Active Learning for Intrusion Detection Systems

Quang-Vinh Dang
Data Innovation Lab
Industrial University of Ho Chi Minh City, Vietnam
dangquangvinh@iuh.edu.vn

***Abstract*—Intrusion Detection Systems (IDSs) play a vital role in the modern cyber-security system. The main task of an IDS is to distinguish between benign and malicious network flows. Hence, the researchers and practitioners usually utilize the power of machine learning techniques by considering an IDS as a binary-classifier. Recent research works demonstrate that an ensemble learning algorithm like xgboost can achieve almost perfect classification in the offline configuration. On the other hand, the performance of a simple and lightweight classification algorithm like Naive Bayes can be improved significantly if we can select a proper sub-training set. In this paper, we discuss the usage of active learning in online configuration to reduce the labeling cost but maintaining the classification performance. We evaluate our approach using the popular real-world datasets and showed that our approach outperformed state-of-the-art results.**

***Index Terms*—cyber-security, intrusion detection systems, active learning**

I. Introduction

Intrusion Detection Systems (IDSs) are important components of modern information technology systems [1]. In short, the task of an IDS is to detect and classify any malicious activities that happen in a computer system that allows the quick and efficient reactions from the administrators, either manual or automatic.

By its nature, the core of an IDS is usually a machine learning classifier. The task of the machine learning algorithm is, given the information of a network flow, classify the network flow in one of two classes: *benign* or *malicious*. It is a well-known binary classification problem that has been studied for a long time in machine learning community. Over years, the researchers and practitioners have evaluated different algorithms and techniques to improve the classification performance of IDSs [2]. In recent years, along with the emergence of IoT devices, the requirement for IDSs have been extended to these environments [3].

Recent research study [1] showed a very interesting result. For the traditional classification task of an IDS, i.e. to classify benign and malicious network flows, ensemble learning algorithms such as xgboost [4] performed almost perfectly and achieved the AUC score around 99%. However, the main concern is that ensemble learning like boosting machines require a huge computational power, hence it is not practically to deploy them in IoT devices. The authors of [1] demonstrated that even in general, a lightweight and simple algorithm like Naive Bayes cannot compete with ensemble machine learning algorithms, we still can improve the classification performance by choosing a proper training dataset while keep avoiding over-fitting. However, the method to choose the sub-training set is not discussed comprehensively in the study.

In this paper, we present our approach by using active learning technique to actively select training sample. Different from existing active learning algorithms, we rely on the idea that *rare* events are more important for learning, particularly for the Naive Bayes algorithm. We review state-of-the-art studies in Section II then present our approach in Section III. We evaluate our idea and show the experimental results in Section IV and conclude our paper in Section V.

II. Literature Review

A. Intrusion Detection Systems

The researchers and practitioners have deployed many different algorithms and variants as the core of IDSs. In general, they can be grouped into supervised and unsupervised settings [1].

Before the era of deep learning, the researchers [5]–[7] have proposed to use traditional machine learning methods such as decision-tree or logistic regression for the problem of intrusion detection. Feature generation by genetic programming has been considered by [7]. However, the following studies do not show a significant improvement of automatic feature generation compared to the manually building ones [1]. Other popular classification algorithms such as SVM have been used until recently [3]. However, given the development of the computational systems and machine learning algorithms, ensemble learning algorithms like random forest are preferred [8].

As deep learning attracts a lot of attention since its victory in ImageNet competition in 2012 [9], the power of deep neural networks have been utilized in IDS. The authors of [10] employed multi-layer neural networks and NLP techniques to analyze the behavior of computer systems. [11] studied the time-dependence features of the network flows.

In the intrusion detection problem, the unsupervised machine learning algorithms are mostly the anomaly-detection algorithms [12]. Anomaly-detector tries to distinguish between normal, or benign flows, with un-normal flows without explicitly defining what *normal* is. The authors of [13] used One-class SVM algorithm for the discussing problem. Recent surveys such as [14] surveyed multiple efforts of using anomaly-detection in IDSs.

One of most powerful anomaly-detection algorithms up to date is Isolation Forest [15], [16]. The main idea of Isolation Forest is that it tries to distinguish a single instance, and more

difficult to distinguish an instance, more *normal* it is. Isolation Forest uses decision tree as its base learner.

B. xgboost

In our model, we use xgboost to predict the performance impact of the algorithm.

xgboost stands for eXtreme Gradient Boosting [4] is a ensemble technique that has been introduced as one member of gradient boosting family. The main idea of gradient boosting techniques is to build multiple sequential learners (will be referred as *weak learners*) where the next learner tries to correct the error made by the previous one.

