Use case study on machine learning for network anomaly detection

Edin Citaku
Advisor: Simon Bauer
Seminar Future Internet SS2018
Seminar Innovative Internet Technologies and Mobile Communications
Chair of Network Architectures and Services
Departments of Informatics, Technical University of Munich
Email: citaku@in.tum.de

ABSTRACT

Technological advancement has reached the point where the vast majority of businesses rely on computer networks. Thus businesses are exposed to a manifold of network security threats such as network intrusions which can pose a serious threat to them. Network intrusions can be very hard to detect and stop since attackers are steadily developing new ways to intrude into a system. Anomaly detection is a promising approach in solving this problem. Any deviations from the normal state of the system are interpreted as harmful. Thus even novel forms of attacks that alter the system in an unusual way will be detected. This assumption can also backfire and lead to a high rate of false positives since networks often show a lot of variation in their traffic. We will discuss possible ways to circumvent this downside of our approach. Machine learning has shown to be an useful tool for generating an effective notion of what normal or abnormal behaviour in a network system is. Since using machine learning for network anomaly detecting is a hot topic and new research has been accumulating steadily we have decided to review the most common applications of machine learning used for anomaly detection and explain how they were implemented in recent research. In the end we discuss which challenges still remain and propose possible ways to overcome them.

Keywords

machine learning, anomaly detection, intrusion detection

1. INTRODUCTION

Network Intrusion Detection(NID) is an issue that has huge concern in network security. Victims of such intrusions can range from small businesses to military facilities. These attacks can cause tremendous costs and can even be a danger to national security thus new development in this area is of great interest. Network intrusions are unauthorized activities on a computer network that attempt to compromise the integrity, confidentiality or availability of resources on said network. Network Intrusion Detection Systems(NIDS) are programs which actively try to detect those intrusions. Network Intrusion detection can be divided into two categories *misuse detection* and *anomaly detection*. In the former, abnormal behaviour is defined first and then all other behaviour is defined as normal. This approach works well in recognizing already known attack patterns but it is not suitable to detect novel forms of attack. Since NIDS and attackers are in a constant arms race and new forms of attacks are developed steadily, *anomaly detection* shows to be a promising approach. Here normal behaviour is defined first and behaviour deviating from this norm is interpreted as harmful. The primary objective of this technique is to find a suitable way to define what normal behaviour exactly is.

Machine Learning(ML) has helped to advance many different areas of research in the past decades thus using it for anomaly detection does seem like a suitable approach. Like many other areas of research Network Anomaly Detection comes with its unique properties which make it necessary to tweak the methods of ML in such a way that their use for this particular problem becomes practical. ML is a type of algorithm, where the solution for a problem is not explicitly programmed but the algorithm learns its task with the use of training data and progressively improves its performance. In this paper we will look at the different use cases of ML in anomaly detection. For this we will first explain the different types of ML and then for each method of ML we will show a use case, that was developed in current research and explain their unique properties. Later on we will discuss the current state of ML in anomaly detection and explain the different hurdles that still remain and propose ways to overcome them.

The remainder of the paper will be structured as follows. In Section 1 we will explain different background studies relevant to ML and anomaly detection. Later in Section 2 we will talk about work already published related to this paper. Then in Section 3 we will explain different types of ML-techniques that are relevant to network anomaly detection. In Section 4 we analyse implementations of A-NIDS. Finally in section 5 we will discuss the obstacles that still remain in NID using ML and possible ways to overcome them.

