Machine Learning Intrusion Detection: A Survey

Andrew Carr

April 2019

1 Introduction

With the increased interconnectivity of humanity via the internet comes a new set of security challenges. A large amount of value is derived from this connectivity since the internet is used heavily by governments, businesses, and other organizations. With that value comes nefarious attackers whose goal is to exfiltrate value from these networks and connections.

Methods used to secure networks, such as firewalls, work well when preventing external users from accessing the internal network, but generally have no mechanism for detecting if an attacker has access to the internal system [23]. Therefore, the use of intrusion detection systems (IDS) has emerged as a way to protect the internal value generating resources on the network. These intrusion detectors run on an organization's internal network and monitor varied sources of traffic. The systems alert security personnel when abusive or unusual activity is found on the network.

These systems operate under the assumption that attackers on the internal network will behave differently than approved users. For example, a member of the Human Resources department never accesses user credit card information, and such actions (by a masquerading attacker) will be flagged as suspect. Similarly, there are often commands and actions taken in known attacks that are drastically different than normal network activity. As such, a rule based or machine learning intrusion detection system system can identify this deviant behavior and alert a security engineer.

Traditional intrusion detection systems are rule based, and look for certain shell commands or operations performed on the network. [20] suggests that due to the large number of commands, it is extremely easy to overlook potential malicious behavior and these methods are "unusable". As such, much of the community ([5], [2], [17], [10], [18], [19], [22], [7]) has adopted machine learning techniques to improve the accuracy of their intrusion detection tools.

Machine learning is a general purpose pattern recognition tool that allows systems to discover trends in large amounts of data. These trends are typically not feasible to model by rules based systems. This is due to the fact that the space of potential attacks is too large and varied to easily capture in code. Much of machine learning is based on statistical analysis, probabilistic modeling, and optimization. These techniques, as discussed in this work, can be leveraged in myriad applications and have achieved success in intrusion detection [15].

2 Intrusion Detection

One reason intrusion detection is a useful, and interesting, field of study is due to the fact that attackers always seek to maintain novelty in their attacks. This novelty increases their likelihood of

exploiting a previously unknown vulnerability in the system. In the security community, intrusion detection is generally categorized into two subfields: Anomaly detection and Misuse based detection ([23], [21], [1]).

2.1 Anomaly based

Anomaly based intrusion detection attempts to identify and detect attacks based on normal network traffic and behavior. As such, any actions that deviate from standard, non malicious, behavior are interpreted as intrusions. This field of intrusion detection has seen the largest application of machine learning [1] as it deals with high volume historical data and trend analysis. Anomaly detection in general is an active area of research [3] with a rich volume of literature. A deep dive into anomaly detection is beyond the scope of this work, but a brief summary is given.

Anomalies are often referred to as outliers, surprises, or exceptions depending on the area of research. As seen in figure (1), anomalies often have the same form, but are identified by other features (y position in the figure).

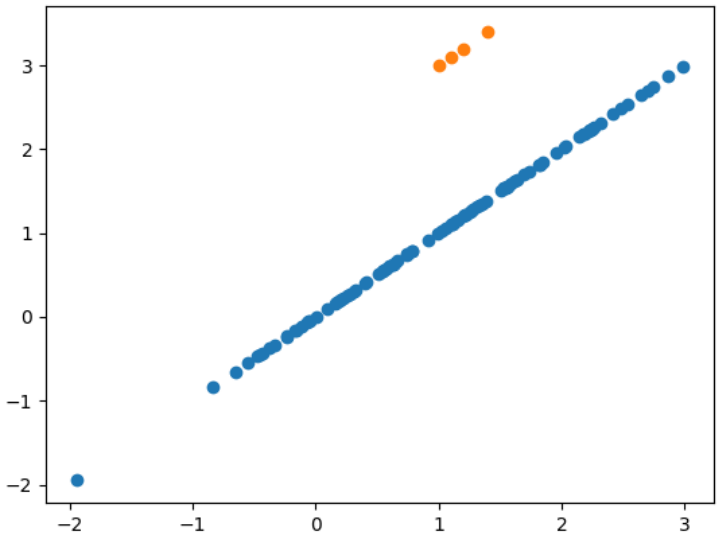

Figure 1: Example of an anomaly group, colored in orange

This is a useful paradigm for intrusion detection systems since attacker behavior often has some similar characteristics to normal network usage, but is identifiable by other means [7].

