Benchmarking Genetic Programming in Dynamic Insider Threat Detection

Duc C. Le, Malcolm I. Heywood, Nur Zincir-Heywood
Faculty of Computer Science, Dalhousie University, Halifax, NS, Canada
lcd@dal.ca, mheywood@cs.dal.ca, zincir@cs.dal.ca

ABSTRACT

In real world applications, variation in deployment environments, such as changes in data collection techniques, can affect the effectiveness and/or efficiency of machine learning (ML) systems. In this work, we investigate techniques to allow a previously trained population of Linear Genetic Programming (LGP) insider threat detectors to adapt to an expanded feature space. Experiments show that appropriate methods can be adopted to enable LGP to incorporate the new data efficiently, hence reducing computation requirements and expediting deployment under the new conditions.

CCS CONCEPTS

• **Security and privacy → Intrusion/anomaly detection and malware mitigation**; • **Theory of computation →** *Genetic programming*;

KEYWORDS

Insider threat detection, cyber-security, dynamic environment

ACM Reference Format:
Duc C. Le, Malcolm I. Heywood, Nur Zincir-Heywood. 2019. Benchmarking Genetic Programming in Dynamic Insider Threat Detection. In *Genetic and Evolutionary Computation Conference Companion (GECCO '19 Companion), July 13–17, 2019, Prague, Czech Republic.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3319619.3322029

1 BACKGROUND

Insider threats represent cyber-security problems in which a wide range of malicious activities are performed from "inside" the organization. Examples include, data exfiltration, intellectual property theft, and information system sabotage. Moreover, the dynamics of a corporate environment introduce additional challenges to insider threat detection. Specifically, it is not possible to assume a permanent deployment environment, thus changes may emerge in the application environment(s) and data collection / processing procedure(s). Of particular interest in this work is the case of new data sources, e.g new sensor types or new monitoring system, that provide additional or completely new information, which can be translated post-processing to new features.

GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6748-6/19/07…$15.00
https://doi.org/10.1145/3319619.3322029

In the literature, evolutionary computation based approaches have been proposed for dealing with different types of dynamics in deployment environments. Particularly in the case of streaming data, where challenges include data non-stationary, class imbalance and a limited label budget, GP has been successfully applied by using suitable sampling and archiving policies [3, 4]. In [2], the effectiveness of ML methods, including GP, is examined for botnet detection under botnet evolution conditions. Recently, a technique is proposed in [6] for outlier detection in feature-evolving data streams based on streaming random projection and ensemble of half-space chains.

In this work, we explore methods to allow a previously trained LGP population to evolve on an expanded feature space, which not only requires learning from additional features, but also maintaining previous performance. This is examined through the use of multiple releases of a publicly available dataset on insider threat.

2 PROBLEM STATEMENT AND PROPOSALS

Problem Statement. As mentioned above, we are interested in operating under conditions where changes in the deployment environment may cause the number of underlying features to change over time. To formalize the problem, considering at one point in time, we have a classifier $\mathscr{C}$ working on a data with feature set $\mathcal{F}$ to output to a set of categories C. Then changes in environment and data collection create a different incoming data stream with feature space $\mathcal{F}_1$, where $\mathcal{F} \subset \mathcal{F}_1$. To avoid training a completely new classifier to accommodate the changes, the challenge is in evolving $\mathscr{C}$ to $\mathscr{C}'$ working in the new environment, or learn from new introduced features, $\mathcal{F}_1 \setminus \mathcal{F}$.

Learning from an expanded feature space. In assuming a evolutionary based method for addressing the challenge, our underlying premise is that LGP can perform feature construction without fundamentally invalidating legacy solutions. In fact, LGP has two basic mechanisms for learning from expanded feature space: (i) it is a population based approach, where changes in data can be learned gradually through generations, and (ii) based on the use of input registers in LGP programs, feature space expansion can be accommodated by appropriately extending the input register vector. Though training generations with appropriate changes in variation operators, especially mutation, new features in $\mathcal{F}' \setminus \mathcal{F}$ can be incorporated into programs, hence allowing a previously trained population to evolve on an expanded feature space. We explore the following two elementary methods to re-purpose a previously trained classifier: (i) adjust, with the same prob., p, the variation operators to take into account new features in $\mathcal{F}' \setminus \mathcal{F}$ and (ii) adjusting the variation operators with a bias $(2 \times p)$ toward selecting a feature $f_1 \in \mathcal{F}_1 \setminus \mathcal{F}$ upon detecting feature space expansion. The bias is reduced gradually to original prob. after 50 generations.

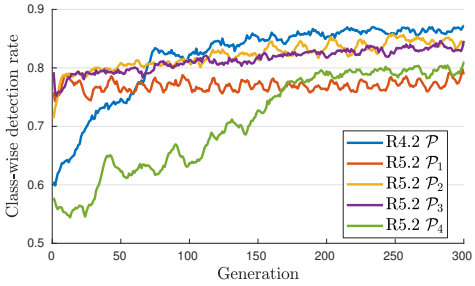

Figure 1: Class-wise detection rates of the populations on training subsets by number of generations

3 EXPERIMENT AND RESULTS

Experiment. This work employs a publicly available dataset for testing insider threat mitigation approaches, the CERT insider threat dataset [1]. Specifically, two releases, 4.2 and 5.2 of the CERT dataset, hereafter R4.2 and R5.2, are used. Details on data description, data processing, and feature extraction can be found in [5]. On the number of features, extracted data from R4.2 and R5.2 contains 107 and 190 features, respectively. For training and evaluating LGP, we use data from the first 50% duration and from a limited set of users (400) of each release for training, and the rest for testing. The setting is to reflect real-world application environments, where ground truth is limited. It is noteworthy that the data is heavily skewed, where the combined malicious data accounts for only 0.5% of training data, and 0.2% of testing data. Hence, multi-objective selection is employed to address two objectives simultaneously: maximize detection rate (over all classes) and maximize accuracy. This is done through the use of Pareto ranking. To measure the performance, detection rates (DR) of normal and malicious classes in a binary classification task are used. Moreover, to keep a low false positive rate, the best individual of LGP population post training is selected with at least 99% training accuracy in all experiments. Results are estimated by average of 5 runs.

