MEng in Electronic & Computer Engineering

Network threat analysis and prevention

Patrick Roughan

Abstract: Network attacks and infiltrations are a big and growing area of concern worldwide. Resulting in huge cost and reputational damage to governments and companies alike. Even with this the number and scale of these attacks is growing, showing that traditional Network Intrusion detection devices are inadequate. Machine Learning is a technology that learns and subsequently predicts an outcome and adapts without instruction. This review presented a number of machine learning models identified in the literature and analyzed the appropriateness of using these technologies to address security issues. The findings from this literature review will inform a future project which aims to build a Machine Learning Classification Algorithm that is able to detect intrusions in a given network with high confidence.

INTRODUCTION

Security in today's world is more important than ever with the exponential growth in exploits and network intrusions. This has been highlighted with the recent cyber-attack on solarwinds which lead to network intrusion into US Government and multiple businesses. In the McAfee Quarterly threat Report 2020, significant growth in malware detection was reported year on year and in some areas exponential growth [1]. When it comes to the growth of internet of things (IoT), according to the Gemalto: State of IoT Security [2], half (48%) of firms are unable to detect whether IoT devices on their networks have suffered some kind of breach, such as being co-opted into being part of a botnet. Another aspect of security, which is relevant to this project, is what Fireeye M-Thread 2020 report [3] classifies as “dwell time”, the length of time a cyber-attacker compromises your network to detection and eradication. It is reported that the median time is 56 days which is a drop from the previous year of 78 days, but still a significant duration of time for an attacker to go undetected and have free reign on the network [3].

The current solution to the above threat is the Network Intrusion Detect system (NIDs). There are two approaches NIDs use to detect an intrusion. Signature-based which is a traditional approach of when an exploit is discovered it is reverse engineered and a signature of it is created which the NID uses to detect the Intrusion. As mention above, with the current dwell time being 56 days, the signature approach doesn’t deal with zero-day vulnerabilities and the lag time is quite high from the first intrusion until your network is safe. The other approach is anomaly-based. Here the NIDs baselines the traffic in the network to what it considers normal and anything outside of that is seen as an anomaly. This approach in today’s world of fast-paced networking and new protocols leads to a high number of false positives. Increased human intervention and decreased levels of detection accuracy.

Project objective

The proposed project is to build a Machine Learning Classification Algorithm that is able to detect intrusions in a given network with high confidence. Then integrate this Network Intrusion detection algorithm into a Software-Defined Network Controller. In which Intrusions can be automatically detected and rules are written and distributed throughout the network to block the Intrusion.

METHOD

A literature search was undertaken to identify recent publications presenting different applications, methods and technologies available to help address the issues of network intrusion using machine learning technology. The literature search investigates the use of more advanced machine learning technologies and their implemtations which could be used to address current security issues. The literature search seeks to understand these advanced technologies, how they can be implemented and their overall performance and potential advantage.

RESULTS

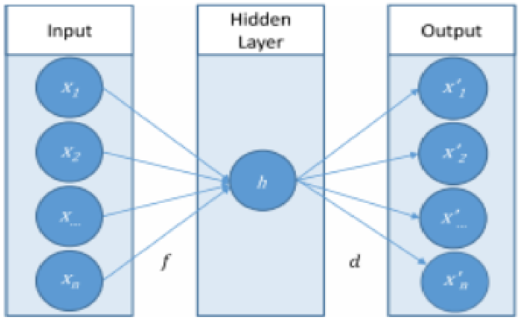
Synopsis of literature

During the initial stages of the literature review, a number of specific technologies were identified; Autoencoders, Transfer learnings, Deep Belief Networks, and the implementation of these technologies. For the purpose of this review and to support the objective of the overall project, the key findings from the identified publications, focusing on these technologies, are presented in the following sections.

Autoencoders

Autoencoders are an unsupervised machine learning approach to feature extraction and dimensionality reduction. It works by first taking the input feature and encoding them into latent representation and then trying to reconstruct the input (decode) by using the best parameters it has learned [Figure 1]. As well as being used for feature extraction and dimensionality reduction they are good for anomaly detection.

Figure 1: Example of a single Autoencoder

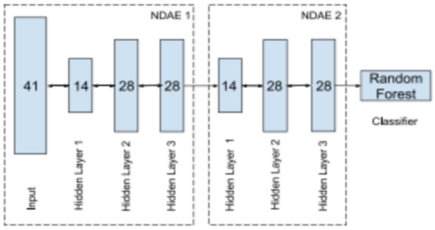


Reference: [4]

Five publications reporting on autoencoders and their implementation were identified from the literature search.

