RAL - Reinforcement Active Learning for Network Traffic Monitoring and Analysis

Sarah Wassermann, Thibaut Cuvelier, Pedro Casas

RAL – Reinforcement Active Learning for Network Traffic Monitoring and Analysis

Sarah Wassermann
AIT Austrian Institute of Technology
sarah.anne.wassermann@gmail.com

Thibaut Cuvelier
CentraleSupélec
cuvelier.thibaut@gmail.com

Pedro Casas
AIT Austrian Institute of Technology
pedro.casas@ait.ac.at

ABSTRACT

Network-traffic data usually arrives in the form of a data stream. Online monitoring systems need to handle the incoming samples sequentially and quickly. These systems regularly need to get access to ground-truth data to understand the current state of the application they are monitoring, as well as to adapt the monitoring application itself. However, with in-the-wild network-monitoring scenarios, we often face the challenge of limited availability of such data. We introduce RAL, a novel stream-based, active-learning approach, which improves the ground-truth gathering process by dynamically selecting the most beneficial measurements, in particular for model-learning purposes.

CCS CONCEPTS

• **Security and privacy → Network security**; • **Computing methodologies → Machine learning algorithms**; *Reinforcement learning*; *Active learning settings*; *Online learning settings*;

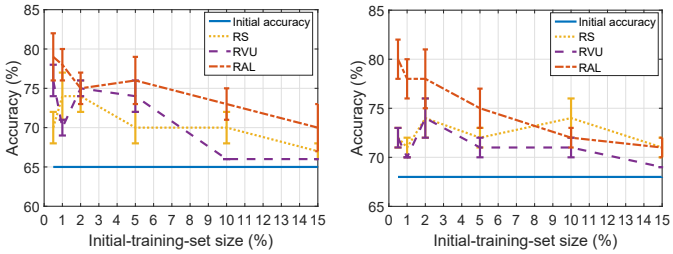
KEYWORDS

Reinforcement Learning; Active Learning; Network Monitoring

ACM Reference Format:
Sarah Wassermann, Thibaut Cuvelier, and Pedro Casas. 2020. RAL – Reinforcement Active Learning for Network Traffic Monitoring and Analysis. In *ACM Special Interest Group on Data Communication (SIGCOMM '20 Demos and Posters), August 10–14, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3405837.3411390

1 INTRODUCTION

One of the main challenges associated with supervised learning for network monitoring and analysis under dynamic scenarios is that of periodically getting access to labels of fresh, previously unseen samples. Labeling new data is usually an expensive and cumbersome process, while not all measurements/data samples are equally valuable. Active learning aims at labeling only the most informative samples to reduce cost. In this paradigm, a learner can choose from which new samples it wants to learn, and can obtain the ground truth by asking an oracle for the corresponding labels. We introduce RAL – Reinforced stream-based Active Learning, a new active-learning approach, coupling stream-based active learning with reinforcement-learning concepts. RAL dynamically decides at which specific times it is better to query the oracle for samples coming as an online, non-periodic, sequential stream. In particular, we model active learning as a contextual-bandit problem, in which rewards are based on the usefulness of the system's querying behavior. In this paper, we apply RAL on the specific case of network-attack detection.



(a) Ping flood. (b) Netscan.

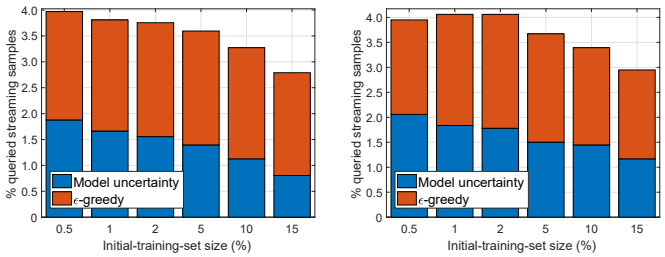
Figure 1: Attack-detection accuracy for RAL/RVU/RS.

2 THE RAL APPROACH

RAL relies on prediction uncertainty and reinforcement-learning principles, using rewards and bandit algorithms. The intuition behind the different reward values is that we attribute a high (positive) reward in case the system behaves as expected, and a low (negative) one otherwise, to penalize it. Our technique obtains rewards/penalties as soon as it is asking for ground truth. In a nutshell, it earns a positive reward in case it queries the oracle and the underlying model would have predicted the wrong label (i.e. the system made the right decision to ask for the ground truth) and a penalty when it asks the oracle even though the underlying classification model would have predicted the correct label (i.e. querying was unnecessary). The rationale for using reinforcement learning is that the system learns not only based on the queried samples themselves, but also from the usefulness of its decisions. Also inspired by the bandit literature and to better deal with concept drifts in the data, we implement an ε-greedy policy, which improves the data-space exploration. This ensures that we have a good chance of detecting potential concept drifts. The proposed approach relies on a committee of experts (i.e. different machine-learning models). Each expert gives its advice for the sample to consider: should the system ask the oracle for feedback or is the expert confident enough about its label prediction? The query decision of the committee takes into account the opinions of the experts, but also their decision power: if the weighted majority of the experts is certain enough, our algorithm will rely on the label prediction provided by the committee, used in the form of a voting classifier. The decision power of each expert gets updated such that the experts which

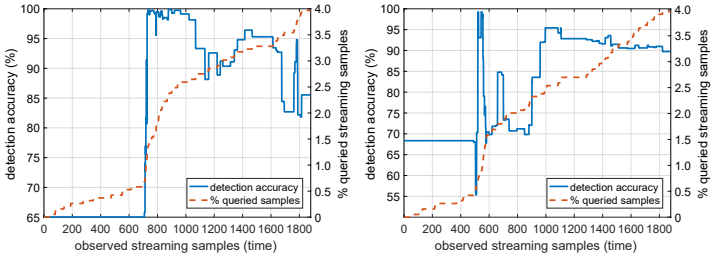
SIGCOMM '20 Demos and Posters, August 10–14, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-8048-5/20/08…$15.00
https://doi.org/10.1145/3405837.3411390



(a) Ping flood. (b) Netscan.