2. RELATED WORK

In the recent past numerous researches have published reviews about the current state of ML in anomaly detection. Chih-Fong Tsai et al.[28] wrote such a review where the focussed on the time span between 2000 and 2007. They compared the different studies by their designs, datastes and other experimental setups. A major focus of their paper was to observe how the focus of the research community has evolved in those years. Animesh Patcha et al.[21] worked on a very thorough review of the uses cases of anomaly detec-

doi: 10.2313/NET-2018-11-1_02

tion research where they also looked at ML approaches. A major difference this paper shows compared to the related work listed is the thorough discussion of the challenges of ML in anomaly detection. Major conclusions from this discussion come from the work of Robin Sommer et al. [26]

3. BACKGROUND STUDIES

3.1 Types of classification

When classifying data traffic as either normal or abnormal there are two types learning we can use. These are namely supervised and unsupervised learning. In the former the training data given to the algorithm is already labelled, while in the unsupervised approach it is not labelled. A label in this context is some kind of additional information that indicates to what class a specific dataset belongs. Unsupervised learning is common in the use case of anomaly detection since actual data of an attack is hard to come by.[19]

3.2 Commonly used Datasets

The most commonly used datasets in network anomaly detection research are the DARPA Intrusion Detection Data Sets [17], that were generated by simulating a Airforce network, and the KDD Cup 1999 Data[5], which is data derived from the DARPA Data sets. These datasets are supposed to measure how well an algorithm performs under real life circumstances. Furthermore, it is possible to compare different algorithms if they have been evaluated on the same datasets.In the recent past a lot of criticism about these datasets has been voiced since they are not gathered from a real networks, but from a simulation and also are already considerably old. Despite this criticism, most of current researchers still uses these datasets to evaluate their approaches.

3.3 Measures of Accuracy

When evaluating an algorithm for anomaly detection, the most important metric is its accuracy. We distinguish between two different types of errors, false positives and false negatives. False negatives occur when an intrusion is taking place but is not detected. A false positive on the other hand means, that an anomaly is detected, but no attack is happening. One could assume, that diminishing false negatives is of primary concern to researchers, since an undetected attack is a serious danger for the integrity of a network. While this is certainly the case, it must be noted, that high rates of false positives are a major problem in the field of anomaly detection and diminishing it, is of high priority for researchers.[7]

4. MACHINE LEARNING TECHNIQUES

There are two different approaches on using ML for network anomaly detection. With single classifiers only one kind of ML is, while for hybrid classifiers multiple tools of ML are used in conjunction.

4.1 Single classifiers

We will take a further look at the following ML approaches for network anomaly detection:

- Decision Tree Learning

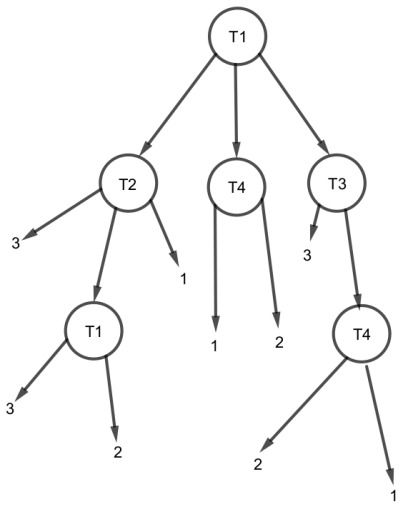

Figure 1: A simple decision tree[13]

- Bayesian Networks
- Genetic Algorithms
- K-nearest neighbor
- Support Vector Machines
- Artifical Neural Networks

4.1.1 Decision Tree Learning

A decision tree is a tool commonly used for classification, where a tree-like graph models different decisions and their consequences. In this graph the attributes of the target are assigned to each node, while the leaf represents a possible classification of the object. In anomaly detection each leaf would be either labelled as ”normal” or as ”abnormal”, although it would be possible to add classes, that specify the type of anomaly. The edges in the graph correspond to different attribute values. To classify an object, you simply traverse the tree, picking the edge, that corresponds to its attribute value. The leaf you reach in the end decides the class your object belongs to. An example for a decision tree can be found in Figure 1. The decision tree is built according to the training data, where different algorithms are possible to use, the so called ID3-algorithm being the most commonly used. Joong-Hee Lee [13] built a model based on decision trees, and showed their performance on the KDD 99 Dataset. The data from the KDD 99 Dataset was first preprocessed, by selecting specific features, that suitably characterize the data. The researchers decided on selecting features, that are already selected in other IDS systems like snort [1], to detect intrusions. To built the tree the researches choose the commonly used ID3 algorithm.