In this experiment, a LGP population as previously trained on R4.2 is then evolved on R5.2 for 300 generations. Four different strategies are then investigated for understanding evolution under feature space expansion ($\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3, \mathcal{P}_4$). The first 3 populations, $\mathcal{P}_1, \mathcal{P}_2$, and $\mathcal{P}_3$, are evolved from the same original population $\mathcal{P}$ that was trained on R4.2. Population $\mathcal{P}_1$ continues evolving the population with the original (old) feature space $\mathcal{F}$. $\mathcal{P}_2$ and $\mathcal{P}_3$ are retrained on R5.2 full feature space $\mathcal{F}_1$, as described in the first and the second methods in § 2. Finally, a population $\mathcal{P}_4$ is trained from scratch on R5.2 as a baseline for comparison.

Initial results. The evolution of populations in terms of class-wise detection rate on R5.2 and R4.2 training subsets are illustrated in Figure 1. Table 1 presents test performance on R5.2 after 0, 20, 50, 100, and 300 generations. Based on Figure 1, it is clear that $\mathcal{P}_{1-3}$ was able to maintain the performance that $\mathcal{P}$ achieved on R4.2 and evolve from that to adapt to changes in R5.2. Initial results of $\mathcal{P}_{1-3}$ on R5.2 is better than results accomplished by $\mathcal{P}_4$ after 100 generations. From that initial advantage, $\mathcal{P}_{1-3}$ maintain clearly better results than $\mathcal{P}_4$ for at least 200 generations. By just 100 generations, or about 40% of R5.2 training data, populations $\mathcal{P}_2$ and $\mathcal{P}_3$ were able to obtain nearly 50% insider threat detection rate on R5.2 test data. Thus, in general, retraining $\mathcal{P}$ with adaptation to R5.2 was able to generate better results than a population trained from scratch on R5.2. On the three populations that evolve from $\mathcal{P}$, $\mathcal{P}_1$ gives worse results than $\mathcal{P}_{2-3}$. Given enough training time, $\mathcal{P}_4$ is also able to surpass $\mathcal{P}_1$. This indicates that simply continuing to train LGP on the old feature space $\mathcal{F}$, it is not possible to make use of the additional features, $\mathcal{F}_1 \setminus \mathcal{F}$, thus new threats go unnoticed.

Table 1: Test results of LGP populations trained over an increasing number of generations.

# generation	Population	Normal DR	Insider threat DR
0	$\mathcal{P}, \mathcal{P}_{1-3}$	98.96	22.35
20	$\mathcal{P}_1$	97.97	31.46
	$\mathcal{P}_2$	97.93	36.57
	$\mathcal{P}_3$	97.72	36.59
	$\mathcal{P}_4$	99.95	10.70
50	$\mathcal{P}_1$	97.98	35.79
	$\mathcal{P}_2$	97.98	42.14
	$\mathcal{P}_3$	98.10	38.40
	$\mathcal{P}_4$	99.61	14.69
100	$\mathcal{P}_1$	97.76	41.63
	$\mathcal{P}_2$	97.89	49.04
	$\mathcal{P}_3$	97.90	46.19
	$\mathcal{P}_4$	99.53	18.25
300	$\mathcal{P}_1$	97.46	45.26
	$\mathcal{P}_2$	97.68	54.99
	$\mathcal{P}_3$	97.86	52.43
	$\mathcal{P}_4$	98.08	52.85

Comparing results by $\mathcal{P}_2$ and $\mathcal{P}_3$, $\mathcal{P}_2$ gives slightly better results overall. On one hand, this may indicate that simply incorporating newly introduced features in normal LGP training mechanism is enough to learn from new features. On the other hand, other methods allowing GP to evolve on unseen features could be explored to improve the performance.

ACKNOWLEDGMENTS

This research is supported by Natural Science and Engineering Research Council of Canada, and Killam, Mitacs, and Nova Scotia graduate scholarships. The research is conducted as part of the Dalhousie NIMS Lab at: https://projects.cs.dal.ca/projectx/.

REFERENCES

[1] CERT and ExactData, LLC. 2016. Insider Threat Test Dataset. https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=508099. (2016).
[2] Fariba Haddadi and A. Nur Zincir-Heywood. 2015. Botnet Detection System Analysis on the Effect of Botnet Evolution and Feature Representation. In *ACM GECCO Companion '15*. 893–900.
[3] Sara Khanchi, Ali Vahdat, Malcolm I. Heywood, and A. Nur Zincir-Heywood. 2018. On botnet detection with genetic programming under streaming data label budgets and class imbalance. *Swarm and Evolutionary Computation* 39 (2018).
[4] Duc C. Le, Sara Khanchi, A. Nur Zincir-Heywood, and Malcolm I. Heywood. 2018. Benchmarking evolutionary computation approaches to insider threat detection. In *ACM GECCO '18*. 1286–1293.
[5] Duc C. Le and A. Nur Zincir-Heywood. 2019. Machine learning based Insider Threat Modellingand Detection. In *IFIP/IEEE International Symposium on Integrated Network Management*.
[6] Emaad Manzoor, Hemank Lamba, and Leman Akoglu. 2018. xStream: Outlier Detection in Feature-Evolving Data Streams. In *ACM SIGKDD '18*. 1963–1972.