Hence, the final boosting model of xgboost will be:

$$F_m(x) = F_0(x) + \sum_{i=1}^{M} \rho_i h_i(x, a_i) \tag{1}$$

$F_0(x)$ is the initial model: $F_0(x) = argmin_\rho \sum_{i=1}^{N} l(y_i, \rho)$, i.e. $F_0(x)$ can be initialized as constant. ρ_i is the weight value of the model number i, h_i is the base model (decision tree) at the i^{th} iteration.

xgboost also introduced a new regularization term. The objective function is usually defined as:

$$obj(\theta) = l(\theta) + \Omega(\theta) \tag{2}$$

with l is the loss function (e.g. squared-error $\sum (y_i - \hat{y}_i)^2$) and Ω is regularization term. The details of xgboost model can be found in the original paper [4].

C. Active Learning

Active Learning is a sub-field of machine learning wherein the algorithm actively select the next instance for labelling [17]. Hence, active learning can reduce the training size while maintaining the performance of the algorithms by choosing only *predictive* instance from the pool for labelling. Active learning assumes an *oracle* that can precisely label all unknown instances. In our setting, the oracle can be considered as xgboost.

Active learning can employ different strategies to select the next sample [17]–[19].

Uncertainty-based sampling is probably the first active learning technique [18]. In this strategy the model picks the *least confidence* sample, defined as:

$$X^*_{LC} = argmax_X (1 - P_\theta(\hat{y}|X)) \tag{3}$$

wherein:

$$\hat{y} = argmax_y P_\theta(y|X) \tag{4}$$

Other confidence measures are proposed such as entropy or margin [19].

The other strategy is called *Query by Committee* [19]. The core idea of this method is to build a set of models, then let all the models to predict in all the unknown instances. The instance that the models disagree most will be chosen to send to the oracle. The authors of [20] proposed two methods: *Bagging* and *Boosting* to construct the committee. The idea of bagging and boosting committee is borrowed from the same terminologies in machine learning. Furthermore, there are multiple ways to measure the *disagreement* of voters. Two popular metrics are K-L divergence and vote entropy.

The third strategy is called *Expected Model Change*. The idea of this strategy is to select the instance which might change the performance of the model most. One example of the strategy is *Expected Gradient Length* [17]:

$$X^*_{EGL} = argmax_X \sum_i P_\theta(y_i|X) || \bigtriangledown l_\theta(L \cup (X_i, y_i)) || \tag{5}$$

The idea is to check what instance if added will change the gradient of the learning model the most. Other examples include Variance Reduction Method [19].

One very important idea in existing active learning methods is "informative instances should not only be those which are uncertain, but also those which are 'representative' of the underlying distribution" [17]. Hence, the density-based method is proposed:

$$X^*_{ID} = argmax_X (\phi_A X x (\frac{1}{U} \sum_{u=1}^{U} sim(x, x^{(u)}))^\beta) \tag{6}$$

The idea of the Equation 6 is to avoid selecting some *outlier* instance by using the function *similarity* (sim).

III. Our Approach

We rely on different idea than the density-based active learning [17].

We recall the Naive Bayes formula that assuming the independence between features:

$$P(f_1, f_2, ..f_n|BENIGN) = \prod_1^n P(f_i|BENIGN) \tag{7}$$

From the Equation 7 we can claim that Naive Bayes has a computational speed advantage compared to other algorithms.

As we can see from the Bayesian formula, we might expect the bigger change of $P(A|B)$ if we observe a *rare* event, i.e. if $P(B)$ is small. This observation contradicts the density-based idea. Given that we are aiming to improve the performance of the Naive Bayes, our idea fits perfectly to the scheme. Hence, instead of choosing a next instance that is not too far from the distribution like other studies [17], [19], we actively choose the instance that is not so common in the observed data.

We use Isolation Forest to measure the difference of an instance to the observed distribution. Furthermore, we trained a xgboost model to predict the *performance impact* in term of change in AUC score if we add a particular training instance into the training dataset.

We visualize our proposal in Figure 1. The threshold for Isolation Forest score is adapted by the probability of picking up a single training instance over time.

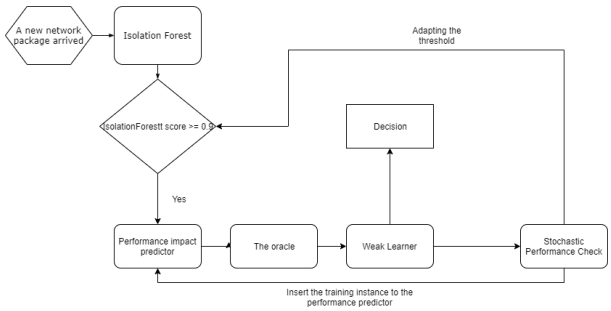

Fig. 1. Active Learning for IDSs

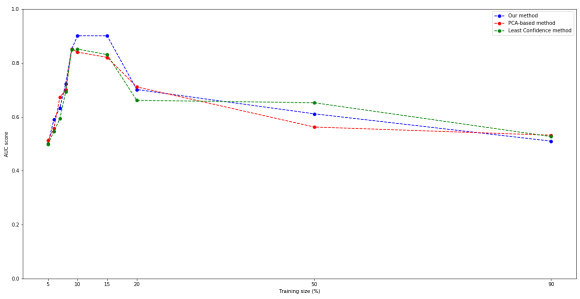

Fig. 2. Performance of different active learning strategies

IV. Experimental Results

A. Dataset

We evaluate our approach using CICIDS'12 dataset [21]. The dataset is collected in total seven days in 2010 with different types of attacks [22] from a real-world computer system. Each network flow is provided with a list of 78 features.