However, one serious drawback to this model of intrusion detection is the high number of false positives [14]. This is because many of the methods used are not sophisticated enough to differentiate between abnormal activity triggered by an authenticated user, and the abnormal behaviour of an intruder to the network.

2.2 Misuse based

In contrast, misuse based intrusion detection is designed and intended to recognize known attack patterns against a particular system. This means that a database of common attacks is kept on the network. The IDS monitors traffic and compares it against the data base of known patterns. This works well against unintelligent attackers using out of the box software. However, if there is a moderately novel attack being perpetrated on the system, it is highly unlikely that misuse based systems will flag the attack as suspicious. Also, extremely high profile attacks on large targets often will be completely novel and often side step misuse based detection tools. In [8] they find that a majority of IDS (at the time) solely employ misuse based systems. These systems have accuracy of around 74% depending on the metric. They also note, however, that by combining both misuse and anomaly detection methods gives a boost to around 80% accuracy. Therefore, it is not uncommon for modern systems to utilize a combination of misuse and anomaly detection into a single, powerful, intrusion detection system that works well against a majority of attack vectors.

3 Machine Learning Approaches

In recent years a large number ([23], [12], [15], [9], [21], [6], [13], [26], [25], [1]) of works have been published introducing machine learning techniques to the problem of intrusion detection. While there are thousands of machine learning techniques, several key methods have been used with success. In particular, many groups make use of prototypical machine learning methods that are often considered standard, or best practice [24].

3.1 Standard Methods

3.1.1 Supervised

When considering supervised machine learning methods for intrusion detection, it is critical to note that these algorithms require labeled data pairs $(x, y) \in D$ where x is a set of features (e.g., network trace) and y is a label that indicates if the features correspond to an intrusion.

As noted in [13] there are three key observations, and trade offs, needed when designing a supervised IDS.

- Labels are often be difficult or even impossible to procure. The manual analysis of the proper traffic or logs is time-consuming and only a subset of the data can be properly labeled.
- Additionally, labels are only valuable to a certain granularity. For example, at the packet level it is often unreasonable to assign labels of true/false when identifying intrusions.
- Finally, in a real application, it is impossible to guarantee that the labels available for the training of the supervised system covers all possible attacks. It is often the case that a new attack can appear that may not have been covered in the training data.
  - Note: When paired with less sophisticated anomaly detection systems, this shortcoming can be partially mitigated.

With these trade offs in mind, there are a number of algorithms explored in relevant work.

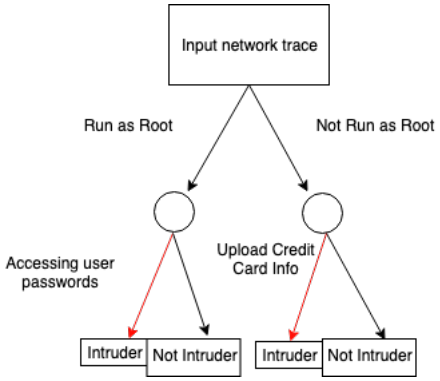

Figure 2: Example of a very simple decision tree on a network trace. We see two paths lead to intruder leaf nodes, and two lead to non intruder nodes. These paths are learned from historic data.