4.1.2 Bayesian Networks

A Bayesian Network[9] is a graphical model representing probabilistic relationships among variables via directed acyclic graphs (DAG). In the DAG the vertices represent events and the edges represent relationships between these events. A numerical component is added with links in the DAG, that represent conditional probabilities of each node in context

of its neighbors. Bayesian networks both have probabilistic as well as causal components making them suitable for classification, even if data is missing.
Figure 2 shows such a bayes network. Rain has an influence on whether the sprinkler is activated. Both the rain and the sprinkler influence whether the grass is wet.
To build a suitable Bayesian network, prior knowledge about the dependencies in the problem are required. It has to be noted, that the prior knowledge the researcher has to bring can show to be a downside since wrong assumptions can decrease the performance and accuracy of the system. A major upside of Bayesian Networks is that they show significantly less computational time in construction and classification compared to other methods of ML. Nahla Ben Amor et al.[3] compared Naive Bayesian Networks to decision trees. They showed, that both had comparable accuracy, while the Bayesian network was computationally much cheaper than the decision tree. Valdes et al. proposed[2] an NADS using naive Bayesian networks. Their model, that is implemented in EMERALD[22] has the capability to detect distributed attacks, where individual attacks are not suspicious enough to generate an alert. One down side of Bayesian networks is that they still show similiar accuracy to simple threshold based systems while still being computationally much more demanding as pointed out by Kruegel et al. [12]. Another problem Kruegel et. al [12] identified is the high percentage of false positives which make the use of traditional Bayesian networks in real life situations impractical. They propose a solution to this problem where they aggregate the outputs of different IDS sensors to produce one single alarm. They assume that one singular sensor is not enough to effectively detect intrusions.

4.1.3 Genetic Algorithms

Genetic algorithms(GA) [30]are a heuristic inspired by the process of natural selection. They are commonly used for optimisation and search problems. GA use techniques inspired by biology, such as mutations, crossover and natural selection. Thus with GA it is possible to derive classification rules or select appropriate features or optimal parameters for the detection process. In recent research Wei Li[15] showed advancements in using GA for intrusion detection. In his research he introduced an algorithm that constructed rules in the form:
if {*condition*} *then*{*act*}
These rules are represented as genes in a chromosome, where up to fifty-seven genes make up a chromosome. In the first generation of the algorithm multiple individuals with a set number of chromosomes and random genes are generated. By only letting individuals with the best fitting rule set, represented by their genes, move forward to the next generation the accuracy of the detection gets gradually better. After each generation the individuals reproduce generating offspring with new sets of genes where the chromosomes of its parents are mixed together. When mixing chromosome sets so called cross-overs can occur with a certain probability where parts of chromosomes can be exchanged between each other. On top of that there is a chance for random mutations introducing singular changes in genes. With all these mechanics it is ensured that the algorithm gradually finds the optimal rule set for the given problem.

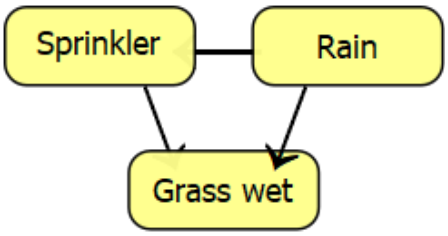

Figure 2: Simple example of a bayesian network[4]

