Deep Learning for IoT

Tao Lin
Amazon
Seattle, US
paper@Ltao.org

***Abstract*— Filter analysis is a fundamental stage in IoT cyber operations. The rapid data sources increase the needs on cyber security analysts' capability in terms of analytical reasoning. To help IoT data analysis more efficient, retrieval methods need to be proposed to facilitate data triaging through retrieval of the relevant historical data filter operations of senior security analysts. This paper presents a research of data retrieval based on deep learning. It further directs the new approaches in adversarial machine learning situation.**

Keywords—Deep Learning, Machine Learning, Adversarial Machine Learning, IoT

I. INTRODUCTION

Machine learning, especially Deep Learning, is increasingly popular not only in daily life, but also in many science disciplines, including Internet of Things or IoT[1]. For example, computer security in terms of IoT network intrusions' detection, and malware identification relies on automatic approaches stemming from machine learning, but those are only two examples of machine learning in IoT security. Whereas, machine learning is good at average cases, such as the well-known example of sorting fish automatically by their inherent features. On the other hand, security is related to worst cases. It is not hard to bypass a machine learning based content filter through malicious manipulations in adversarial settings. An example of this is combining malicious samples with benign files, evading several PDF malware classifiers. Therefore, the safe adoption of machine learning approaches in IoT security settings is an unsolved challenge.

Adversarial machine learning is crucial in life-critical IoT systems[2], such as roadside sign recognition used by autonomous vehicles. To be specific, small nonobvious manipulations in roadside signs can lead to distinct opposite results in specific machine learning methods. It is not an easy task to guarantee accuracy and sensitivity simultaneously.

This paper will focus on two aspects to implement machine learning in adversarial environments using more robust and feasible approaches. First, this paper will suggest do some research on machine learning's transferability. The second research question is about effective defense against adversaries in machine learning in IoT.

Adversaries can launch transferability attacks through constructing an independent machine learning model to simulate some other models just using the input data and output labels, without any insights on the original machine learning models' parameters, even the models' type. Transferability is significant not only in adversarial machine learning, but also in many other machine learning applications. On the one hand, although deep learning has achieved impressive successes in many areas, many details are still unclear. Some simple machine learning approaches have similar results with complicated and computationally expensive deep learning algorithms. An open question is whether or not we can simplify the features or hierarchies in deep learning models through transferability. The other question is that although different machine learning models can generate similar outputs from same inputs, we cannot use same evading techniques to attack different machine learning algorithms. In other words, we can protect machine learning models through these transferability properties.

In addition, compared to many state-of-the-art approaches on evading and positioning machine learning models, there is little research to defend the adversaries. Protecting classifiers through ensemble learning, hiding the classification probability scores, or hiding features, even hiding entire classifiers are not appropriate methods, partly because of machine learning's transferability. I propose that defending the adversaries by leveraging reinforcement learning through adversarial training, which is intentionally generating adversarial examples as part of the training procedure. The main challenge is how to craft relevant adversarial examples to simulate the real settings.

II. DEEP LEARNING BASED RETRIEVAL OF IOT FILTER OPERATIONS

Due to the following observations, deep learning could play an essential role in developing better IoT data filter operation retrieval systems.

Firstly, the methods we have discussed in the previous sections make use of pre-determined similarity measurements when checking which historical data filter operations are most relevant to the current cyber situation. On the other hand, there is no guarantee that the pre-determined similarity metrics are the most suitable. Machine learning could be leveraged to help learn the most suitable similarity metrics.

Secondly, data filter operation retrieval systems must be able to handle a variety of uncertainties such as the uncertainty introduced by false positives, false negatives, and incomplete information.

Machine learning could be leveraged to increase retrieval systems' capability in dealing with the uncertainties.[3] Machine learning, especially neural networks, is a potential approach, which can be used for data filter operation retrieval in a SOC. There are a variety of artificial neural networks, such as convolutional neural networks, long short-term memory, and deep belief networks. Instead of providing a comparative

viewpoint, below we only discuss the potential application of recurrent neural networks.

III. Data Filter Operation Retrieval based on Recurrent Neural Networks

For data filter operation retrieval[4], the most promising neural networks approach seems to be recurrent neural networks (RNN), mainly because this type of neural network is good at dealing with sequence data. One of the most notable features in data filter operations is that security-related events are sequential.

The fundamental philosophy behind RNN models is that rather than rewriting all information, each element in an RNN model updates the current state by adding new information. Accordingly, when an RNN is trained to classify the newly arrived data filter operations, the RNN can be incrementally maintained to incorporate substantial new data triaging knowledge.

