Analysis of Machine Learning in Intrusion Detection Systems

The Machine Learning Approach to Detecting Cyber Intrusions

Brian Asta
Department of Network and Computer Security
SUNY Polytechnic Institute, Utica, NY 13502

Abstract

With the rapid expansion of computers and the networks connecting them, there has been an equally rapid need for security. Intrusion detection systems (IDS) are an integral part of keeping computer systems secure. IDS help administrators keep tabs on unusual activity on a network or device. The problem with IDS, in their current state, is they are not able to adapt to the constantly growing networks and the changing world of malicious activity. With the introduction of machine learning in IDS it may be possible for these systems to be more accurate, efficient, and flexible, allowing them to catch new and unforeseen attacks. With implementing new technologies come new challenges, especially with developing an accurate and dynamic IDS. This paper presents a brief introduction into this topic, including the NSL-KDD Dataset, a comparison and analysis of machine learning techniques applied to IDS, a preliminary attempt to make an intelligent IDS, and a look at the current commercial market.

Keywords—Network Security; IDS; Anomaly Detection; Neural Network; Machine Learning; NSL-KDD

I. Introduction

Intrusion detection systems are an important utility to have on a network. They are used to monitor, analyze, and alert administrators to unusual or malicious activity [2][14]. IDS are usually placed within the network and behind a firewall, so if anything gets past the firewall it can be picked up on. When an IDS raises an alert, an intrusion prevention system (IPS) or the administrators can take further action. Intrusion detection systems come in two varieties: host based and network based. Host based IDS (HIDS) are pieces of software on a device or system, such as an antivirus program. This software will usually look at system files or behavior of a user, such as programs and commands used. Relying on HIDS is most appropriate for a home or small business that does not have a large network. Network based intrusion detection systems (NIDS) [18] are usually placed at the network's Internet connection or behind a firewall. NIDS will look at the network traffic and activity for the entire network. This is better for large companies as it allows for the IDS to be configured and managed, and alerts to be analyzed in a central point. This central location allows one IDS to cover an entire network. There are also two different ways and IDS can identify an intrusion: signature-based and anomaly-based. Signature-based detection uses pattern matching to compare the incoming and outgoing traffic to a set of rules. These rules are based off a list on known malicious or unwanted activity. Anomaly-based detection gathers a basis for "normal" traffic and if there are any deviations from it, an alert will be made (hence the name anomaly detection). Each of these methods have their strengths and weaknesses. Signature-based detection can be very accurate when detecting known attacks, minimizing false positives (a "false positive" is when non-malicious traffic gets flagged as malicious). When faced with an unknown attack, accuracy may be affected and this may result in a false negative (a "false negative" is when malicious traffic is falsely marked as non-malicious). Anomaly detection is theoretically better for detecting novel attacks [2]. The reason for this is anomaly detection looks for traffic that is "not normal". So instead of looking for known malicious activity, it is just looks for activity that is out of the ordinary. This can cause high rates if false positives [4]. While false positives can be annoying for administrators, false negatives are the real issue, because then malicious activity has gone unnoticed. The application of machine learning may be the remedy for the weaknesses of these detection methods.

Machine learning, in its most basic definition, is the ability for a computer or program to progressively improve at doing a task. There are two main ways a machine can learn: supervised and unsupervised learning. In supervised learning the machine learning algorithm (MLA) is "trained" using data with known character (malicious or not). In unsupervised learning, the MLA finds patterns representing "normal" (non-malicious) behavior and draws its own conclusions from a dataset. The ability of machine learning to deal with and learn from data makes it appealing to researchers across many