4.1.4 K-nearest neighbor

k-nearest neighbor[16] (k-NN) is an algorithm used for classification and regression. It computes the distance between different points in the input vectors and assigns the point to a cluster of its k-nearest neighbors. k is an important parameter which can have a huge impact on the performance of the algorithm. Typically k is chosen through empirical measures, e.g. different instances of k are tried out, and the instance which yields the best results is chosen. To measure the distance between two data points typically euclidean distance is used, but metrics, such as the Hamming distance can be suitable to. k-NN is called instance based learning or lazy learning which means that new problems are directly compared to the data seen in training without any steps of explicit generalization. Once the algorithm has been trained, the classification of new data is simple: A majority vote is done where the new data point is assigned the class chosen by the majority of its k-nearest neigbors. One use case is the research of Yihua Liao and V. Rao Vemuri [16]. The Algorithm used system calls in the network as input data and generated individual program profiles. The researchers achieved a false positive rate as low as 0.44%, while detecting all intrusions. For training they used the 1998 DARPA BSM audit data. Another interesting approach using k-NN was done by Sutharshan Rajasegarar et al. [23]. They used k-NN to detect anomalies in wireless sensor networks and applied clustering before the actual anomaly detection to minimize computational overhead.

4.1.5 Support Vector Machines

SVM are supervised learning tools used for classification. Objects are represented as hyper dimensional vectors. The SVM maps the training vectors into a hyper dimensional space and then derives an optimal hyper plane dividing the different classes. The decision boundaries are obtained by the support vectors representing the hyper planes rather than the whole training samples. Thus it is robust to outliers. The SVM is mainly designed as a tool for binary classification where training data for both classes is present. Since it is very hard to obtain network data representing an attack different approaches have been made to modify SVMs in such a way that only the data for one class is needed. This type of SVM is called One-class SVM[24]. Such a SVM is able to detect whether an input vector is similar to the dataset it was trained on. Kun-Lun Li et al.[14] developed an approach of ML for anomaly detection where they improved the traditional one-class SVM. In regular one-class SVMs the origin of the hyperplane is defined as the second classification type. The researches modified this method in such a way, that also points ”close enough” to the origin, are detected as anomalies, as seen in Figure 3.

doi: 10.2313/NET-2018-11-1_02

4.1.6 *Artificial Neural Networks*

Artificial Neural networks(ANN) are information processing units inspired by biological neural networks in animal brains. These ANN constitute of connected nodes called artificial neurons each having the ability to process an input signal and then send it to other neurons connected to it. Multilayer Perceptron(MLP) is a architecture of ANN where the neurons are ordered in layers. It consists of input nodes, called sensory nodes more hidden computation nodes and an output layer of computation nodes. Each connection between the nodes is associated to a scalar weight which is initially assigned at random. During the training stage those scalar weights are adjusted usually with the so called back propagation algorithm. An example for an MLP can be seen in Figure 4. Another common architecture of ANN is the so called self-organizing feature map(SOFM)[11]. In this ANN two dimensional discretized representations of the input space of the training data, called *maps*, are generated. Chandrika Palagiri[20] proposed an NIDS using artificial neural networks. They both implemented MLP and SOFM. The results were as expected all anomalies were being detected with the main concern of a high false positive rate. 76% of detected anomalies were false positives. The researcher suggested, that the high rate of false positives stems from the lack of training data for each individual type of attack. The classification times of the program were fast making them suitable for real time classification.

4.2 Hybrid Classifiers

An approach to network anomaly detection gaining rapid popularity is the so called hybrid classification where multible tools of ML are used in different steps of the algorithm. Typically raw data is being preprocessed to some kind of intermediate result. Those results will be taken as input for the second ML tool to produce the final classification. It is possible for example to simply use the step of preprocessing to eliminate unrepresentative training data to then do the actual learning in the second step. You can also use hybrid classification to integrate two different techniques in which the first one aims to optimize the learning performance(e.g. parameter tuning) of the second model of prediction.[29]

4.2.1 *Hybrid classification with enhanced SVM*

One example of such a hybrid classifier is found in the work of Taeshik Shon et al. [25]. The researchers first use a SOFM to create a profile of normal traffic packets for the