Support Vector Machines (SVM) are a widely used machine learning algorithm for a number of reasons. Simply put, they are computationally efficient and generally fit messy data well. A Support Vector Machine first maps the input features into a high dimensional space. It then learns an optimal separating hyper-plane between the data points of different labels. This is done using a hinge loss function.

$$\begin{aligned} \hat{y} &= Wx + b \\ \ell(\hat{y}, y_t) &= max(0, 1 - y_t(Wx + b)) \end{aligned} \tag{1}$$

This means that intruder traces will be in one high dimensional cluster, while normal activity will be in another. These attributes have made them widely adopted in the intrusion detection literature. In [4] they claim SVMs work better than other methods because they scale well with the number of data points. Additionally [13], [12], and [6] found that SVMs are one of the best algorithms in that they have fewer false positives and false negatives while more often producing true positives and true negatives. One draw back, however, pointed out in [21] is that SVMs tend to find similarities and patterns and may not generalize well to scenarios where outlier detection is critical.

Tree Based Algorithms are another classic method often used in problems where supervised labels are available. These methods, often referred to as decision trees or forests, build branching data structures where each branch is created based on a decision split in the data.

In the modern work [1] they found that Random Forests (an extremely popular gradient boosted tree algorithm) had 99.1% accuracy when identifying intruders (on the KDD data set with 4 attack variants, see their introduction for more details). One trade off noted in this work is that tree based methods often have slightly higher false negative rates. However, upon further analysis of their results, we found the difference to not be statistically significant with a two sided p test with $\alpha = 0.05$.

Data is an important tool in both supervised and unsupervised machine learning. Intrusion detection is a wide field, with many disparate data sources. A standard data set used is the

Knowledge Discovery and Data Mining (KDD) [11] data set. This data was introduced to help improve intrusion detection and has the following features and 4.8 million data points.

Attributes	Type
Total duration of connections in second	continuous
Total number of bytes from sender to receiver.	continuous
Total number of bytes from receiver to sender	continuous
Total number of wrong fragments	continuous
Total number of urgent packets	continuous
Protocol type	discrete
Type of service	discrete
The status of the connection (normal or error)	discrete
Label (1) if the connection established from to the same host. Otherwise label (0)	discrete

Figure 3: Features included in the KDD data set

3.1.2 Unsupervised

In contrast to supervised methods, that require ground truth labeling of a network trace, unsupervised methods are used when there is a large collection of unlabeled data. For example, if a new security engineer is hired into an organization that was previously lax in their security principles, she could use unsupervised methods as a starting point in the identification of attacks.

Unsupervised methods are primarily referred to as clustering algorithms. These methods group data into a number of clusters based on certain similarity metrics. The simplest of these metrics is distance[1] to the mean value μ for each point in a cluster.

k-Means is an iterative approach to find mean values that minimize the distance from every point to a mean value. Unfortunately, this method suffers from an impactful drawback. The value k must be chosen. This means that the number of clusters is assumed to be known. In [13], they use a fixed value $k = 2$ for attacks and no attacks. This, of course, is a potentially biased choice if there are different types of attacks, and more than one type of verified user on the system. They observe that the detection rate drops from $\sim 80\%$ to $\sim 55\%$.

In contrast, [14] does a small sweep over $k \in \{5, 10, 25\}$. They discover that $k = 10$ is a better choice than the other values since their attack detection rate reaches 100% faster. Their overall accuracy is 72%. This contrast to [13] shows that the value of k is extremely impactful to final model performance, and should be tuned accordingly.

Hierarchical clustering is an unsupervised method that builds tree like structures from the data such that various clusters are separated in the forest. In [12] the authors explore the Dynamically Growing Self-Organizing Tree (DGSOT) method for hierarchical clustering. Interestingly, instead of using this method to simply cluster the data, they use an SVM on different "key nodes" to improve model performance. This is done by classifying additional data points into clusters as the clusters are being built to speed up performance, and improve accuracy. They compare their (DGSOT + SVM) method to a standard hierarchical method Rocchio Bundling + SVM and find the following results across several attack types in the KDD data set.