fields. Machine learning may allow intrusion detection systems to be more accurate and efficient to accommodate bigger and faster networks, but may also aid in the detection of unknown and unforeseen malicious activity. Machine learning may be able to make signature-based detection more dynamic so it can pick up on new attacks and allow anomaly-based detection to improve in accuracy over time. This should cut down on both false positives and false negatives in both detection methods. As with any developing technology, there are many challenges [4]. The overarching problem is to implement an ineffective IDS into a network, that is complicated by two other problems. The first of which is the lack of voluminous, real world, labeled datasets. Since the machine needs data to build a model, if the data is not representative of real world networks then the model will not be useful and will put the IDS at a disadvantage. The main contributing factor to this problem is companies do not want to give out their network data due to confidentiality. The second problem is feature selection. Feature selection is picking out appropriate parts of the data that correctly identify attacks. The problem arises from the fact that different attacks have a variety of features [13], and future attacks may have unknown features. This makes it difficult to generate a set of features that covers all the bases but is not so large that it is computationally expensive to work with. With more time and research we may be able to overcome these challenges.

II. The NSL-KDD Dataset

A large part of machine learning comes down to the dataset. Even though there is a lack in abundance of datasets, there are a few that suit this application. One of them is the Network Socket Layer-Knowledge Discovery and Data mining (NSL-KDD) Dataset, currently considered to be the benchmark dataset for training and testing IDS [1]. This is evident since analysis of this dataset has been the focus of a number of papers already [1, 2, 5, 9, 10, 15]. Following is a brief review of the dataset.

A. Origins and evolution of the dataset

The NSL-KDD dataset is a revised version of the KDD Cup '99 dataset which is derived from the DARPA 98 dataset [9, 10, 15]. The KDD 99 dataset was flawed due to the inclusion of many redundant records in the training and testing sets [10, 15]. This caused the results of the machine learning to be skewed toward some attacks over others. Some improvements present in the NSL-KDD dataset include [5]:

- Removal of redundant records in the training and testing set, resulting in non-biased classifiers or results.
- Inverse proportionality of records from each difficulty group over the original KDD '99 dataset, resulting in more accurate testing of different learning methods.
- Reduction of redundant records also makes the dataset easier to run, requiring less computational resources.

B. Content of the dataset

The NSL-KDD dataset contains many different files including the full set and percentages taken out for training and testing. Each subset contains traffic deemed to be non-malicious (normal) as well as traffic that is known to be malicious. The classes of attacks in the dataset are broken down into four main categories. Each of these categories have a subset of attack types.

- DOS: Denial of service attack
- Probing: Gathering information about the target
- U2R: Unauthorized local admin access
- R2L: Unauthorized access from a remote machine

The breakdown of the number of records in each attack category is shown in Table 1 below. The table shows the contents of each subset and contains the count and percent of records in each.

Table 1: Breakdown of data in the NSL-KDD subsets

Data subset	Total Records	Normal Class	DOS Class	Probe Class	U2R Class	R2L Class
Trian 20%	25192	13449	9234	2289	11	209
		53.39%	36.65%	9.09%	0.04%	0.83%
Train	1259373	67343	45927	11656	52	995
		53.46%	36.46%	9.25%	0.04%	0.79%
Test	22544	9711	7458	2421	200	2754
		43.08%	33.08%	10.74%	0.89%	12.22%

This data is from [1].

Each set contains normal records and records of each attack type, all representative of real world networks. Since DOS attacks are more common than R2L attacks, there are more records for DOS than R2L.

Next, you can see that the training set is broken into two pieces, a full set and a 20% subset. A smaller subset is given so you can train on a smaller scale to cut down on computational costs, if need be. This allows for researchers to compare MLA without feeding them the entire sets. The testing set is also significantly smaller than the full training set but contains a higher number of malicious records. The full training set is larger because it will be the basis for all future classification of normal and malicious traffic. Thus, it contains a slightly more even distribution of records, in terms of normal versus malicious (53.46% vs. 46.54%). The testing set contains, proportionally, more malicious records so it can

effective test the accuracy of the training by seeing if it can recognize all the bad traffic.

The NSL-KDD dataset is the best dataset currently available for the application of machine learning in intrusion detection systems.