Shone et al. [4] presented a stacked nonsymmetric deep autoencoder (NDAE) for unsupervised learning feature learning. Unlike traditional autoencoders the nonsymmetric approach utilises only the encoder phase. The claim was that they were able to reduce computational time overhead without a significant impact on accuracy and efficiency. By stacking the Autoencoders this allowed the model to learn complex relationships between different features and prioritise the most descriptive features. The final part of the model was to output the learned features from the stack NDAE to a Random forest shallow learning classifier in order to classify network traffic into normal data and known attacks [Figure 2].

Figure 2: Stacked NDAE Classification Model



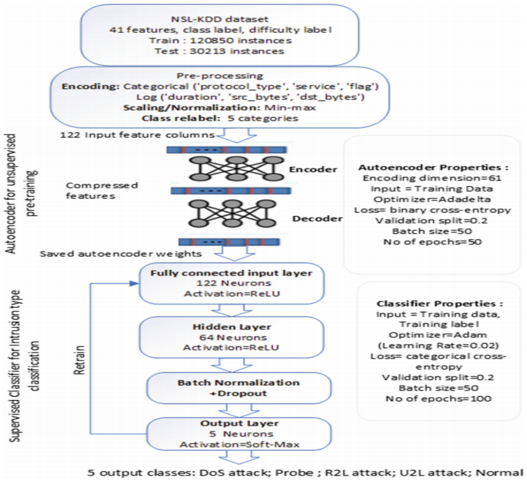
Abbreviation: NDAE, non-symmetric deep autoencoder
Reference: [4]

Azar et al. [5] investigated two areas, malware analysis and network anomaly detection, for the purpose of this review the focus is on network anomaly detection. For this stage, the authors proposed a deep autoencoder with a unique training phase and topology. A minimum number of features were used to reduce the dimensionality and the output latent representations, and reducing the memory footprint making it suitable for realtime and small devices for example in IoT.

The deep autoencoder approach gives better performance and the use of a pre-training and a fine-tuning phase allowing better initial weights to be obtained. Combined with using backpropagation for updating the weights the proposed model provided a discriminative concept space to distinguish between a normal anomalous flow of network data.

Rezvy et al. [6] used another deep autoencoder as a pre-training phase for feature extraction and the output was fed into a deep neural network that acted as a supervised classifier for intrusion detection. The paper focused on low latency and high accuracy (Figure 3).

Figure 3: Workflow and architecture of the proposed autoencoded dense neural network

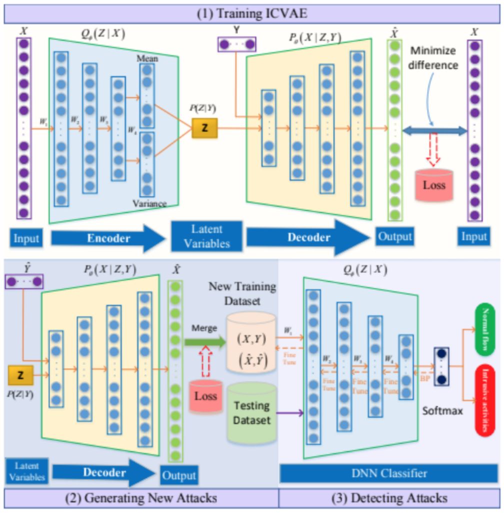

NSL-KDD dataset
41 features, class label, difficulty label
Train : 120850 instances
Test : 30213 instances
Pre-processing
Encoding: Categorical ('protocol_type', 'service', 'flag')
Log ('duration', 'src_bytes', 'dst_bytes')
Scaling/Normalization: Min-max
Class relabel: 5 categories
122 Input feature columns
Encoder
Compressed features
Decoder
Saved autoencoder weights
Autoencoder for unsupervised pre-training
Autoencoder Properties :
Encoding dimension=61
Input = Training Data
Optimizer=Adadelta
Loss= binary cross-entropy
Validation split=0.2
Batch size=50
No of epochs=50
Fully connected input layer
122 Neurons
Activation=ReLU
Hidden Layer
64 Neurons
Activation=ReLU
Batch Normalization +Dropout
Output Layer
5 Neurons
Activation=Soft-Max
Retrain
Supervised Classifier for Intrusion type classification
Classifier Properties :
Input = Training data, Training label
Optimizer=Adam
(Learning Rate=0.02)
Loss= categorical cross-entropy
Validation split=0.2
Batch size=50
No of epochs=100
5 output classes: DoS attack; Probe ; R2L attack; U2L attack; Normal

Reference: [6]

Yang et al. [7] proposed an improved conditional variational AutoEncoder (ICVAE) with DNN. Here the authors identified one of the issues associated with these models, which is biased due to some minority attacks R2L, for instance, having a very low number of occurrences in the training sets. This in the real world is an issue with zero-day and new attack vectors, resulting in very poor detection performance. The ICVAE proposed extracting high-level features and reduced the dimensionality. They were also able to generate new attack samples that they feed into the training data set to reduce the bias on certain attack classifications. It was also used to initialize the weights for the DNN, from which the output was fed into for classification. Figure 4 presents the proposed intrusion detection framework described in the publication.