[1]This distance is usually calculated with cosine distance, but $L2$ distance can be used to similar effect

	RB+SVM	DGSOT + SVM
Normal	98%	95%
DOS	34%	97%
U2R	11%	23%
R2L	27%	43%
Probe	88%	91%

Table 1: Accuracy of clustering methods with an additional classification step. Each row is a type of attack, which is known before hand to be 4 attacks and another cluster for normal network traffic

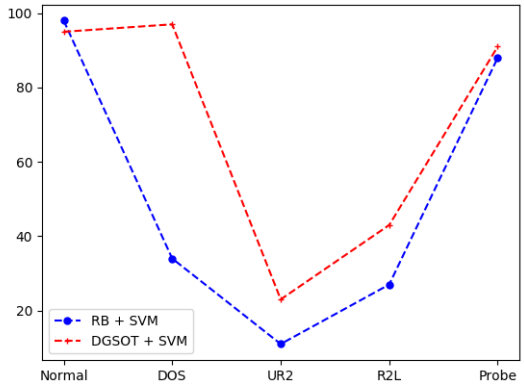

Figure 4: Hierarchical clustering comparison

These results are more clear on a graph, and they show that not only does hierarchical clustering perform well in general, their improved use of the DGSOT method increases accuracy by a healthy margin.

Also, it is important to reiterate that in the case of unsupervised methods, no labeled data is required. Therefore, an unsupervised clustering algorithm could be applied to many current systems (including anomaly, misuse, and hybrid systems) to gain additional insight into traffic. By labeling a small subset from each cluster, security engineers could identify potential threats cheaply and effectively using these methods.

3.2 Statistical, Probabilistic, and Optimization Methods

Naive Bayes (NB) classification is an extremely common probabilistic technique. While it is a standard supervised method, its probabilistic nature is interesting enough to warrant special attention. Naive Bayes is a simple frequency based method commonly used in Spam detection for emails. It counts the occurrence of features in conjunction with certain labels. It makes a fundamental assumption of independence of data points (hence the name naive). In the case of intrusion detection, this implies that two actions taken are independent. An astute reader will question the validity of this assumption, with good cause. This assumption is definitely untrue as

seen in the simple example.

```
$ cd Documents/
$ ls
$ vim file.txt
```

These three commands, if executed in a different order, can have fundamentally different results. Therefore, we can clearly see that they are not independent. However, despite this obvious draw back in underlying assumption, Naive Bayes is a useful baseline. In [1], they find that NB achieves accuracy of 91.23% on the KDD data set. Similarly, [16] found that Naive Bayes was only marginally worse in accuracy than far more complex methods with an astonishingly low false positive rate of 1.3%. The success of this method is astonishing [15] given its unrealistic assumptions. It also is a potentially low hanging fruit that is easily implemented alongside current architectures.

Statistical Methods are an additional way to gain insight into intruder data. These methods are slightly more mathematically involved than other machine learning methods. For example, in [25], the authors compare and contrast Hotelling's T^2 test, Chi-Square Multivariate test, and a Markov chain test. For these methods, a few important pieces of notation are introduced.

$$X = \text{Multivariate Data Matrix} \tag{2}$$

$$\bar{X} = \text{Multivariate Sample Mean} \tag{3}$$

$$S = \text{Sample Covariance Matrix} \tag{4}$$

$$S = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \hat{X})(X_i - \hat{X})^T \tag{5}$$

Then, for the T^2 test, we observe the value of this metric.

$$T^2 = (X - \hat{X})^T S^{-1}(X - \hat{X}) \tag{6}$$

If T^2 is large, than the data deviates from the previously observed values. This clever number can be used to greatly improve anomaly detection rates with accuracy around 68%[2].

