Using Artificial Intelligence in Intrusion Detection Systems

Matti Manninen
Helsinki University of Technology
mimannin@niksula.hut.fi

Abstract

Artificial Intelligence could make the use of Intrusion Detection Systems a lot easier than it is today. They could learn the preferences of the security officers and show the kind of alerts first that the officer has previously been most interested. As always, the hardest thing with learning AIs, is to make them learn the right things. AIs could learn the same things as a rule-based system by watching a security officer work. AIs could also link together events that, by themselves, are insignificant but when combined may indicate that an attack is underway. In this article I'll compare AI-based solutions to traditional IDS solutions, and analyze how the AIs could be taught.

KEYWORDS: Intrusion detection, artificial intelligence, neural networks, computer security

1 Introduction

This article focuses on finding out how to make an IDS environment learn the preferences and work practices of a security officer, and how to make it more usable by showing the most often viewed anomalies first. The goal is quite simple, but programming such a system that makes it happen is not simple in any way. Making the learning process invisible and continuous further adds to the difficulty of this task.

The most difficult task for the AI would be figuring out the connections between different events. There are several ways to achieve this, neural networks being the most prominent.

1.1 Motivation behind Using IDSs

Networks and the computers in them get more complex every day. This means that there are also more and more services available for malicious exploitation. New vulnerabilities are found from common programs daily and even one vulnerability in a single computer might compromise the network of an entire company.

There are two parallel ways to address this threat. The first way is to ensure that a computer doesn't have any known security vulnerabilities, before allowing it to the network it has access rights. The other way, that I will discuss in this article, is an Intrusion Detection System. IDSs concentrate on detecting malicious network traffic, such as packets that would exploit a known security vulnerability.

1.2 Main Characteristics of Different IDSs

There are two main types of IDSs: online IDSs and offline IDSs. The offline IDSs analyze connection data from the logs after the connections have happened. This can happen just after the connection has started, so that further access could be denied if the connection was classified as an intrusion. The online IDSs, if possible, analyze data before connection is allowed. They also monitor the connections so that a connection can be dropped if it starts to seem like an intrusion.

The difference between these could be very minimal. The limiting factor is the time it takes to analyze a connection. Offline IDSs have less time constraints and should therefore be more accurate. Also, as they analyze the situations afterwards, they have more information available then, than when the last connection was just beginning. Also online IDSs may benefit from analyzing the logs from previous connections, in order to enhance accuracy.

1.3 Scope of Paper

The article is divided into three separate domains. Even though they are partly related, each has their own specialities to consider. These parts are

1. Usability of the learning process. Introduction of noise into data in different cases.
2. Ways to detect intrusion based on learned examples. Responses to noise in the data.
3. Showing the events to the security officer in the correct order. Minimizing false alarms.

2 Related Work

James Cannady [3] has written a highly referenced article about intrusion detection using neural networks. In the article, he studies in detail the advantages and disadvantages of neural networks for this application. The conclusion of the article was that neural networks are very suitable for IDS. However, the article shows that training the neural network is not trivial and may in fact require substantial

effort.

Mehdi Moradi and Mohammad Zulkernine [6] have further researched the neural network approach to intrusion detection. Their focus is on off-line analysis. They are particulary interested in classifying the intrusions, instead of just detecting them. According to their article, a properly trained neural network can also work as an online IDS for the types of attacks it has been trained for.

Magnus Almgren and Erland Jonsson [1] have studied how AI-based IDSs can learn the types of alerts it should show the security officer. The focus of the article is on AIs that learn from normal use of the system; in other words, they don't require any additional user input for the learning process.

Similarly as Almgren and Jonsson, Stefano Zanero and Sergio M. Savaresi [9] have studied invisibly learning AIs, but concentrated on different clustering methods, such as self-organizing maps. The article is fairly vague on how and when those methods are at their best, but it does prove that such clustering methods are also suitable for an IDS.

Jeremy Frank [4] compares different AI methods for IDSs. Even though the study is from 1994, the results still apply to current systems. The most important finding in the article is that feature selection can be used to improve performance dramatically.