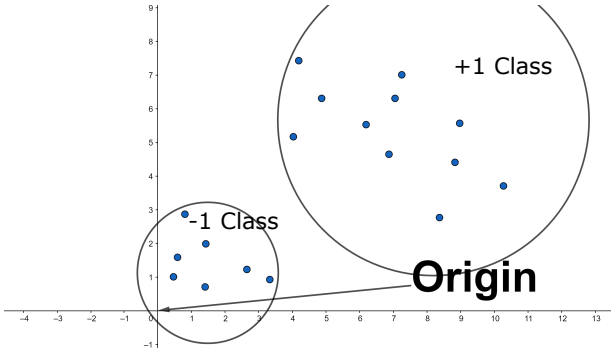

Figure 3: Improved one-class SVM[14]

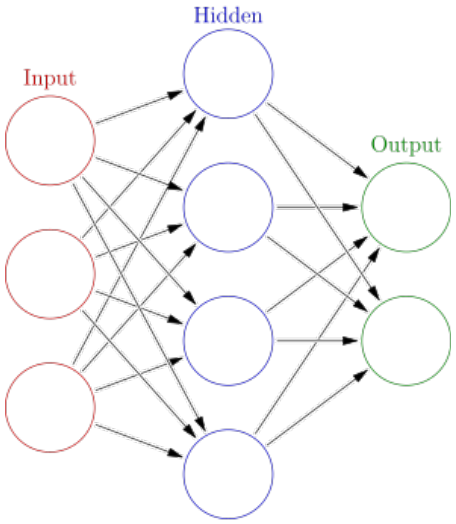

Figure 4: Example of an ANN[8]

SVM. After using a packet filtering scheme to reject incomplete network traffic they apply feature selection using GA on this data to extract optimized data from the raw internet traffic. Once all of this preprocessing is done they use an enhanced SVM to classify the data. The enhanced SVM is a derivative of the traditional SVM which is used for unsupervised learning. The researchers deem this enhancement as necessary because in the traditional SVM approach both normal and abnormal data is needed. Since the goal is to detect *novel* attacks such data is not easily available, and the traditional approach must be modified.

4.2.2 *Decision Tree Classifier with GA-based Feature Selection*

Gary Stein et al.[27] proposed a NIDS based on decision tree classification where the data is preprocessed with GA before the decision tree is generated. The preprocessing step has the purpose of feature selection and is based on the wrapper model.[10] Between each generation of the GA the optimal features, represented by individuals with specific genes are selected for the next generation. This selection is done by constructing a decision tree based on the feature selection each individual represents. These decision trees are then evaluated by their performance. The individuals, whose corresponding decision tree yields the best results then reproduce offspring carrying their genes to the next generation. The feature selection is coded into the genes with binary values where 0 means that a feature is not selected and 1 means that the feature is selected. The researchers showed, that this GA based feature selection improved the accuracy of the final decision tree, compared to a decision tree, where no feature selection was done.

5. CHALLENGES OF MACHINE LEARNING IN NIDS

Despite numerous advancements in the past decades ML for NIDS is still purely a topic of interest for researchers. No NIDS using ML is widely used in industrial settings. This hints that there are still challenges to overcome before

doi: 10.2313/NET-2018-11-1_02

Table 1: Types of ML and their characteristics

Type of ML	Generation	Classification
Decision Tree	Treelike Structure, generated on the basis of training data	The tree is traversed, according to attribute values
Bayesian Network	Directed Acyclic Graph built according to the input data and with additional knowledge by researcher	Decision is done according to the graph
Genetic Algorithms	Features of the classification are first picked at random, then they gradually improve with mechanisms inspired by evolutionary biology, like selection, mutations and crossover.	Rules generated by algorithm are followed.
k-Nearest Neighbor	Input vectors are assigned to a cluster of its k-nearest neighbors	Classification is done, by majority vote. A new object belongs to the same class as the majority of its k-nearest neighbors.
Support Vector Machine	Assigns training vectors a to hyperspace, and computes an hyperplane that divides points into two classes	Classification is simply done by checking what side of the hyper plane the observed vector belongs too.
Artificial Neural Networks	Artificial Neurons, inspired by their biological counter parts, are simulated. These neurons adapt to training data .	Neurons communicate their classification decision with output nodes .