The Chi-Square test, on the other hand, is performed by observing changes in X^2 as given.

$$X^2 = \sum_{i=1}^{P}\frac{(X_i - \hat{X}_i)^2}{\hat{X}_i} \tag{7}$$

Importantly, this does not take variance or correlation into account, but merely observes shifts from the mean value. [25] found it to perform slightly worse than the T^2 test. However, in interesting contrast to these results [26] found that as the size of the data set grows, the Chi-Square test out performs the Hotelling's T^2 test. This is due, in large part, to the numerical instability

[2]Note: This is on a different data set than our other methods, and so results can only be compared within the statistical methods section, and not across sections.

of the inverse matrix calculation in equation (6). The results are surprising with the Chi-Square method achieving a 20% better detection rate than T^2 in some cases[3].

Finally, the Markov Chain performed best out of these statistical tests. It takes into account the ordering probability of events in a single network trace. By taking this probability into account, the Markov Chain can output a likelihood that the series of events led to an attack or if they were normal network traffic. This method outperformed the other two statistical tests by a slight margin. This is interesting because it conforms with the intuition formed in our discussion of Naive Bayes. The order of commands executed is important in determining if a network trace is malicious.

3.3 Neural Methods

The adoption of deep learning methods in intrusion detection is disappointing. The author's research is primarily focused on these neural networks and their training schemes. Interestingly, in the intrusion detection community, almost none of the popular tools of the trade are being used. In [6] they train a 1-layer fully-connected neural network which is absolutely minuscule compared to the simplest networks used in the field today. Surprisingly, this group achieves 100% accuracy in detecting intrusion on their data set (also the KDD data set). However, their methodology is seriously flawed. While they follow standard practice and split the data into a training set and a test set, they train their neural network on both the train and test set before the evaluation set. This means they have overfit to their data and essentially memorized the labels. Another indicator that this is the case is they found all methods (except an untrained distance based approach) had 100% accuracy. This result is disheartening, but potentially indicative of a disconnect between the fields of machine learning and intrusion detection.

Additionally, [1] also trained a neural network model called the Multi-Layer Perceptron (MLP). This is a common name given to the same 1-layer fully-connected network seen previously. In this work, the authors follow proper training procedure and achieve modestly successful results. They found the neural network achieved 97.8% accuracy in detecting intrusions on the KDD data set. This is unsurprising given the lack of model sophistication. Interestingly, they trained this model on a single CPU and reported it took approximated 176 minutes to converge. This is orders of magnitude slower than accelerated GPU training. They did not report convergence criteria, it is possible that neural networks could perform even better if properly constructed and trained using modern methods. That appears, however, to be an unexplored avenue of inquiry.

4 Conclusion

For intrusion detection, many of the canonical algorithms have been thoroughly explored with impressive results. In recent years, deep learning has monopolized the machine learning community. There have been small attempts at studying deep learning's efficacy for intrusion detection, but the methods used are exceedingly simple. To study deep learning in the future, which has been shown to be more effective than traditional machine learning for many tasks, one needs to collect more data. As such, in order to improve state of the art machine learning for intrusion detection, there needs to be a concerted effort to collect more data. One hypothesis as to why attention to this field seems to have slowed over recent years is due to a lack of difficult and sufficient benchmark

[3]Their results lead to the conclusion that in data streams over 1MM actions should be analyzed with the Chi-Square test

data sets. However, in general machine learning methods perform well, and often out perform rules based systems.

References

[1] Mohammad Almseidin, Maen Alzubi, Szilveszter Kovacs, and Mouhammd Alkasassbeh. Evaluation of machine learning algorithms for intrusion detection system. In *2017 IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY)*, pages 000277–000282. IEEE, 2017.

[2] Amazon. Guardduty intelligent threat detection aws, 2018. URL `https://aws.amazon.com/guardduty/`.

[3] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

[4] Yuehui Chen, Ajith Abraham, and Bo Yang. Hybrid flexible neural-tree-based intrusion detection systems. *International journal of intelligent systems*, 22(4):337–352, 2007.

[5] CrowdStrike. Crowdstrike introduces enhanced endpoint machine learning capabilities and advanced endpoint protection modules., 2017. URL `https://goo.gl/wVh3s9`.