3 Findings

This section contains all the results of the research that I have done. The section is divided into subsections following loosely the list in section 1.1.

3.1 The Learning Process

The learning process is the interaction period that the AI needs in order to function properly. In several learning methods, the process is continuous so that the AI would constantly adapt to new threats. There are several possible methods to teach an AI [1]. In this article I will cover the following:

- Following the user's clicks
- Explaining the clicks to AI
- Presenting a reason why AI has classified an event and asking the user if that is correct
- Asking the user questions about the ranking

An IDS AI might consist of several subsystems, possibly determined by the user interface. At the very least, it might be a good idea to have one AI for detecting and classifying events in a network and another AI for determining which events the security officer wants to know about.

Several AIs of course means that all of them require teaching before use. On the other hand, several AIs make it possible to have a common classifying AI and a private user interface AI for each security officer. For the classifying AI, the training data could be generated based on a rule-based expert system as in the experiment by Cannady [3]. Another way would be to record the usual traffic in the network and go through it to classify everything manually. This would probably work especially well for a system that tries to detect normal connections rather than unusual ones.

The user interface AI could use some standard settings as a starting point. These standard settings should be quite light and have only few samples so they would quickly change towards that what the security officer wants. For example, the user interface AI could determine the order in which to show the events and which of the connected events, that by themselves have little importance, should be shown.

3.1.1 Usability of the Learning Process

The usability of the learning process is very important. If the process requires considerable work hours, it will soon become very expensive.

From the security officer's point of view, it is easiest if the AI just learns his preferences when he uses the system normally. The easiest way to implement this is to monitor the clicks that the user makes [1]. This kind of click monitoring approach assumes that the event, that the security officer clicks, contains something interesting and that it is important to show similar events the next time as well.

This method is very easy for the user; it requires no additional input. However, one severe problem is that similar types of events tend to enhance themselves until the system overwhelms the security officer with that type of event. Collecting additional negative feedback from the user might help against this [1]. Another problem, that isn't mentioned in the study by Almgren and Jonsson, is the noise caused by this approach. Just following the clicks might easily result in some unimportant events to be listed as important because of misclicking. On the other hand, it could be that some common events have to be checked in response to a certain type of other event. If the AI cannot properly figure out the connection between the main event and these common events, it might easily think that the common events are important since they gain significantly more clicks than the main events because there might be several such common events, which are related to one main event.

Another way of interaction is to directly teach the system in some cases. There are many ways to accomplish this. Maybe the easiest for the security officer, is to tell the AI that something that it shows either is or isn't actually important. This would provide the ability for negative feedback that would help in the chain reaction problem where positive

feedback keeps on enhancing itself. Some spam filters use this kind of approach where the system is self-enhancing, but asks the user in case there is reasonable doubt about whether or not the mail is spam. The same could be used in IDSs: events with reasonable detection certainty enhance the rules and uncertain ones are left for the user to decide.

A more detailed way of direct teaching would be to tell the AI why the selected event is interesting [1]. This would help the system in determining what should be stressed with that kind of event.

Another detailed way of direct teaching would be the opposite. The AI tells the user why it thinks some event is important, and the security officer answers whether it was right or not [1].

Luckily, one can use a mixture of these teaching methods. Most of the time, the system should probably use the clicking method, but when needed, other methods could be used as well [1]. The other methods could be needed when there are some false positives or false negatives.

3.1.2 Noise from the Learning Process

The noise produced by the learning process is an important factor in the usefulness of the gathered material. If the process produces a lot of noise the learning period will have to be significantly longer than with a process that produces fewer false samples, in order to gather the same amount of valid data. With pre-made sample material, the noise can be eliminated, but then the learning material cannot take the personal preferences of a security officer into account.

The clicking method for training is especially prone to noise. The idea is that everything that the user clicks would be important in some way. However, in reality there are lots of misclicks and false leads and thus also unimportant events get clicked.

3.2 Detecting the Intrusion

There are various principles on which an IDS can be based upon. There are also several different types of AIs that can be used for each of these principles.