C. Application of the dataset

There are two main uses for this dataset. The first is to apply it to the neural network for training and testing. This is what was talked about above, using the records in the dataset to train and test the accuracy of the machine learning algorithm. Using the dataset for this purpose allows for direct comparison to other researcher's works. The second use is just for research into optimal feature selection [7, 9]. This would be done if a researcher wants to use their own data but still compare it using the features listed in the dataset. Using custom curated datasets allows researchers to tailor the training and testing to their network and needs. The works referenced in this paper make use of the dataset in one or both ways.

D. Criticism of the dataset

Although the NSL-KDD dataset is considered the benchmark for testing machine learning IDS there is some criticism. The biggest criticism comes from McHugh [15]. This study considers the original DARPA dataset. The first criticism was the lack of background data or "noise". The lack of this background data allows for the IDS to have a lighter than real-world load on it. This could cause results for performance and accuracy to be inaccurate because the IDS won't have to sift through the noise. The second criticism was that the attacks were synthesized. On top of that, no attempt was made to make sure the attacks were distributed realistically into the normal traffic data. In the paper, when considering these shortcomings, the author said "Perhaps and perhaps not." The author goes on to say that many tests are done in contrived environments to control factors that could "confound the results". Following this McHugh said that it would be the responsibility of the researchers to show that the environment did not affect the results. These are valid points: how can we be sure this artificial data is really representative of real world networks?

Since McHugh's paper was written in 2000 and this dataset came out in 1998-99, both the data and the criticisms are somewhat out of date in 2018 (more so the dataset). The NSL-KDD Dataset, and its revisions, is still derived from a twenty-year old dataset. Our networks have changed very much in this time. On top of research for performance comparison for MLA, more work needs to focus on generating new datasets. This is the only way to validate IDS research results.

III. Previous works and machine learning techniques for IDS

Although a relatively new topic, researchers and scholars have done a considerable amount of work in applying machine learning to intrusion detection systems. Some of this work spans back ten years. Most of the studies focus less on the actual IDS part and more on the accuracy of machine learning techniques.

A. Deep Belief Network & Support Vector Machine Hybrid [6]

Salama, Ramadan, and Darwish used a hybrid MLA method to achieve their results. The researchers used Deep Belief Network (DBN), a type of deep learning neural network, along with Support Vector Machine (SVM), a supervised learning algorithm. The DBN was used as a feature selector and was compared to other methods of feature selection. The researchers were successful in reducing the number of features from 41 to 13. SVM was used as a classifier to separate the data from the NSL-KDD dataset into the 5 categories listed above. Independently, on 40% of the training set DBM and SVM resulted in 88.33% and 89.54% accuracy respectively. When both were implemented with the proposed hybrid method, an accuracy of 92.84% was achieved.

B. IDS and ML IDS Comparison [19]

In this work, the authors directly compared a standalone IDS to an IDS with machine learning applied. This was a very in-depth paper about virtual lab setup, traffic rate comparison, and MLA testing. The comparison starts with the standalone IDS systems Snort and Suricata. The authors compared how traffic flow affects computational cost and detection accuracy of the IDS. They concluded that Snort was the better of the two to apply machine learning to because it was better in all the tests. The average accuracy for Snort alone was 92.11%. They also compared five different MLA: Support Vector Machine, Decision Trees, Fuzzy Logic, Bayes Net, and Naïve Bayes. Out of these five, SVM achieved the highest average detection accuracy of 95.06% followed by Fuzzy Logic at 93.8%. When SVM was applied to Snort a detection accuracy of about 98% was achieved. This work showed that the application of machine learning provides a noticeable improvement.