widespread use is viable. In the following we want to discuss these hurdles and what steps have to be taken to overcome them.[26]

5.1 Outlier Detection and Closed World assumption

Machine learning tools are best at finding similarities between data sets. They generally do not exceed at finding outliers in a given dataset. Finding outliers is what network anomaly detection at its core is. We try to find patterns in network flow that do not match the normal state of the network. One could argue that NIDS are doing simple classification because they classify their input data into two categories namely "normal" and "abnormal". This comparison still does not hold, since in traditional ML approaches training data of *all* classes in large quantities are required. This requirement is impossible to fulfil for network anomaly detection because we specifically try to find *novel* attacks. In the approach of anomaly detection we deem all abnormal behaviour as potentially harmful. This assumption does not hold true in real life networks. As quoted by Witten et al.[31]:" *The idea of specifying only positive examples and adopting a standing assumption that the rest are negative is called the closed world assumption. . . . [The assumption] is not of much practical use in real- life problems because they rarely involve "closed" worlds in which you can be certain that all cases are covered.*" Specifically this flawed assumption is the reason for the biggest problem in Network anomaly detection: The high rate of false positives. This make them impossible to use in the real world which leads us to the next problem.

5.2 High cost of errors

Generally an error in intrusion detection is associated with a much higher cost compared to other applications of ML. Thus even if the anomaly detection has the same accuracy, or even better than applications of ML in other domains, it is not necessary that such a method is of any real world value. A false positive, meaning detecting an anomaly that is not there costs extensive time of analysis just to determine it was innocent behaviour after all. Even a small rate of false positives can render a NIDS unusable.[6]. High false positive rates are a major problem of anomaly detection. False negatives on the other hand, have potential of causing serious damage: Even a single compromised system can seriously undermine the integrity of the IT structure. Thus researches have to juggle between false positives and false negatives. To reduce the amount of false positives the algorithm at hand must become *more* sensitive to abnormal behaviour thus raising the rate of false positive at the same time. According to Robin Sommer et al.[26] the number one priority should be to reduce the rate of false positives since they are a big reason why business are not willing to use ML driven anomaly detection. A possible way to reduce the rate of false positives as suggest by Robin Sommer et al.[26] is to reduce the scope of the algorithm so that the algorithm only detects a specific kind of anomaly instead of anomalies in general. This approach also circumvents the problem of juggling between false positives and false negatives. By reducing the scope the relations between the data get much tighter making it easier to differentiate between normal and abnormal behaviour.

5.3 Diversity of Network Traffic

Another problem is that network traffic is more diverse than one would intuitively expect. The networks basic characteristics like bandwidth or duration of connections can vary greatly throughout the day. A solution to this problem is to simply apply aggregation to the input data. This makes it easier for the tool at work to find a notion of normal behaviour and does not waste any valuable information since network data is most of the time noisy anyway. For example in a business where bandwidth usage around the time of lunch is heavily reduced from other times in the day a spike in network flow could be of concern and should be detected as an anomaly. But in a lot of other cases, such anomalies are not a concern of network security. This Network diversity in fact makes it really hard for systems to find a suitable notion of normal.

doi: 10.2313/NET-2018-11-1_02

5.4 High computational demand

A major issue of ML tools is the high computational demand they take while not providing any major performance improvements compared to much simpler methods. In fact in a lot of cases simple threshold based systems provide solutions with comparable detection accuracy compared to their ML counterparts. This can mean higher cost for a business since more computational time is needed to set up the system. Not only is setting up the system by applying the training data computationally demanding also the actual classification can take a considerable amount of time. Time that could be crucial in stopping an attack. On top of that in a threshold based systems detected anomalies are easy to understand and deciding what further actions have to be taken is simple. On the other hand with ML based systems the sheer complexity makes understanding why an anomaly was detected very hard and thus makes it harder to react accordingly. Thus the accuracy of ML tools has to increase in order to justify their high complexity and computational demand.