But, before training and deploying any RNNs in a SOCs, the SOC should cautiously consider the potential adversaries.

A new challenge which is faced by a SOC but is not addressed in other knowledge retrieval systems is that data filter operations are being retrieved in adversarial settings. That is, the attacker may purposely obfuscate their attack actions in such a way that the accuracy of filter operation retrieval could be significantly reduced.

Recently, substantial research work has shown that most existing machine learning classifiers are highly vulnerable to adversarial examples. The RNNs deployed in a SOC should be resilient to adversarial examples.

IV. IoT Data Filter Model

Cyber security data filter is targeted at determining whether the incoming data sources are worth of further investigation in a timely and quick manner. To achieve this goal, security analysts usually conduct a sequence of data filter operations to filter malicious network events and to group them according to the potential attack chains. Therefore, the unit of data filter analysis is a network event. Network events are the data reported by various network monitoring sensors, including SIEM tools and human intelligence agents,

A *network event* can be abstracted as a multi-tuple of its characteristics,

$$e = < t_{occur}, t_{detect}, type, attack_{prior}, sensor, protocol, ip_{src}, port_{src}, ip_{dst}, port_{dst}, severity, confidence, msg >,$$

where t_{occur} is the time the event occurred; t_{detect} is the time the event first being detected; *type* is the type of network connection activity (e.g., Built, Teardown or Deny); $attack_{prior}$ is the attack type of the event being detected by a sensor/agent based on prior knowledge; *sensor* is the sensor/agent who detected this event; *protocol* is the network protocol; ip_{src}, $port_{src}$, ip_{dst}, $port_{dst}$ are respectively the source IP, source port, destination IP, and destination port; *severity* and *confidence* specify the level of severity and confidence of the event, respectively; *msg* specifies other characteristics of the event, which depends on the sensor.

An example of a data filter process is an analyst performs a sequence of data filter operations to identify suspicious network events. Each data filter operation specifies a constraint for the events to narrows down the original data set. As the examples, there are mainly three types of data filter operations:

- FILTER (*D*, C): to filter a set of events (*D*) based on a constraint (C).
- SEARCH (*D*, C): to search a keyword (C) in an event set (*D*).
- SELECT (*D*, C): to select a subset of events with a common feature C from a set (*D*).

The analyst specified several criteria of the suspicious or correlated network events based on the domain knowledge and experience. Each criterion specifies a constraint on the network event characteristics, so that a data filter operation can select and correlated network events.

V. Filter Operations through Time

Gradient-based learning requires a closed-form relation between the model parameters and the loss function. This relation allows to propagate the gradient information calculated on the loss function back to the model parameters, in order to modify them accordingly. While this operation is straightforward in models represented by a directed acyclic graph, such as a Feedforward Neural Network (FFNN), some caution must be taken when this reasoning is applied to RNNs, whose corresponding graph is cyclic. Indeed, in order to find a direct relation between the loss function and the network weights, the RNN has to be represented as an equivalent infinite, acyclic, and directed graph. The procedure is called unfolding and consists of replicating the network's hidden layer structure for each time interval, obtaining a particular kind of FFNN. The key difference of an unfolded RNN with respect to a standard FFNN is that the weight matrices are constrained to assume the same values in all replicas of the layers, since they represent the recursive application of the same operation.

Training a neural network commonly consists of modifying its parameters through a gradient descent optimization, which minimizes a given loss function that quantifies the accuracy of the network in performing the desired task. The gradient descent procedure consists of repeating two basic steps until convergence is reached. First, the loss function L_k is evaluated on the RNN configured with weights $\mathbf{W}_k$, when a set of input data X_k are processed (forward pass). Note that with $\mathbf{W}_k$ we refer to *all* network parameters, while the index k identifies their values at epoch k, as they are updated during the optimization procedure. In the second step, the gradient $\partial L_k / \partial \mathbf{W}_k$ is backpropagated through the network in order to update its parameters (backward pass).

VI. Knowledge Matching and Rule Relaxation

Given the rule-based representation, a past incident can be described by a rule condition, which includes every specific

detail at that moment, such as the time slot and the geographical location of the events. Therefore, an experience will not repeat itself with each same single detail.

As shown in the retrieval framework, the current context will be searched in the knowledge base. However, the rule matching requires every single detail of the rules to be matched, which may limit the usefulness of the retrieval results.

The problem is: how can we make a limited number of experiences useful for assisting to detect similar events? An experience will not be useful if we do not abstract the particular details. It is significant to retrieve the key parts of an experience and to relax the experience by choosing the portions, which are not too specific.

To make the rule matching more flexible, Chen et al. proposed rule relaxation based on the Horn clause representation.