C. Feature selection using SVM [20]

In this work by Mukkamala, Janoski, and Sung focused on feature selection and how it affected computational costs and accuracy. The machine learning algorithm the authors focused on was Support Vector Machine and its strengths in intrusion detection. The two main strengths were real-time performance and scalability for the number of features. Similar to the first work discussed here, the authors were able to pick out a subset of 13 of the 41 features in the NSL-KDD dataset. Two separate experiments compared accuracy, computational time, and scalability. The first experiment tested the full set of 41

features against the reduced subset of 13 features on a dataset of 14,000 records. The CPU runtime for the set of 41 features was 1.60 minutes and for 13 features it was 1.06 minutes. Meanwhile accuracy was nearly unaffected: 41 features resulted in 99.53% and 13 features resulted in 99.52%. These results were compared to the results of a second experiment where the authors used a 55,000-record dataset. For 41 features the results were 15.44 min. and 99.60% accuracy and 13 features resulted in 10.04 min. and 99.57% accuracy. These results show the how well accuracy and speed scale with SVM as a classifier.

D. Flow-Based Anomaly Detection [17]

While normally intrusion detection is based on packet inspection, the approach by Yousef and Jovanovic used NetFlow data. NetFlow data is just the basic information from the packets, such as information from the headers. This means that there the actual data from the packet is not analyzed. The upside to this method is companies might be more inclined to share this type of network traffic to researchers as it contains less payload information. In this work they use the NSL-KDD Dataset for feature selection but they used their own data for training and testing. Their testing was done with an anomaly-based intrusion detection. A neural network-based two-stage system was proposed. This consisted of a flow collector, feature preparation, the first neural network stage for anomaly detection, the second stage for classification, and then a way of generating alerts. With this model they tested three different algorithms: Resilient Backpropagation, Levenberg-Marquardt, and Radial Basis Function. Each of these algorithms were tested in both neural network stages. In stage one the detection rate results were as follows: 92.7% (RBP), 94.2% (LM), and 91.1% (RBF). In the classification stage they obtained 99.4%, 99.42% and 95.4% classification accuracy, respectively. Stage one results are most important because they represent the IDS ability to detect anomalous traffic. Stage two results represent accuracy with which the IDS correctly classifies anomalous records. Surprisingly, these results were quite accurate despite not looking at the actual packet data.

E. User behavior classification [3]

The work by Ryan, Lin, and Miikkulainen took a completely different approach, looking at user behavior on a system. Since this method did not look at network traffic and the scope was limited to one system, this would be classified as a host based intrusion detection system. The data for these tests were collected and tested on a system at the University of Texas in the Electrical and Computer Engineering Department. The authors chose this system because its operating system, NetBSD, was able to give them audit trail logging. This allowed for easy collection of data. The other reason they picked this system was because the users were known to the researchers and it was unlikely that an unknown user would connect to it. This allowed for a more controlled data gathering and testing. This study is an example of unsupervised learning applied to intrusion detection since the MLA needs to detect patterns and draw its own conclusions from the data.

As for results, the method resulted in proper user identification in 22 of 24 cases and an anomaly detection rate of 96%. One problem with this study was the small number of users on the system. With only 10 total users logging in from time to time, the sample size was very small. The authors said this was a contributing factor to why this system was chosen. This would mean the data they collected was manageable to work with. The NNIDS was also easy to train, being based on commands entered by users as reported in log files. In addition, it was computationally inexpensive to run and could work offline on logs generated during the day.

Some downfalls to this method are apparent. Without real-time detection, this method may be too late to pick up on intrusions. Another problem is scalability: this was tested on a system with 10 or so users. If this was scaled up to real-world network sizes of 1000's of users it would most likely negatively affect performance. Both real-time detection and scalability issues make this method ineffective due to computational needs.

F. Conclusion of previous works

Although a small sample, the research papers discussed here were representative of a wide variety of methods and used a variety of machine learning techniques in intrusion detection. These works also illustrated the performance differences among different MLA and IDS schemes with ML. Our review of the literature did not find a single study that showed machine learning degrading the performance of intrusion detection systems.

IV. My attempt and project proposal

My project originated as an attempt to make a Neural Network Intrusion Detection System of my own. My idea was to make a plug-in of sorts that added machine learning functionality on top of Snort, an open source IDS. After doing work on it I concluded that my ability to make the NNIDS in the time constraint for my capstone project was not there. Here is a summary of what I did accomplish as well as where I would like to take this project given more time and resources.