Hindy et al. [8] focused on zero-day attack detection. The study used three algorithms to firstly drop correlated features and scale the features. The model then trained the autoencoder using hyperparameters identified from a random search and finally the model detected zero-day attacks using a mean squared error of the decoded output of autoencoder and the original input and see if it is greater than a given threshold. The threshold plays the role of determining the value for an instance in order to be considered a zero-day attack. The study then compares the results against a Support Vector Machine model.

Figure 4: The proposed intrusion detection framework

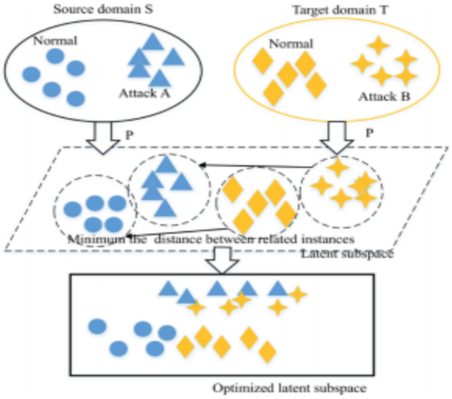


Abbreviation: ICVAE: improved conditional variational AutoEncoder
Reference: [7]

Transfer Learning

Zhao et al [9] proposed to use transductive transfer learning, which is an approach that uses "what it knows" in order to predict an unknown. The study indicated that networking attacks share common features and as such are similar. They created a clustering-enhanced transfer learning approach (CeHTL), which computes the similarities between each cluster and then selects two similarities to get the mappings of target domain to the source domain. They used Kmeans++ and euclidean distance to calculate the similarities between the source and target domain clusters (Figure 5).

Figure 5: Structure of CeHTL approach

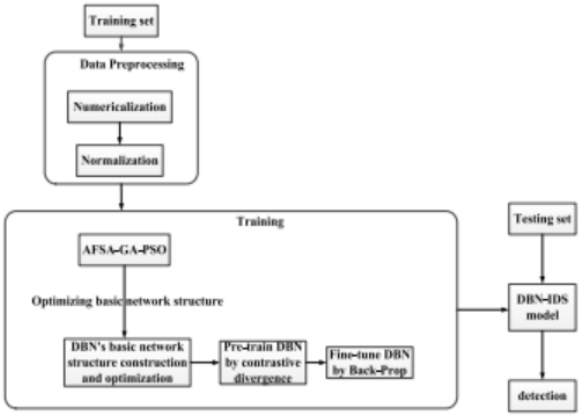


Reference: [9]

Deep Belief Network

Wei et al [10] introduced the concept of AFSA-GA-PSO which is a hybrid particle swarm optimization algorithm, to optimize the Deep belief network (DBN) during training. The concept of AFSA-GA-PSO is a combination of a Genetic Algorithm Particle swarm Optimization and Artificial fish swarm algorithm, in which the network structure is optimised to ensure classification accuracy of the DBN (Figure 6).

Figure 6: A summary of intrusion detection classification model construction and optimization method

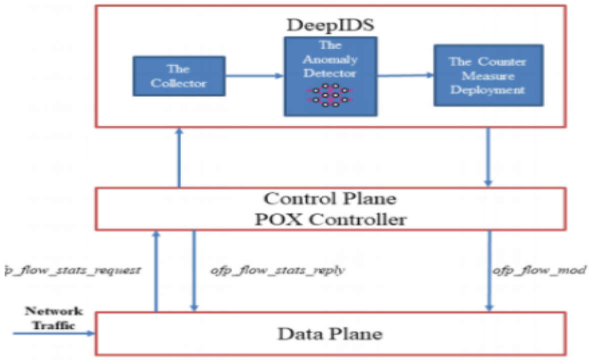

Training set
Data Preprocessing
Numericalization
Normalization
Training
AFSA-GA-PSO
Optimizing basic network structure
DBN's basic network structure construction and optimization
Pre-train DBN by contrastive divergence
Fine-tune DBN by Back-Prop
Testing set
DBN-IDS model
detection

Reference: [10]

Examples of implementation

Tang et al [11] proposed an implementation of a Machine Learning (DeepIDS) in a Software Defined Network. This was a flow based anomaly detection system. The authors implemented this by first collecting network flow in a time-based bins of length T (defined as a proportion of time). They then inputted these bins into the Deep Learning Model. They compared a fully connected Deep Neural Network and a Gated Recurrent Unit recurrent neural network (GRU-RNN). Then employed a countermeasure which gathers all the information about the classified anomalies from the previous step and inserted flow entries in order to drop the offending packets or redirect them to a honeypot (Figure 7).