5.5 Difficulties of Evaluation

One big concern in anomaly detection is that the datasets most commonly used for evaluation are not representative to current real world networks. The two most common used datasets are the DARPA/Lincoln Lab packet traces and the KDD Cup dataset derived from them which are both publicly available. Both datasets are almost two decades old making studies of current forms of attack impossible. Furthermore the DARPA dataset was artificially generated, by simulating an Air Force network. Whether one can derive conclusions for real world networks by researching synthetic datasets is highly doubtful. The KDD Cup datasets which are directly derived from the DARPA datasets show the same problems. On top of that, both datasets show simulation artefacts.[18]. These artefacts can heavily bias the evaluation, since it is possible that the ML tool at hand learns through these artefacts. In that case any deduction to the real world is impossible. A commonly used example that illustrates the issue pretty well is a story where US researchers were trying to detect tanks in images using neural networks.[32] The mistake they made was that all the pictures of tanks were made on a cloudy day, while the pictures of images without tanks were made on a sunny day. What happened was that the neural network was trained on detecting the weather instead of detecting the tanks. In our case the equivalent would be that our algorithm is not trained on detecting anomalies but on detecting artefacts in the dataset. Thus it is necessary to find another source of data to evaluate IDS. Some researchers choose to use data gathered from their own networks. This solution is not satisfactory since big networks, commonly used by businesses, are fundamentally different than smaller scale networks in research facilities. [26] A way to gather suitable data for evaluation would be to directly gather it from businesses. The problem is that most businesses are not willing to share data regarding their networks, since a lot of sensitive information can be deduced from it which in turn can hurt these businesses. Thus the data has to be anonymized, preferably in such a way that no information relevant to anomaly detection is lost.

6. CONCLUSION

Network anomaly detection is of high importance to businesses since a network intrusion can have catastrophic consequences. Machine Learning shows to be a promising approach because novel forms of attack can be effectively detected. Through Machine Learning the system learns what normal and abnormal behaviour in the network means. Since most attacks alter the system in an unusual way they will be detected. This paper briefly describes the foundations of different machine learning approaches, such as support vector machines or artificial neural networks and explains how they can be used to detect network anomalies. To do this we present recent research for each approach and explain the most important details of it. Later we discuss the hurdles that still remain in network anomaly detection and how to overcome them. It is clear that the whole research community will benefit from more reliable datasets. More work has to be done in this area. With new approaches of ML being developed steadily and computers getting faster there is still hope that Machine Learning in anomaly detection will be widely used in the near future.

7. REFERENCES

[1] https://www.snort.org/[Online, accessed 01-April-2018].

[2] V. A. and S. K. Adaptive, model-based monitoring for cyber attack detection. In *Recent Advances in Intrusion Detection. RAID 2000. Lecture Notes in Computer Science*. 2000.

[3] N. B. Amor, S. Benferhat, and Z. Elouedi. Naive bayes vs decision trees in intrusion detection systems. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 420–424. ACM, 2004.

[4] AnAj. Simplebayesnetnodes.svg. https://commons.wikimedia.org/wiki/File:SimpleBayesNetNodes.svg, 2013. [Online, accessed 01-April-2018].

[5] T. U. K. Archive. Kdd cup 1999 data. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, 1999.

[6] S. Axelsson. The base-rate fallacy and its implications for the difficulty of intrusion detection. In *Proceedings of the 6th ACM Conference on Computer and Communications Security*, pages 1–7. ACM, 1999.

[7] D. Colquhoun. The reproducibility of research and the misinterpretation of p-values. *Royal Society Open Science*, 4(12), 2017.

[8] Glossar.ca. Colored neural network.svg. https://commons.wikimedia.org/wiki/File:Colored_neural_network.sv, 2013. [Online, accessed 01-April-2018].

