## **Deep Neural Networks for Network Intrusion Detection Systems in Cyber Security**

**Abstract**—Intrusion detection system (IDS) has become an essential layer in all the latest ICT system due to an urge towards cyber safety in the day-to-day world. Reasons including uncertainty in finding the types of attacks and increased the complexity of advanced cyber-attacks, IDS calls for the need of integration of Deep Neural Networks (DNNs). In this paper, DNNs have been utilized to predict the attacks on Network Intrusion Detection System (N-IDS). A DNN with 0.1 rate of learning is applied and is run for 1000 number of epochs and KDDCup-’99’ dataset has been used for training and benchmarking the network. For comparison purposes, the training is done on the same dataset with several other classical machine learning algorithms and DNN of layers ranging from 1 to 5. The results were compared and concluded that a DNN of 3 layers has superior performance over all the other classical machine learning algorithms.

**Index Terms**—Intrusion detection, deep neural networks, machine learning, deep learning.

**Introduction**

In the modern world, fast-paced technological advancements have encouraged every organization to adopt the integration of information and communication technology (ICT). Hence creating an environment where every action is routed through that system makes the organization vulnerable if the security of the ICT system is compromised. Therefore, this calls for a multilayered detection and protection scheme that can handle truly novel attacks on the system as well as be able to autonomously adapt to the new data. There are multiple systems that can be used for shielding such ICT systems from vulnerabilities, namely anomaly detection and IDSs. A demerit of anomaly-detection systems is the complexity involved in the process of defining rules. Each protocol being analyzed must be defined, implemented, and tested for accuracy. Another pitfall relating to anomaly detection is that harmful activity that falls within usual usage pattern is not recognized. Therefore, the need for an IDS that can adapt itself to the recent novel attacks and can be trained as well as deployed by using datasets of irregular distribution becomes indispensable. Intrusion Detect Systems (IDSs) are a range of cybersecurity-based technology initially developed to detect vulnerabilities and exploits against a target host. The sole use of the IDS is to detect threats. Therefore, it is located out-of-band on the infrastructure of the network and is not in the actual real-time communication passage between the sender and receiver of data. Instead, their solutions will often make use of a TAP or SPAN ports to analyze the inline traffic stream’s copy and will try to predict the attack based on a previously trained algorithm, hence making the need of a human intervention trivial.

In the field of cybersecurity, algorithms of machine learning have played an essential part. Especially, due to the incredible performance and potential of deep learning networks in recent days in various problems from a wide variety of fields which were considered unsolvable in past, the reliability of applying it for Artificial Intelligence (AI) and unsupervised challenges have increased [39]. Deep-learning is nothing but a partition of machine-learning that mimics the functions of the human brain and hence the name artificial neural network. The concept of deep learning consists of creating hierarchical representations that are complex that involve the creation of simple building blocks to solve high-level problems.

Therefore, it becomes obvious that Deep neural networks and IDSs, when combined, can work at a superhuman level. Also, since the IDSs are out-of-band on the infrastructure, common attacks like DoS which primarily aims at choking the network band to gain access of the host, cannot bottleneck the performance of it, hence this security layer cannot tamper with ease. Towards the end, the sections are organized as follows: Section II reviews the work related to IDS, different deep neural networks, and some discussions about KDDCup-’99’ dataset that was published. Section III takes an in-depth look at Deep Neural Networks (DNN) and the applications of ReLU activation function. Section IV analyses the dataset used in this paper, explains the shortcoming of it and evaluates the results. Section V concludes and states a plausible workflow into the future of this research work.

**BACKGROUND**

Deep neural networks (DNNs) are Artificial Neural Network (ANN) with a multi-layered structure comprised within the input-output layers. They can model complex non-linear relationships and can generate computational models where the object is expressed in terms of the layered composition of primitives. Below we roughly cover simple DNNs and applications of ReLU and why it is preferred over other activation functions.

1. Deep Neural Network (DNN)

While traditional machine learning algorithms are linear, deep neural networks are stacked in an increasing hierarchy of complexity as well as abstraction. Each layer applies a nonlinear transformation onto its input and creates a statistical model as output from what it learns. In simple terms, the input layer is received by the input layer and passed onto the first hidden layer. These hidden layers perform mathematical computations on our inputs. One of the challenges in creating neural networks is deciding the hidden layers’ count and the count of the neurons for each layer. Each neuron has an activation function which is used to standardize the output from the neuron. The” Deep” in Deep learning refers to having more than one layer which is hidden. The output layer returns the output data. Until the output has reached an acceptable level of accuracy, epochs are continued.