A. Materials and process

In my attempt I used Snort [11], a signature-based intrusion detection system. On my computer I set up a virtual environment that consisted of three virtual machines. One machine was the IDS, another was the attacker, and the last was the target. I installed and setup Snort; this consisted of setting the VM up with enough resources so it would not be bottlenecked. After that I downloaded the dependencies and installed Snort. Configuration took a while as I was unfamiliar with how Snort was supposed to be set up. It took

considerable effort to get it up and running. Since Snort is a signature-based IDS I had to get a rule set. For this I used the registered rule set. With these rules downloaded and configured it was time to test it. At first, I used commonly known attacks and without a hitch, Snort picked up on them. At this point I did some research to see how rules were made. I picked a premade rule (for ping scans) and turned detection of it off, then I made my own rule in its place. After a few attempts I was able to make rules of my own.

This is where I wanted my "plug-in" to come into play. My plan was to implement machine learning into Snort using TensorFlow [12], a Python machine learning library. This "plug-in" would take information from Snort for training and testing purposes. I started to run into issues with prototyping the Python and TensorFlow part of my project. I ran into many errors, tried to troubleshoot them, and tried to find resources that might help. At this point I was forced to pivot away from a hands-on project. I didn't have the knowledge of machine learning and TensorFlow to continue.

B. Future direction

Two research papers [19, 20] were very similar to what I wanted to do. If I had more time this is the direction I would have liked to had taken my project, directly comparing performance (accuracy and efficiency) of IDS and IDS with machine learning applied. I also would have liked to continue to develop such a plug-in to implement MLA into IDS. The only thing I did not see in any of the papers was the detection of unknown attacks. This would be the main goal of my future work.

V. Current commercial market

Machine learning in cybersecurity is not limited to intrusion detection systems. Several companies are developing new products that use machine learning or implementing machine learning into their current products. The first one I looked into was Chronicle. This was both the newest and biggest. Chronicle is a daughter company of Alphabet, which is the parent company of Google. I also found two small startups working on intelligent IDS, Darktrace and BluVector. None of these companies give much information about their products, which is understandable because they are trying to sell them. This research shows that there is somewhat of a demand for these types of products.

Moving away from companies that are producing machine learning cybersecurity products, I looked at processor manufacturers that are making machine learning computer chips. The application of these are not limited to cybersecurity but may be able to assist in this field. Some of the big companies I looked at include:

- Intel
- Nvidia
- ARM

Intel and ARM are trying to make dedicated machine learning processors as well as trying to implement ML into enterprise and consumer products for better computational power. Nvidia on the other hand makes graphics processor chips. Over the past few years, it has become obvious that GPUs have a shear processing advantage over conventional CPUs. Nvidia's products will continue in the place of normal GPUs but improve performance by offloading the workload onto a machine learning card. Both of these types of processors could be implemented into future IDS to take some of the load off the main processor. In addition, moving the machine learning to a dedicated processor could make the MLA more efficient and can improve security of the IDS itself.

VI. Conclusion

This paper reviewed the basics of intrusion detection systems and the application of machine learning algorithms, including associated challenges. The NSL-KDD dataset was also reviewed. Despite some of the criticisms of its predecessor datasets, the NSL-KDD Dataset is currently the most effective way of benchmarking neural network intrusion detection systems due to its feature selection and large but manageable number of records. More research needs to be conducted to generate new, modern datasets that are more representative of real-world networks.

Several previous works on this topic were reviewed in detail. The works covered a variety of topics including feature selection, detection accuracy, and computational cost. Detection accuracy results varied from the high 80% to the high 90%. This wide range of results is expected when implementing new technologies into IDS. The beneficial side to machine learning is that it can improve on accuracy and detection ability over time. More research will be needed to test the ability to detect new attacks.