In addition to detecting an intrusion, it would be preferable to also be able to detect the type of the attack. This would make it possible to suggest proper actions that should be taken [9].

3.2.1 Detection Principles, Different Aproaches to Detect an Intrusion

The two main ideas are either to detect forbidden and suspicious activity or to detect normal activity and alarm about anything that isn't normal.

Detecting forbidden activity requires data, generated or gathered, about previous similar events that have been classified to be connected to intrusions.

The other option is to teach the system what is normal. After that, the system can then try to distinguish unusual events from the usual ones. This method sounds good in theory, but the reality is quite different. Such systems often produce a lot of false alarms and are very poor at identifying what actually seems to be wrong [9]. In reality, the differences between legal and illegal events are very small. Because of this, an AI that tries to keep false negatives at a minimum would have to be tuned to detect even the slightest of variances from the learned data. In turn, this would produce a lot of false positives because there are bound to be differences anyway.

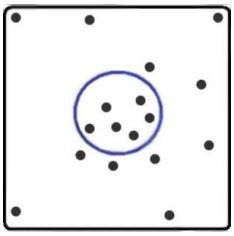

Figure 1: AI that focuses on detecting allowed activity

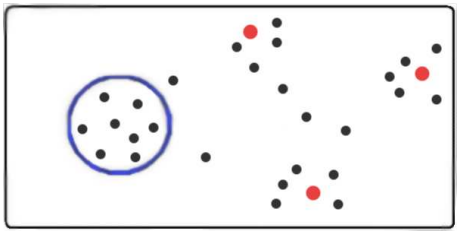

Figure 2: AI that focuses on detecting denied activity

In one sense this kind of approach could be thought of as a circle, everything that's inside is normal and everything that's outside is forbidden, as pictured in Fig. 1. The circle must be quite small, ie. close to the center, so that forbidden events, that are very near to normal, would be correctly identified. The approach where forbidden activity is detected is a different case. There is also the circle, as the training data also contains data about what's normal, but in addition to the circle, there are also other dots, that represent the different kind of forbidden events, as pictured in Fig. 2. Then if we have an event that is quite similar to normal activity but also a bit similar to some forbidden event, it would be drawn towards both the normal and the forbidden centers. Therefore it would in any case end up quite far from the normal center. This explains why the detection sensitivity in systems that detect forbidden events can be kept at a lower level, while still maintaining the same detection capability.

3.2.2 Traditional IDSs

A common factor for traditional IDSs is that they don't use any kind of AIs for intrusion detection. Rule-based expert systems are the most traditional of IDSs. They match events and scenarios to rules that define denied or allowed scenarios. State transition analysis and color-coded petri nets follow a sequence of actions with a graph or a state machine to determine if an attack is detected. Traditional IDSs are mostly used in misuse detection. In the field of anomality detection they are no match for AI-based systems [7].

The biggest difference between AI-based and traditional IDSs is that only AIs can learn new rules on their own. This means that in traditional systems the security officer must insert new rules for each new attack type or each new allowed program. In AI-based systems it is possible to teach the system by examples rather than rules.

3.2.3 Different AI Types

There are several different soft computing techniques and algorithms that can be successfully used to detect intrusions. These techniques include [2]:

- Fuzzy logic
- Probabilistic reasoning
- Neural networks
- Genetic algorithms

Combinations of these can also be used. For example, genetic algorithms can be used to build a neural network and probabilistic reasoning can be built on fuzzy logic. Neural networks are the most common AI type for an IDS [9].

Neural networks are basically sets of individual cells that have weighted connections to other connected cells. The training process of a neural network consists of setting weights for each connection and comparing the output with the desired output. This is iterated until the desired accuracy with a test set of data has been achieved [9].

Genetic algorithms can be used to keep the number of iterations as small as possible. Genetic algorithms first randomize values for each set. Then they select a few of the best sets and mix and differentiate values from each of these. This is continued until the desired result is reached or the maximum number of generations has passed [2].