[9] F. V. Jensen. *Bayesian Networks and Decision Graphs*. 2001.

[10] R. Kohavi and G. H. John. The wrapper approach. In *Feature extraction, construction and selection*, pages 33–50. Springer, 1998.

[11] T. Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

[12] C. Kruegel, D. Mutz, W. Robertson, and F.Valeur. Bayesian event classification for intrusion detection. In *Computer Security Applications Conference, 2003. Proceedings. 19th Annual.*

[13] J.-H. Lee, J.-H. Lee, J.-H. R. Seon-Gyoung Sohn, and T.-M. Chung. Effective value of decision tree with kdd 99 intrusion detection datasets for intrusion detection system. In *10th International Conference on Advanced Communication Technology*, 2008.

[14] K.-L. Li, H.-K. Huang, S.-F. Tian, and Xu. Improving one-class svm for anomaly detection. In *Proceedings of the 2003 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.03EX693)*, volume 5, pages 3077–3081 Vol.5, Nov 2003.

[15] W. Li. Using genetic algorithm for network intrusion detection. *Proceedings of the United States Department of Energy Cyber Security Group*, 1:1–8, 2004.

[16] Y. Liao and V. Vemuri. Use of k-nearest neighbor classifier for intrusion detection11an earlier version of this paper is to appear in the proceedings of the 11th usenix security symposium, san francisco, ca, august 2002. *Computers & Security*, 21(5):439 – 448, 2002.

[17] R. Lippmann, R. K. Cunningham, D. J. Fried, I. Graf, K. R. Kendall, S. E. Webster, and M. A. Zissman. Results of the 1998 darpa offline intrusion detection evaluation. *Proc. Recent Advances in Intrusion Detection*, 1999.

[18] M. V. Mahoney and P. K. Chan. An analysis of the 1999 darpa/lincoln laboratory evaluation data for network anomaly detection. In *International Workshop on Recent Advances in Intrusion Detection*, pages 220–237. Springer, 2003.

[19] N. M. Nasrabadi. Pattern recognition and machine learning. *Journal of electronic imaging*, 16(4):049901, 2007.

[20] C. Palagiri. Network-based intrusion detection using neural networks. *Department of Computer Science Rensselaer Polytechnic Institute Troy, New York*, pages 12180–3590, 2002.

[21] A. Patcha and J.-M. Park. An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Computer networks*, 51(12):3448–3470, 2007.

[22] P. A. Porras and P. G. Neumann. Emerald: Event monitoring enabling responses to anomalous live disturbances. In *20th NIST-NCSC National Information Systems Security Conference (1997)*.

[23] S. Rajasegarar, C. Leckie, M. Palaniswami, and J. Bezdek. Distributed anomaly detection in wireless sensor networks. pages 1 – 5, 11 2006.

[24] B. Schoelkop, J. C. Platt, J. Shawe-Taylor, and A. J. Smola. Estimating the support of a high-dimensional distribution. pages 1443–1471, 2001.

[25] T. Shon and J. Moon. A hybrid machine learning approach to network anomaly detection. *Information Sciences*, 177(18):3799–3821, 2007.

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

[27] G. Stein, B. Chen, A. S. Wu, and K. A. Hua. Decision tree classifier for network intrusion detection with ga-based feature selection. In *Proceedings of the 43rd annual Southeast regional conference-Volume 2*, pages 136–141. ACM, 2005.

[28] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, and W.-Y. Lin. Intrusion detection by machine learning: A review. *Expert Systems with Applications*, 36(10):11994–12000, 2009.

[29] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, and W.-Y. Lin. Intrusion detection by machine learning: A review. *Expert Systems with Applications*, 36(10):11994–12000, 2009.

[30] D. Whitley. A genetic algorithm tutorial. *Statistics and computing*, 4(2):65–85, 1994.

[31] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2016.

[32] E. Yudkowsky. Artificial intelligence as a positive and negative factor in global risk. *Global catastrophic risks*, 1(303):184, 2008.