Figure 7: The DeepIDS Architecture

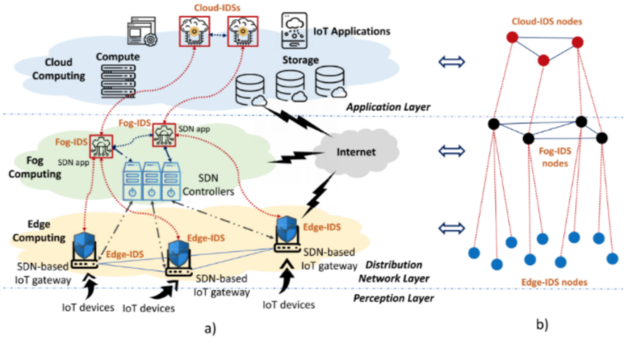


Reference: [11]

Ngyen et al [12] proposed a solution to secure cloud-based IoT networks through a tier distributed approach they call SeArch. The concept is that the Machine Learning Network Intusion detection is distributed through a cloud network and depending on the Layer ie Edge, Fog and Cloud, a different classification model is used in order to account for resources.

At the Edge, where the SDN IoTgateways resided a Support vector machine model is used as it would require less resources. It collects flow statistics that are fed into a machine learning model for classification and then performs policy making. At the Fog level there would be more compute and memory. A Self Organising Map (SOM) is used and at the Cloud level a Deep learning model is used due to the availability of resources. Each layer communicates and updates the others in order to produce more accurate decisions (Figure 8).

Figure 8: Collaborative and intelligent NIDS architecture for SDN-based cloud IoT Networks (a) NIDS node graph of the proposed architecture (b)



Reference: [12]

Critical appraisal

When reviewing the above publications I was interested in investigating potential issues such as: (1) Scalability, (2) proposed solution, (3) issues relating to networking, such as networking traffic volume, (4) dealing with diversity in the networking protocols, (5) dealing with low frequency attacks and zero day attacks, (6) how dynamic were the solutions when dealing with changing behaviour in the network, and (7) how easy was the solution to update and change.

The models identified and presented previously, used a range of different datasets in order to train and test the model. Datasets included KDD Cup 99, NSL-KDD, UNSW-NB15, cicids2017, CAIDA. There was also variation in the number of datasets used, single or multi datasets.

Among the presented models, a number of attack types were used for analysis; Dos, Probe, R2L (Remote to Local), U2R (User to Root) and the strength of the model results were assessed based on Accuracy, Precision, Recall, False Alarm and F-Score and reported using the following metrics: True Positive (TP) - Attack data that is correctly classified as an attack; False Positive (FP) - Normal data that is incorrectly classified as an attack; True Negative (TN) - Normal data that is correctly classified as normal; and False Negative (FN) - Attack data that is incorrectly classified as normal [4].

One of the main issues when using machine learning technology with network packets and flows is the high number of features i.e. packet header fields, which result in very high dimensions and what is referred to as the curse of dimensionality. This results in a model requiring large amounts of memory and time required to train the model, this would mean parameters would need to change every time to retrain the model. Among the first five papers reviewed all of them implemented a form of autoencoder to deal with feature extraction and deal with the dimensionality issue. There were different autoencoder approaches. The non-symmetric stack claimed to improve computational time as well as discovering complex relationships between features [4]. Yousefi-Azar [5] and Rezvy [6] both used deep autoencoders for the feature extraction comparing the increase in detection due to the feature extract with different classifiers, shallow and deep learning.

Across the Autoencoders different classifiers were used from shallow learning and deep learning which resulted in the range of values when it came to Accuracy, Precision, Recall, False Alarm and F-Score. Ranging from mid 80's to high 90's percentages. The one section the models did poorly on was U2R and RSL which was due to the limited number of samples for these types of attack in the datasets. This was the issue as discussed of low frequency attacks. Hindy et al [7] implemented a novel approach using autoencoders to generate new attack sample to add to the training set to reduce this issue.

Zhao et al [8] and Wei et al [9] looked at the issue of zero day attacks. Zhao et al [8] used thresholds and compared the decode output to the input to determine if there was a zero day attack. Wei et al [9] introduced a very different approach using transfer learning which can be used as an approach to deal with zero day attacks as it used the combat attacks.