We follow the settings of [1] for a fair comparison.

B. Experimental Results

We evaluate Least Confidence strategy against our strategy. Both strategies started with 5% of the training set as the initial set. When the network flow arrived the algorithm decides to include the particular network flow into the training set or not. In both cases, a test-set is hold out for evaluation.

In general, our method can achieve the AUC score of 90% as the highest score, compared to the score of 85% achieved by the method based on Lease Confidence or PCA [1].

We display the results in Figure 2. We could see that all the active learning strategies peak their performance when we select around 10% of the training data, clearly suggested that there is a proper sub-training size which accounts for around one tenth of the training samples that actually bring the predictive power to the machine learning algorithms. Furthermore, the performance of Naive Bayes drops quickly if we increase the number of training size (by adaptive threshold method we described above). Contradict with the common sense in the machine learning community that we always need more data, the results showed that we rather need *good* data.

V. Conclusions

In this paper we present our proposal to use outlier detection algorithm as the active learning base learner to improve the performance of lightweight intrusion detection methods. The algorithm does not require huge computational power, hence it is suitable for low-power devices like IoT devices or smartphones. In the future we will study other methods to improve the performance of lightweight methods to be comparable with ensemble methods, such as considering the trustworthy of the partners [23].

References

[1] Q. Dang, "Studying machine learning techniques for intrusion detection systems," in *FDSE*, ser. LNCS, vol. 11814. Springer, 2019, pp. 411–426.

[2] Z. He, T. Zhang, and R. B. Lee, "Machine learning based ddos attack detection from source side in cloud," in *CSCloud*. IEEE Computer Society, 2017, pp. 114–120.

[3] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, "Network intrusion detection for iot security based on learning techniques," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 3, pp. 2671–2701, 2019.

[4] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *KDD*. ACM, 2016, pp. 785–794.

[5] C. Krügel and T. Toth, "Using decision trees to improve signature-based intrusion detection," in *RAID*, ser. LNCS, 2003.

[6] N. B. Amor, S. Benferhat, and Z. Elouedi, "Naive bayes vs decision trees in intrusion detection systems," in *SAC*. ACM, 2004, pp. 420–424.

[7] G. Stein, B. Chen, A. S. Wu, and K. A. Hua, "Decision tree classifier for network intrusion detection with ga-based feature selection," in *ACM Southeast Regional Conference (2)*. ACM, 2005, pp. 136–141.

[8] P. A. A. Resende and A. C. Drummond, "A survey of random forest based methods for intrusion detection systems," *ACM Comput. Surv.*, vol. 51, no. 3, pp. 48:1–48:36, 2018.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.

[10] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, 2019.

[11] W. Wang, Y. Sheng, J. Wang, X. Zeng, X. Ye, Y. Huang, and M. Zhu, "HAST-IDS: learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection," *IEEE Access*, 2018.

[12] Z. Chiba, N. Abghour, K. Moussaid, A. E. Omri, and M. Rida, "Intelligent and improved self-adaptive anomaly based intrusion detection system for networks," *IJCNIS*, vol. 11, no. 2, 2019.

[13] E. Eskin, "Anomaly detection over noisy data using learned probability distributions," in *ICML*. Morgan Kaufmann, 2000, pp. 255–262.

[14] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network anomaly detection: Methods, systems and tools," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 1, pp. 303–336, 2014.

[15] F. T. Liu, K. M. Ting, and Z. Zhou, "Isolation forest," in *ICDM*. IEEE Computer Society, 2008, pp. 413–422.

[16] Q. Dang, "Outlier detection on network flow analysis," *CoRR*, vol. abs/1808.02024, 2018.

[17] B. Settles, "Active learning literature survey," University of Wisconsin-Madison Department of Computer Sciences, Tech. Rep., 2009.

[18] A. Ziai, "Active learning for network intrusion detection," *CoRR*, vol. abs/1904.01555, 2019.

[19] G. M. Alqaralleh, M. A. Alshraideh, and A. Alrodan, "A comparison study between different sampling strategies for intrusion detection system of active learning model," *JCS*, 2018.

[20] N. Abe and H. Mamitsuka, "Query learning strategies using boosting and bagging," in *ICML*. Morgan Kaufmann, 1998, pp. 1–9.

[21] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Computers & Security*, 2012.

[22] Q. Dang and J. François, "Utilizing attack enumerations to study SDN/NFV vulnerabilities," in *NetSoft*. IEEE, 2018, pp. 356–361.

[23] Q. Dang and C. Ignat, "Computational trust model for repeated trust games," in *Trustcom/BigDataSE/ISPA*. IEEE, 2016, pp. 34–41.