Gowadia, Farkas and Valtorta have studied a probabilistic reasoning based AI in IDS use in their article [8]. They used Bayesian networks to assess the probability of an intrusion. They had multiagent system in which agents in separate computers could communicate and tell each other threat estimates from each one's point of view. The agents concentrated on detecting different intrusions on separate domains. The authors concluded that this kind of approach is feasible. The system offers a possibility to select the percentage that an event must reach in order to be shown. The multiagent approach adds redundancy to the IDS and increases efficiency. Bayesian networks are also widely used in learning spam filters for e-mail.

Because of their popularity in IDS use, I will mainly focus on neural network based AIs. They have several important advantages over other rule-based systems. They are able to efficiently use incomplete or distorted data and to figure out relations between events, which helps in detecting attacks from multiple sources. They are very fast in classifying the events. They are able to learn and identify new threats that haven't been expressly taught to them. The neural networks return a probability instead of a boolean value, which makes it possible for them to predict probable following events in case there would be an attack going on. In turn, this would make it possible to defend against them in advance, in case that the system is an online IDS [3].

However, the neural networks do have their disadvantages. Training them costs a considerable amount of computing time. With a continuously learning system this might lead to high hardware requirements. However, the biggest problem is the black box nature of neural networks [3]. This means that there are no clear rules on which the results are based; instead the answers just pop out, with no clear explanation about the process or the reasons for the result.

There are several types of neural networks that are suitable for IDSs. Some of these are:

- Self-organizing maps (SOMs)
- Multi layer perceptrons (MLPs)
- Feature selection

Self-organizing maps are a tool for pattern discovery. SOMs work by clustering similar entries into groups in a map. Lichodzijewski et al. studied SOMs with respect to IDS use in their article [5]. They concluded that self-organizing maps can best be used in a context where events do not have timestamps. This was a surprising finding, as intuitively one would easily think that events outside office hours could more easily be attacks and should therefore be reacted to more aggressively. As the study suggest, this is not the case. In reality, maybe the timestamps just cause some clustering of their own, which disturbs other relevant clustering more than it helps it. This result could well apply to all neural network based IDS AIs as they all try to search for similarities between different events. In this case similar time stamps might link unrelated events together which would just confuse the AI.

Moradi and Zulkernine [6] used a multi layer perceptron based neural network with good results. MPLs consist of an input layer, one or several hidden layers and an output layer. Each of these layers has nodes that are connected to upper and lower layers. MLPs can be used to classify different types of attacks rather than just classifying an event as an attack or normal traffic. The results were that MLPs are very efficient in this kind of classifying. However, data gathering for the required features is quite slow, which makes the test

system an offline IDS rather than an online IDS.

Frank [4] studied using feature selection for the neural network values. Feature selection is a method that, in a way, compresses and streamlines the data. It can be used to select the most important characteristics from a set of training data. This decreases the computing time for the actual neural network training and reduces impact of unimportant features that might otherwise disturb the accuracy.

3.3 Effects of Noise

Noise always causes problems, regardless of the used intrusion detection methods. The amount and type of these problems presumably varies from method to method. I would figure that the most common problems are:

- False alarms
- Showing the events to the security officer in a wrong order
- Classifying important events as unimportant

Unfortunately, noise seems to be a little studied subject with regard to the training data of IDSs. Knowledge about effects of noise in general signal processing can help when trying to understand what noise could cause in IDSs.

3.4 Showing the Alarms

One aspect of a good IDS is showing only the important alerts to the security officer. This section focuses on finding out what factors effect in determining what alerts are important and what are not. Also, the effects of showing just the correct alarms, or failing to do so, are described in this section.

3.4.1 Determining the Proper Order of Importance

The correct order of importance helps the security officer to react to the most severe threats first. If the ordering works properly and the security officer also trusts that it works, it will save time as he doesn't have to compare each of the shown threats to determine the order of importance himself. If there are hundreds of alarms or events and the ordering doesn't work properly, the important ones might never be seen, if they wind up at the end of the list. In some systems, new events might flow in constantly, at a rate that would quickly bury away any events, which are listed as unimportant.