In regard to rule-based representation, researchers can relax the constraints by removing conditions from antecedents of that rule.

Conditions with lower priorities can be relaxed. Experience includes specific details, such as the exact time and location of the incident. Therefore, an experience will not repeat itself with each single detail remaining the same. It is desirable to retrieve the important parts of an experience and to relax the experience by trimming those portions that are too specific.

The higher the degree to which an experience can be relaxed, the higher the possibility exists that it can be matched against a new situation. Figure 1shows that the knowledge generated by relaxation form a hierarchy: the most specific knowledge at the bottom while the top is the most relaxed ones.

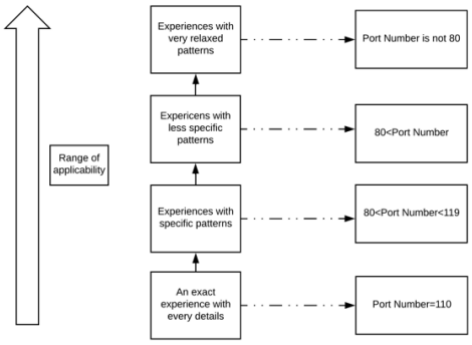

Figure 1 Experience relaxation levels.

Overall, upper-level experiences have better precision. While lower level experiences provide broader coverage. The entire experience hierarchy is formed through a consistent process, where each level of relaxation is defined with a specification guideline (i.e., how a higher-level experience should be relaxed into lower-level ones). All experiences on the same level will have a consistent specificity. According to Figure 2, rule matching is performed on each piece of knowledge in the network. Rule relaxation enables a larger set of matching candidates. Meanwhile, it may influence the precision of the results.

Figure 2 Hierarchical experience networks.

VII. Context-Driven and Efficient

Our goal is to retrieve the relevant traces based on the current context of an analyst' data filter process and to quickly update the results along with the context change. Our approach is developed based on two main insights.

Insight 1: Recurrent neural network can be used to represent the context information because it can perfectly capture the course of data filter operations and the dependencies between these operations.

Insight 2: The retrieval results need to be updated dynamically along with the changes of the current context. It requires the graph-based approach to be efficient enough for timely updates. Therefore, we need to avoid graph isomorphism analysis.

Inspired by these two insights, we adopted a deep learning approach. The concept of neural network originally comes from biological brain, which composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems.

VIII. Challenges in Using Machine Learning for Data Filter Operation Retrieval

Machine learning has been playing an increasingly important role in performing various tasks in SOCs. For example, network intrusion detection systems and malware classification systems are leveraging more and more automation achieved through machine learning.

However, although machine learning is good at (dealing with) average cases, it is not easy to implement any machine learning methods for data filter operation retrieval systems, since data filter operation retrieval systems are related to worst cases. It is possible to bypass a machine learning based content filter through malicious manipulations in adversarial settings. The attacker could combine malicious samples with benign events to evade several retrieval classifiers. For example, some very small manipulations in events logs can lead to distinct opposite results in data filter operation retrieval systems. It is not an easy task to guarantee accuracy and sensitivity simultaneously. In data filter operation retrieval, because of the inherent temporal relationships between events, the adversary has the possibility to infer the similarity metrics to bypass the retrieval system.

IX. Conclusion

A major challenge of data filter in IoT area is the inefficient performance of junior security analysts caused by the lack of experience.

It can be effectively addressed through retrieval of the relevant past data filter operations performed by the senior analysts. We conducted a novel research on data filter knowledge retrieval methods and discussed the new directions in solving the retrieval problem in this field.

X. Acknowledge

Part of this work is from the author's PhD study, before the author joining Amazon. Professor Fu Chen from Central University of Finance and Economics provided many constructive suggestions and perspectives for this work during author's PhD study. This work and Professor Fu Chen were supported in part by National Science Foundation of China under No.61672104.

References

[1] T. Lin, "A Data Filter Retrieval System for Cyber Security Operations Center," *Pennsylvania State Univ. Thesis*, 2018.

[2] T. Lin, C. Zhong, J. Yen, and P. Liu, "Retrieval of Relevant Historical Data Filter Operations in Security Operation Centers," in *From Database to Cyber Security*, Springer, Cham, 2018, pp. 227–243.

[3] T. Lin, J. Gao, X. Fu, and Y. Lin, "A Novel Bug Report Extraction Approach," in *International Conference on Algorithms and Architectures for Parallel Processing*, 2015, pp. 771–780.

[4] C. Zhong, T. Lin, P. Liu, J. Yen, and K. Chen, "A cyber security data filter operation retrieval system," *Comput. Secur.*, vol. 76, pp. 12–31, 2018.