1. Application of rectified linear units (ReLU)

ReLu has turned out to be more efficient and has the capacity to accelerate the entire training process altogether [20]. Usually, Neural networks use a sigmoidal activation function or tanh (hyperbolic tangent) activation functions. But these functions are prone to vanishing gradient problem. Vanishing gradient occurs when lower layers of a DNN have gradients of nearly null because units of higher layers are nearly saturated at the asymptotes of the tanh function. ReLU offers an alternative to sigmoidal non-linearity which addresses the issues mentioned so far.

**Experiment**

We consider Keras as a wrapper on top of TensorFlow as software framework. For exponentially increasing the agility of processing of data in deep-learning architectures, a GPU enabled tensorflow in a single Nvidia-GK110BGL Tesla-k40 has been used.

1. Datasets description

The DARPA’s program for ID evaluation of 1998 was managed and prepared by Lincoln Labs of MIT. The main objective of this is to analyze and conduct research in ID. A standardized dataset was prepared, which included various types of intrusions which imitated a military environment and was made publicly available. The KDD intrusion detection contest’s dataset of 1999 was a well-refined version.

1. Shortcomings of KDDCup-’99’ dataset

ReLu has turned out to be more efficient and have the A detailed report and major shortcomings of the provided synthetic data set such as KDDCup-’98’ and KDDCup-’99” were discussed by [26]. The main condemnation was that they failed to validate their data set simulation of real-world network traffic profile. Irrespective of all these criticisms, the dataset of KDDCup-’99’ has been used as an effective dataset by many researchers for bench-marking the IDS algorithms over the years. In contrast to the critiques about the creation of the dataset, [27] has revealed a detailed analysis of the contents, identified the non-uniformity, and simulated the artifacts in the simulated network traffic data. The reasons behind why the machine learning classifiers have limited capability in identifying the attacks that belong to the content categories R2L, U2R in KDDCup-’99’ datasets have been discussed by [28]. They have concluded that it is not possible to get acceptable detection rate using classical ML algorithms. It is also stated that the possibility of getting high detection rate in most of the cases by producing a refined and augmented data set by combining the train and test sets. However, a significant approach has not been discussed. The DARPA / KDDCup-’88 failed to evaluate the traditional IDS resulting in many major criticisms. To eradicate this [29] used Snort ID system on DARPA / KDDCup-’98’ tcpdump traces. The system performed poorly resulting in low accuracy and the impermissible false positive rates. It failed in detecting dos and probing category but contrasting performing better than the detection of R2L and U2R.

1. DARPA / KDDCup-’99’ dataset

The DARPA’s ID evaluation group, accumulated network-based data of IDS by simulation of an air force base LAN by over 1000s of UNIX nodes and for continuously 9 weeks, 100s of users at a given time in Lincoln Labs which was then divided into 7 and 2 weeks of training and testing respectively to extract the raw dump data TCP. MIT’s lab, with extensive financial support from DARPA and AFRL, used Windows and UNIX nodes for almost all the inbound intrusions from an alienated LAN unlike other OS nodes. For the purpose of dataset, 7 distinct scenarios and 32 distinct attacks which total up to 300 attacks were simulated. Since the year of release of KDD-’99’ dataset, it is the most vastly utilized data for evaluating several IDSs. This dataset is grouped together by almost 4,900,000 individual connections which includes a feature count of 41. The simulated attacks were categorized broadly as given below:

• Denial-of-Service-Attack (DoS): Intrusion where a person aims to make a host inaccessible to its actual purpose by briefly or sometimes permanently disrupting services by flooding the target machine with enormous amounts of requests and hence overloading the host.

• User-to-Root-Attack (U2R): A category of commonly used maneuver by the perpetrator starts by trying to gain access to a user’s pre-existing access and exploiting the holes to obtain root control.

W

• Remote-to-Local-Attack (R2L): The intrusion in which the attacker can send data packets to the target but has no user account on that machine itself, tries to exploit one vulnerability to obtain local access cloaking themselves as the existing user of the target machine.

• Probing-Attack: The type in which the perpetrator tries to gather information about the computers of the network and the aim for doing so is to get past the firewall and gaining root access.

KDDCup-’99’ set is classified into the following three groups: Basic features: Attributes obtained from a connection of TCP/IP comes from this group. Most of these feature’s result in implicitly delaying the detection.