This paper also documented the attempt to build a NNIDS and the associated challenges. A working machine learning IDS was not completed, however a short project proposal was produced. This laid out a clear direction for the project if given more time and resources.

Lastly, this paper reviewed the current commercial market for machine learning based IDS. Several companies are working to produce intelligent IDS. The hard part will be convincing enterprise consumers to adopt this new technology. This paper also briefly discussed machine learning processors and their possible application to IDS. Their ability to take some workload off the main processor and maybe even make the MLA more efficient (as it would be on a dedicated chip) could make these very beneficial in IDS.

Even with all the research that has been completed for this topic there is still work to do to create an efficient and effective flexible IDS. As networks increase in size, speed,

and traffic flow, faster and more effective NNIDS will have to be researched and developed. Machine learning shows a promising future in making intrusion detection systems faster, more accurate, and adaptive. The application of machine learning is not restricted to IDS, and the possibilities are endless. It is clear that machine learning techniques may be crucial tools for the future of cybersecurity, but they may also enable future cyberattacks, a prospect that warrants further research.

References

[1] L. Dhanabal, and S.P. Shantharajah, "A Study on NSL-KDD Dataset for Intrusion Detection Systems Based on Classification Algorithms," in International Journal of Advanced Research in Computer and Communication Engineering, Vol. 4, Issue 6, June 2015

[2] Q. Niyaz, W. Sun, A.Y. Javaid, and M. Alam "A Deep Learning Approach for Network Intrusion Detection System" University of Toledo College of Engineering

[3] J. Ryan, M. Lin, and R. Miikkulainen "Intrusion Detection with Neural Networks,"

[4] P. Garcia, J. Diaz, G. Macia, and E. Vazquez "Anomaly-based network intrusion detection techniques, systems and challenges" University of Granada

[5] http://www.unb.ca/cic/datasets/nsl.html

[6] M.A. Salama, H.F. Eid, R.A. Ramadan, A. Darwish, and A.E.Hassanien "Hybrid Intelligent Intrusion Detection Scheme"

[7] M. Moradi, and M. Zulkernine "A Neural Network Based System for Intrusion Detection and Classification of Attacks.

[8] M. Zamani, and M. Movahedi "Machine Learning Techniques for Intrusion detection" Dept. of Computer Science University of New Mexico, 9 May 2015

[9] H. Kayacik, A. Zincir-Heywood, and M. I. Heywood "Selecting Features for Intrusion Detection: A Feature Relevance Analysis of KDD 99 Intrusion Detection Datasets," Dalhousie University, Faculty of Computer Science

[10] S. Revathi, A. Malathi "A Detailed Analysis on NSL-KDD Dataset Using Various Machine Learning Techniques for Intrusion Detection," PG and Research, Dept. of Computer Science, Dec. 2013

[11] https://www.snort.org/

[12] https://www.tensorflow.org/

[13] J. Frank "Artificial Intelligence and Intrusion Detection: Current and Future Directions," University of California at Davis, Division of Comp Sci. June 1994

[14] D. Denning "An Intrusion-Detection Model," IEEE Transaction of Software Engineering, Vol, SE-13, No. 2, 1987

[15] J. McHugh "Testing Intrusion Detection Systems: A Critique of the 1990 and 1999 DARPA…" ACM Transaction of Information and System Security, Vol. 3, No. 4, Nov. 2000

[16] M. Mannien, "Using Artificial Intelligence in Intrusion Detection Systems," Helsinki University of Technology

[17] A Yousef, Z. Jovanovic "Flow-Based Intrusion Detection Systems using Neural Network," International Conference on Internet Computing, July, 2012

[18] SANS Institute "Application of Neural Networks to Intrusion Detection," SANS Institute 2001

[19] S.A. Raza, B. Issac, "Performance Comparison of Intrusion Detection Systems and Application of Machine Learning to Snort System" 2017

[20] S. Mukkamala, G. Janoski, and A. Sung "Intrusion detection: Support Vector Machines and Neural Networks"