Tang et al [10] introduced another tweak that can be used to get a better performance from the given model this time using a swarm algorithm instead of an autoencoder.

Nguyen et al [11] and Young et al [12] focused on implementation of Machine Learning NIDs in software defined and IoT networks. Nguyen et al [11] dealt with the centralised placement close to the controller of the NIDs module and Young et al [12] tackled the issue of network volume by implementing a distributed topology to handle the classification locally and at edge.

Limitations

With the range of topics and implementations discussed, there were issues relating to some attack classification (U2R and R2L) and although there were some attempts to tackle the issue the results from the models were not always positive.

Relationship to the proposed project

There were a lot of concepts discussed that tie into the proposed project around approaches to choosing and implementing a Machine Learning Classifier for Network Intrusion detection. Also some areas around dealing with issues around implementing, updating and maintaining a classifier in a modern large software defined network.

CONCLUSION

There is a need for a better approach to protect modern networks which is of growing concern and can be seen from the number studies being conducted in the area. However, there still seems to be some issues around dealing with the types of attacks and variation in results across different types of models. These are some of the areas that will be investigated in the proposed project.

REFERENCES

1. McAfee Labs Threats Report, November 2021. Accessed (Jan 2021) https://www.mcafee.com/enterprise/en-us/assets/reports/rp-quarterly-threats-nov-2020.pdf
2. Gemalto: State of IoT security Report. Accessed (Jan 2021) https://doi.org/10.1016/S1353-4858(19)30018-2
3. M-Trends 2020 – Fireeye mandiant services Special report. Accessed Jan 2021 https://content.fireeye.com/m-trends/rpt-m-trends-2020
4. Shone N., Ngoc T. N., Phai V. D. and Shi Q., "A Deep Learning Approach to Network Intrusion Detection," in IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 2, no. 1, pp. 41-50, Feb. 2018, doi: 10.1109/TETCI.2017.2772792.
5. Yousefi-Azar M., Varadharajan V., Hamey L. and Tupakula U., "Autoencoder-based feature learning for cyber security applications," 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, 2017, pp. 3854-3861, doi: 10.1109/IJCNN.2017.7966342.
6. Rezvy S., Petridis M., Lasebae A., Zebin T. (2019) Intrusion Detection and Classification with Autoencoded Deep Neural Network. In: Lanet JL., Toma C. (eds) Innovative Security Solutions for Information Technology and Communications. SECITC 2018. Lecture Notes in Computer Science, vol 11359. Springer, Cham. https://doi-org.dcu.idm.oclc.org/10.1007/978-3-030-12942-2_12
7. Yang, Y.; Zheng, K.; Wu, C.; Yang, Y. Improving the Classification Effectiveness of Intrusion Detection by Using Improved Conditional Variational AutoEncoder and Deep Neural Network. Sensors 2019, 19, 2528. https://doi.org/10.3390/s19112528
8. Hindy, H.; Atkinson, R.; Tachtatzis, C.; Colin, J.-N.; Bayne, E.; Bellekens, X. Utilising Deep Learning Techniques for Effective Zero-Day Attack Detection. Electronics 2020, 9, 1684. https://doi.org/10.3390/electronics9101684
9. Zhao, J., Shetty, S., Pan, J. et al. Transfer learning for detecting unknown network attacks. EURASIP J. on Info. Security 2019, 1 (2019).
10. Wei P., Li Y., Zhang Z., Hu T., Li Z. and Liu D., "An Optimization Method for Intrusion Detection Classification Model Based on Deep Belief Network," in IEEE Access, vol. 7, pp. 87593-87605, 2019, doi: 10.1109/ACCESS.2019.2925828.
11. Tang, T.A.; Mhamdi, L.; McLernon, D.; Zaidi, S.A.R.; Ghogho, M.; El Moussa, F. DeepIDS: Deep Learning Approach for Intrusion Detection in Software Defined Networking. Electronics 2020, 9, 1533.
12. Nguyen T. G., Phan T. V., Nguyen B. T., So-In C., Baig Z. A. and Sanguanpong S., "SeArch: A Collaborative and Intelligent NIDS Architecture for SDN-Based Cloud IoT Networks," in IEEE Access, vol. 7, pp. 107678-107694, 2019, doi: 10.1109/ACCESS.2019.2932438.
13. Young G. O., “Synthetic structure of industrial plastics (Book style with paper title and editor),” in Plastics, 2nd ed. vol. 3, J. Peters, Ed. New York: McGraw-Hill, 1964, pp. 15–64.