[6] Amin Dastanpour, Suhaimi Ibrahim, Reza Mashinchi, and Ali Selamat. Comparison of genetic algorithm optimization on artificial neural network and support vector machine in intrusion detection system. In *2014 IEEE Conference on Open Systems (ICOS)*, pages 72–77. IEEE, 2014.

[7] Endgame. Using deep learning to detect dgas, 2016. URL `https://arxiv.org/pdf/1611.00791.pdf`.

[8] Wei Fan, Matthew Miller, Sal Stolfo, Wenke Lee, and Phil Chan. Using artificial anomalies to detect unknown and known network intrusions. *Knowledge and Information Systems*, 6(5): 507–527, 2004.

[9] Anup K Ghosh, Aaron Schwartzbard, and Michael Schatz. Learning program behavior profiles for intrusion detection. In *Workshop on Intrusion Detection and Network Monitoring*, volume 51462, pages 1–13, 1999.

[10] IBM. Machine learning analytics app, 2016. URL `https://goo.gl/DCFCBN`.

[11] KDD. Knowledge discover and data mining challenge. 1999. URL `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`.

[12] Latifur Khan, Mamoun Awad, and Bhavani Thuraisingham. A new intrusion detection system using support vector machines and hierarchical clustering. *The VLDB journal*, 16(4):507–521, 2007.

[13] Pavel Laskov, Patrick Düssel, Christin Schäfer, and Konrad Rieck. Learning intrusion detection: supervised or unsupervised? In *International Conference on Image Analysis and Processing*, pages 50–57. Springer, 2005.

[14] Yihua Liao and V Rao Vemuri. Using text categorization techniques for intrusion detection. In *USENIX Security Symposium*, volume 12, pages 51–59, 2002.

[15] Liu Liu, Olivier De Vel, Qing-Long Han, Jun Zhang, and Yang Xiang. Detecting and preventing cyber insider threats: a survey. *IEEE Communications Surveys & Tutorials*, 20(2):1397–1417, 2018.

[16] Roy A Maxion and Tahlia N Townsend. Masquerade detection using truncated command lines. In *Proceedings International Conference on Dependable Systems and Networks*, pages 219–228. IEEE, 2002.

[17] Microsoft. Machine learning in azure security center., 2016. URL `https://azure.microsoft.com/en-us/blog/machine-learning-in-azure-security-center/`.

[18] RSA. Netwitness ueba, 2018. URL `https://www.rsa.com/en-us/products/threat-detectionresponse/ueba`.

[19] RSA. Netwitness, 2018. URL `https://www.rsa.com/en-us/products/threat-detectionresponse/`.

[20] Malek Ben Salem, Shlomo Hershkop, and Salvatore J Stolfo. A survey of insider attack detection research. In *Insider Attack and Cyber Security*, pages 69–90. Springer, 2008.

[21] Robin Sommer and Vern Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *2010 IEEE symposium on security and privacy*, pages 305–316. IEEE, 2010.

[22] Splunk. Siem - security information and event management, 2018. URL `https://goo.gl/Ljtc6t`.

[23] Chih-Fong Tsai, Yu-Feng Hsu, Chia-Ying Lin, and Wei-Yang Lin. Intrusion detection by machine learning: A review. *expert systems with applications*, 36(10):11994–12000, 2009.

[24] Christina Warrender, Stephanie Forrest, and Barak Pearlmutter. Detecting intrusions using system calls: Alternative data models. In *Proceedings of the 1999 IEEE symposium on security and privacy (Cat. No. 99CB36344)*, pages 133–145. IEEE, 1999.

[25] Nong Ye, Xiangyang Li, Qiang Chen, Syed Masum Emran, and Mingming Xu. Probabilistic techniques for intrusion detection based on computer audit data. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 31(4):266–274, 2001.

[26] Nong Ye, Syed Masum Emran, Qiang Chen, and Sean Vilbert. Multivariate statistical analysis of audit trails for host-based intrusion detection. *IEEE Transactions on computers*, 51(7): 810–820, 2002.