3.4.2 False Alarms

No system is perfect and there are bound to be some false alarms in any case. False alarms are often also called false positives. Usually some information that is available to the security officer isn't available for the AI as any variable. In some cases, this extra information could be the only thing that would explain a connection to be legal even though, by all measures, it seems like an attack. False alarms of this type will, and even should, happen, no matter how well you train an AI. It would be far worse not to alarm about something suspicious, than to alarm and let the security officer himself find the information that isn't available in the system.

The false alarms that are caused by day-to-day operation are the ones that matter. If the AI cannot be taught how to safely ignore these, it will cost a considerable amount of the security officer's time, as these alarms just continue to repeat day after day. In addition to costing time, false alarms can easily lead to a situation where the security officer himself just ignores all alarms that look like these. This can easily be exploited by performing an actual attack that would look similar enough to ignore. Even though this approach wouldn't fool the AI and it would alarm, the security officer might just ignore the real attack along with the false alarms.

The other issue with these kinds of false alarms is that the AI learns some wrong things, which incorrectly causes it to ignore the similar real attacks. The reason for these false alarms might well have been a similar real attack in the training data. If the security officer doesn't know this, he might think that the AI has received some noise that causes these false alarms and clear some of the training data in order to get rid of the alarms. After this, the AI wouldn't know about the real attack, and would therefore also classify similar attacks as normal. In other words, this would cause false negatives.

4 Conclusions

AI-based solutions can be used for IDSs. It seems that neural networks are the most popular selection for this kind of AI with a good reason. Wide variery of choices for a neural network type make it possible to select a type that works in a given application. Other AI types have also been proved to be suitable for IDS use. They have been studied less, which could partly explain why they cannot currently fully compete with neural networks.

At this point AIs seem to have the needed accuracy for IDS use. Configuring an AI-based IDS is easier than configuring a traditional IDS. This decreases deployment costs which is an important factor for companies. Because of this, they could more easily test different easy-to-deploy IDSs to see which of them is most the secure and requires the least monitoring in their network.

The most surprising thing in writing this article was the fact that I couldn't find almost any information on the effects of the noise in the learning data to the accuracy of an IDS. To me, this would definately seem like a subject that would be worth studying, especially as invisibly learning, ie. click monitoring, AIs will usually gather up noise.

References

[1] M. Almgren and E. Jonsson. Tuning an ids - learning the security officer's preferences. In *11th Nordic Workshop on Secure IT Systems - Nordsec 06*, pages 43–52, 2006.

[2] P. P. Bonissone. Soft computing: the convergence of emerging reasoning technologies. *Soft Computing—A Fusion of Foundations, Methodologies and Applications*, 1(1):6–18, 1997.

[3] J. Cannady. Artificial neural networks for misuse detection. In *Proceedings of the 1998 National Information Systems Security Conference (NISSC'98) October 5-8 1998. Arlington, VA.*, pages 443–456, 1998.

[4] J. Frank. Artificial intelligence and intrusion detection: Current and future directions. In *Proceedings of the 17th National Computer Security Conference*, Baltimore, MD, 1994.

[5] A. H. M. Lichodzijewski, P.; Nur Zincir-Heywood. Host-based intrusion detection using self-organizing maps. In *Proceedings of the 2002 International Joint Conference on Neural Networks*, 2002.

[6] M. Moradi and M. Zulkernine. A neural network based system for intrusion detection and classification of attacks. In *2004 IEEE International Conference on Advances in Intelligent Systems*.

[7] A. Mounji. *Rule-Based Distributed Intrusion Detection*. PhD thesis, University of Namur, 1997.

[8] V. G. C. F. M. Valtorta. Paid: A probabilistic agent-based intrusion detection system. In *Computers & Security*, pages 529–545, 2005.

[9] S. Zanero and S. M. Savaresi. Unsupervised learning techniques for an intrusion detection system. In *SAC '04: Proceedings of the 2004 ACM symposium on Applied computing*, pages 412–419, New York, NY, USA, 2004. ACM Press.