1. Identifying network parameters

Hyper-tuning of parameters to figure out the optimum set of parameters to achieve the desired result is all by itself a separate field with plenty of future scope for research. In this paper, the learning is kept constant at 0.01 while the other parameters were optimized. The count of the neurons in a layer was experimented by changing it over the range of 2 to 1024. After that, the count was further increased to 1280 but didn’t yield any appreciable increase in accuracy. Therefore, the neuron count was tuned to 1024.

1. Identifying network structures

Conventionally, increasing the count of the layers results in better results compared to increasing the neuron count in a layer. Therefore, the following network topologies were used to scrutinize and conclude the optimum network structure for our input data.

• DNN with 1,2,3,4,5 layers. For all the above network topologies, 100 epochs were run and the results were observed. Finally, the best performance was showed by DNN 3 layer compared to all the others. To broaden the search for better results, all the common classical machine learning algorithms were used, and the results were compared to the DNN 3 layer, which still outperformed every single classical algorithm. The detailed statistical results for different network structures are reported in table I.

1. Proposed Architecture

An overview of proposed DNNs architecture for all use cases is shown in Fig. 1. This comprises of a hidden-layer count of 5 and an output-layer. The input-layer consists of 41 neurons. The neurons in input-layer to hidden-layer and hidden to output-layer are connected completely. A back-propagation mechanism is used to train the DNN networks. The proposed network is composed of fully connected layers, bias layers, and dropout layers to make the network more robust.

|  |  |  |
| --- | --- | --- |
| **TABLE 1**  **NETWORK STRUCTURE INFORMATION** | | |
| Layer (type) | Output Shape | Param |
| Dense-1 (Dense) | (NIL, 1024) | 43008 |
| Dropout-1 (Dropout) | (NIL, 1024) | 0 |
| Dense-2 (Dense) | (NIL, 768) | 787200 |
| Droupout-2 (Dropout) | (NIL, 768) | 0 |
| Dense-3(Dense) | (NIL, 512) | 393728 |
| Droupout-3 (Dropout) | (NIL, 512) | 0 |
| Dense-4(Dense) | (NIL, 256) | 131328 |
| Droupout-4 (Dropout) | (NIL, 256) | 0 |
| Dense-5(Dense) | (NIL, 128) | 32896 |
| Droupout-5 (Dropout) | (NIL, 128) | 0 |
| Dense-6(Dense) | (NIL, 1) | 129 |
| Activation-1 (Activation) | (NIL, 1) | 0 |

Input and hidden layers: This layer consists of 41 neurons. These are then fed into the hidden layers. Hidden layers use ReLU as the non-linear activation function. Then weights are added to feed them forward to the next hidden layer. The neuron count in each hidden layer is decreased steadily from the first to the output to make the outputs more accurate and at the same time reducing the computational cost. Regularization: To make the whole process efficient and time-saving, Dropout (0.01). The function of the dropout is to unplug the neurons randomly, making the model more robust and hence preventing it from over-fitting the training set. Output layer and classification: The out layer consists only of two neurons Attack and Benign. Since the 1024 neurons from the previous layer must be converted into just 2 neurons, a sigmoid activation function is used. Due to the nature of the sigmoid function, it returns only two outputs, hence favoring the binary classification that was intended in this paper.

Fig. 1. Proposed architecture



F1

F2

F41

Input Layer

Hidden Layer1

H1,1

H1,2

H1,1024

Hidden Layer5



H2,1

H2,2

H1,128

Attack

Benign

Output Layer

**Results**

For the scope of this paper, the KDDCup-’99’ dataset was fed into classical ML algorithms as well as DNNs of varying hidden layers. After the training was completed, all models were compared for f1-score, accuracy, recall and precision with the test dataset. The scores for the same have been compared in detail. DNN 3-layer network has outperformed all the other classical machine learning algorithms. It is so because of the ability of DNNs to extract data and features with higher abstraction and the non-linearity of the networks adds up to the advantage when compared with the other algorithms.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **RESULTS** | | | | |
| Algorithm | Accuracy | Precision | Recall | f1-score |
| DNN-1 | 0.929 | 0.998 | 0.915 | 0.954 |
| DNN-2 | 0.929 | 0.998 | 0.914 | 0.954 |
| DNN-3 | 0.930 | 0.997 | 0.915 | 0.955 |
| DNN-4 | 0.929 | 0.999 | 0.913 | 0.954 |
| DNN-5 | 0.927 | 0.998 | 0.911 | 0.953 |
| Ada Boost | 0.925 | 0.995 | 0.911 | 0.951 |
| Decision Tree | 0.928 | 0.999 | 0.912 | 0.953 |
| K-Nearest Neighbour | 0.929 | 0.998 | 0.913 | 0.954 |
| Linear Regression | 0.848 | 0.989 | 0.821 | 0.897 |
| Navie Bayes | 0.929 | 0.988 | 0.923 | 0.955 |
| Random Forest | 0.927 | 0.999 | 0.910 | 0.953 |
| SVM Linear | 0.811 | 0.994 | 0.770 | 0.868 |
| SVM-rbf | 0.811 | 0.992 | 0.772 | 0.868 |