Figure 2: Percentage of queries issued by RAL. Queries indicate the underlying origin – either due to model uncertainty, or by exploration.



(a) Ping flood. (b) Netscan.

Figure 3: RAL temporal convergence.

agree with the entire committee are obtaining more power in case that particular decision is rewarding, i.e. informative (otherwise, these experts get penalized). These weights are updated through EXP4 [1]. In case the system decides not to query, the committee is used as a voting classifier to perform the predictions. With RAL, we propose a system built on reinforcement-learning principles in the challenging setting of stream-based learning, while previous work [2, 3] relied on reinforcement learning only in pool-based settings.

3 EVALUATION

We benchmark RAL against a state-of-the-art active-learning algorithm (RVU) [4], using prediction uncertainty for querying decisions and randomizing the certainty threshold for concept drifts, as well as against random sampling (RS), on the MAWI dataset [5], a public cyber-security dataset. We focus on two different types of attacks, namely flooding and netscan intrusions. For each benchmarked algorithm, we proceed as follows: first, we subdivide the considered datasets into three consecutive, disjoint parts, namely the *initial training set*, the *streaming data*, and the *validation set*. The validation set consists of the last 30% of the data, the initial training set is a variable fraction of the first samples, and the streaming part includes all the remaining samples. We then train a model on the initial training set and check its accuracy on the validation part (*initial accuracy*). Next, we run the specific algorithm on the streaming part and let it pick the samples it decides to learn from. We retrain the models after a new queried label. Finally, we evaluate the final model, trained on the initial training set plus the selected samples, again on the validation set, and analyze its prediction accuracy (*final accuracy*). In the context of this evaluation, RAL's committee is a voting classifier composed of a k-NN model with $k = 5$, a decision tree, and a random forest with 10 trees. We use the same model for RVU and RS. We use the following parameter values for RAL: $\rho^+ = 1$, $\rho^- = -1$, $\theta = 0.9$, $\varepsilon = 2.5\%$, and $\eta = 0.01$. We test RVU with the parameters recommended in [4].

Figure 1 shows that RAL outperforms both RVU and RS on average. Furthermore, when it comes to the number of queried samples, we observe in Figure 2 that RAL queries, on average, a small share of samples – less than 4%, which is significantly less often than RVU (not shown here for space limitations), which asks about 17 percentage points more with respect to RAL.

We also study the convergence of RAL's attack-detection performance with respect to the evolution of the streaming samples (i.e., time), for the two MAWI attack datasets. More precisely, we evaluate RAL on the validation set every time a new sample is queried. We set the initial training-set size to the first 0.5% of the data; according to Figure 1, such a small initial training-set provides the best results. In Figure 3, we plot the accuracy convergence for the ping-flood and netscan detection. We observe that the detection accuracy is not clearly converging in the two scenarios: the ping-flood-detection performance seems to converge to 90%, while there does not seem to be any convergence for the netscan case. This is not surprising, considering the fact that these datasets present multiple concepts drifts and are very dynamic.

We investigated the reasons behind the sharp accuracy increases and found that they are most of the time highly correlated with queries issued by the committee (and not the ε-greedy scenario), as the models had a low confidence in their prediction. Acquiring the labels for those samples proves to be very beneficial for RAL. The degradation of the detection performance is likely due to the noise in the dataset and to the concept drifts. Significant decreases in accuracy are mostly caused by samples queried by random exploration (ε-greedy) and not RAL's committee, even though this mechanism also often provides performance boost by forcing our system to explore the data space.

As an overall conclusion for RAL, the presented initial results show that RAL is a promising approach for stream-based network-monitoring applications, making the most out of the information which can be extracted from a stream of measurements, while reducing the need for costly labeled, ground-truth data. RAL provides a completely different exploration-exploitation trade-off than existing algorithms, as it queries fewer samples of higher relevance.

REFERENCES

[1] P. Auer et al.: The Nonstochastic Multiarmed Bandit Problem. SIAM Journal on Computing, vol. 32(1), pp. 48–77, 2002
[2] Y. Baram et al.: Online Choice of Active Learning Algorithms. Journal of Machine Learning Research (JMLR), vol. 5, pp. 255–291, 2004
[3] W.N. Hsu et al.: Active Learning by Learning. AAAI Conference on Artificial Intelligence (AAAI), 2015
[4] I. Žliobaitė et al.: Active Learning With Drifting Streaming Data. IEEE Transactions on Neural Networks and Learning Systems, vol. 25(1), pp. 27–39, 2014
[5] R. Fontugne et al.: MAWILab: Combining Diverse Anomaly Detectors for Automated Anomaly Labeling and Performance Benchmarking. ACM CoNEXT, 2010