–686.

[11] I. Sharafaldin, A. Gharib, A. H. Lashkari, and A. A. Ghorbani, "Towards a reliable intrusion detection benchmark data set," *Softw. Netw.*, vol. 2017, no. 1, pp. 177–200, July 2017.

[12] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection data set and intrusion traffic characterization," in *Proc. 4th Int. Conf. Inform. Syst. Secur. Privacy*, Funchal, Portugal, Jan. 2018, pp. 108–116.

[13] B. Shteiman, Hyperclaw79, "Modified HULK script for DoS attacks version 2.0," in *GitHub Repository*, 2017, Available: https://github.com/Hyperclaw79/HULK-v2. Accessed: Dec. 4, 2019.

[14] slundberg, "Shap," *GitHub Repository*, 2019, Available: https://github.com/slundberg/shap. Accessed: Dec. 11, 2019

[15] S. Suh, D. H. Chae, H. Kang and S. Choi, "Echo-state conditional variational autoencoder for anomaly detection," 2016 *Int. Joint Conf. on Neural Networks (IJCNN)*, Vancouver, BC, 2016, pp. 1015–1022.

[16] A. Warzyński and G. Kołaczek, "Intrusion detection systems vulnerability on adversarial examples," *2018 Innovations in Intelligent Systems and Applications (INISTA)*, Thessaloniki, 2018, pp. 1–4.