**Conclusion**

This paper has elaborately recapitulated the usefulness of DNNs in IDS comprehensively. For the purpose of reference, other classical ML algorithms have been accounted and compared against the results of DNN. The publicly available KDDCup-’99’ dataset has been primarily used as the benchmarking tool for the study, through which the superiority of the DNN over the other compared algorithms have been documented clearly. For further refinement of the algorithm, this paper considers DNNs with different counts of hidden layers and it was concluded that a DNN with 3 layers has been proven to be effective and accurate of all. Since the neurons are trained with a bygone benchmarking dataset, as discussed, several times in this paper, this comes as a pitfall for this methodology. Fortunately, it can be vanquished by using a fresh dataset with the essences of the latest attack strategies before the actual deployment of this artificial intelligence layer to the existing network systems to ensure the agility of the algorithms real-world capabilities. From the empirical results of this paper, we may claim that deep learning methods are a promising direction towards cyber security tasks, but even though the performance on artificial dataset is exceptional, application of the same on network traffic in the real-time which contains more complex and recent attack types is necessary. Additionally, studies regarding the flexibility of these DNNs in adversarial environments are required. The increase in vast variants of deep learning algorithms calls for an overall evaluation of these algorithms in regard to its effectiveness towards IDSs.

**References**

[1] R. Lippmann, J. Haines, D. Fried, J. Korba and K. Das. ”The 1999 DARPA off-line intrusion detection evaluation”. Computer networks, vol. 34, no. 4, pp. 579 595, 2000. DOI http://dx.doi.org/10.1016/S1389- 1286(00)00139-0.

[2] W. Lee and S. Stolfo. ”A framework for constructing features and models for intrusion detection systems”. ACM transactions on information and system security, vol. 3, no. 4, pp. 227261, 2000. DOI http://dx.doi. Org/10.1145/382912.382914.

[3] B. Pfahringer. ”Winning the KDD99 classification cup: Bagged boosting”. SIGKDD explorations newsletter, vol. 1, pp. 6566, 2000. DOI http://dx.doi.org/10. 1145/846183.846200.

[4] M. Vladimir, V. Alexei and S. Ivan. ”The MP13 approach to the KDD’99 classifier learning contest”. SIGKDD explorations newsletter, vol. 1, pp. 76 77, 2000. DOI http://dx.doi.org/10.1145/846183. 846202.

[5] R. Agarwal and M. Joshi. ”PNrule: A new framework for learning classier models in data mining”. Tech. Rep. 00-015, Department of Computer Science, University of Minnesota, 2000.

[6] C. Elkan. “Results of the KDD’99 classifier learning”. SIGKDD explorations newsletter, vol. 1, pp. 63 64, 2000. DOI http://dx.doi.org/10.1145/846183. 846199.

[7] S. Sung, A.H. Mukkamala. ”Identifying important features for intrusion detection using support vector machines and neural networks”. In Proceedings of the symposium on applications and the Internet (SAINT), pp. 209216. IEEE Computer Society, 2003. DOI http: //dx.doi.org/10.1109/saint.2003.1183050.

[8] H. Kayacik, A. Zincir-Heywood and M. Heywood. ” Selecting features for intrusion detection: A feature relevance analysis on KDD 99 intrusion detection datasets”. In Proceedings of the third annual conference on privacy, security and trust (PST). 2005.

[9] C. Lee, S. Shin and J. Chung. ”Network intrusion detection through genetic feature selection”. In Seventh ACIS international conference on software engineering, artificial intelligence, networking, and parallel/distributed computing (SNPD), pp. 109114. IEEE Computer Society, 2006.

[10] S. Chavan, K. Shah, N. Dave, S. Mukherjee, A. Abraham and S. Sanyal. ”Adaptive neuro-fuzzy intrusion detection systems”. In Proceedings of the international conference on information technology: Coding and computing (ITCC), vol. 1, pp. 7074. IEEE Computer Society, 2004. DOI http://dx.doi.org/ 10.1109/itcc.2004.1286428