Automated Pattern Identification and Classification: Anomaly Detection Case Study

[Extended Abstract]

R. G. Goss
University of Cape Town
Department of Computer Science
Cape Town, South Africa
ryan@goss.co.za

G. S. Nitschke
University of Cape Town
Department of Computer Science
Cape Town, South Africa
gnitschke@cs.uct.ac.za

CCS CONCEPTS

•Security and privacy → Artificial immune systems; •Computing methodologies → *Genetic algorithms;* **•Computer systems organization →** *Neural networks;*

KEYWORDS

anomaly detection, genetic algorithms, neural networks

ACM Reference format:
R. G. Goss and G. S. Nitschke. 2017. Automated Pattern Identification and Classification:
Anomaly Detection Case Study. In *Proceedings of GECCO '17 Companion, Berlin, Germany, July 15-19, 2017,* 2 pages.
DOI: http://dx.doi.org/10.1145/3067695.3076018

1 EXTENDED ABSTRACT

Protecting computer networks against unlawful intrusion, or otherwise malicious attack, is important. This task is increasingly relevant, especially as networks expand their reach by connecting to public networks, such as the Internet. This seemingly unregulated network provides a multitude of opportunities for cyber attackers, motivated by many factors, including revenge, prestige, politics and money. *Intrusion Detection Systems* (IDS) are tools designed to thwart these attacks, both pro-actively and reactively. These systems have become fundamental components of computer security architecture [7], identifying vulnerabilities and mitigating attacks against the networks they protect. Historically, these systems are often plagued by high rates of false alarms [12].

In this study, the efficacy of the *Automated Pattern Identification and Classification* (APIC) [5] *Machine Learning* (ML) pipeline method was evaluated as an *Anomaly Intrusion Detection* (AID) system to determine if using an ML-pipeline method could reduce false positive rates compared to similar methods using the same data set. This abstract first provides a brief overview of the APIC method, followed by the results of an experiment where the utility of the method was evaluated on a popular, publicly available IDS test data set. The experiment showed the method scored high degrees of accuracy, exhibiting fewer false positives than most methods.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
GECCO '17 Companion, Berlin, Germany
© 2017 Copyright held by the owner/author(s). 978-1-4503-4939-0/17/07…$15.00
DOI: http://dx.doi.org/10.1145/3067695.3076018

The APIC method is an efficient, adaptable, general method capable of identifying patterns in mixed, noisy data sets across a variety of domains [3–5]. These domains include *Internet Protocol* (IP) networking, where many classification tasks remain unsolved, or poorly solved with respect to completeness, accuracy, portability or false positive rates. The APIC method automatically discovers and creates near-optimal independent classifiers for identifying future instances of each pattern discovered in mixed, noisy data sets. The method provides high degrees of automation, incorporating feature selection (extraction), pattern identification and classifier development processes into a single pipeline method.

The APIC method processes data sets comprised of n-dimensional, numerical feature vectors representing data for each problem. Feature vectors with increased dimensionality are inherently more task intensive and incur additional overheads when training ML algorithms. Using *Evolutionary Algorithms* (EA) [1], the APIC method identifies the best feature subsets for each problem, removing redundant or irrelevant features, such that the accuracy achieved by models trained using the subset equals or surpasses that of the original data set. The APIC method adopts a similar approach to that of Huang et al.[6] and Vafaie et al. [13], encoding feature sets as bit-strings, where individual features are toggled on and off by a *Genetic Algorithm* (GA) [1], in search of the most optimal subset. A low dimensional feature subset is encouraged by applying a penalty to the fitness score of each genotype, proportional to its dimensionality. The final fitness score is determined by $f_{score} = f_{score} * 1/|genotype|$, where each genotype's fitness score is multiplied by the magnitude of its genes, divided into one. This ensures those with lower dimensionality are more favourable than those with higher cardinality, promoting the reduction of unnecessary features, ultimately reducing task complexity.

The APIC method clusters the data set, tuning hyper-parameters using a GA. The clusters discovered by the best scoring clustered data set form labelled training sets for the production of supervised or semi-supervised classifiers. The method allows the number of clusters to be set explicitly, or to be inferred automatically through an automated discovery process. Where the number of patterns, or *targets*, are not specified, the method uses the *Density Based Spatial Clustering of Applications with Noise* (DBSCAN) [2] algorithm to discover them automatically. If the number of targets, k, are known *a priori*, the k-means [8] algorithm is used to partition the datum into k clusters.

A unique classifier is produced for each cluster discovered by the search process. Automatic discovery and annotation of training

Method	AR	DR	FPR
NBC [15]	82.80%	13.80%	17.6%
KMC+NBC [15]	99.80%	95.40%	0.13%
RF [7]	99.99%	99.81%	0.01%
XM-RF [7]	99.99%	99.95%	0.00%
APIC	99.97%	99.92%	0.004%

Method	AR	DR	FPR
TANN [12]	96.91%	98.95%	3.83%
KM-1R [9]	99.26%	99.33%	2.73%
NBC [15]	88.20%	85.00%	33.70%
KMC+NBC [15]	99.00%	98.80%	2.2%
XM-1R [7]	93.68%	95.20%	9.26%
RF [7]	99.91%	99.94%	0.11%
XM-RF [7]	99.96%	99.99%	0.02%
APIC	99.91%	99.90%	0.08%

Table 1: The *Accuracy Rate* (AR), *Detection Rate* (DR) and *False Positive Rates* (FPR) rate achieved by each method.

sets in this manner removes dependency on human intervention, resulting in increased completeness and accuracy of the system. The APIC method produces a customised *Topology and Weight Evolving Artificial Neural Network* (TWEANN) [14] classifier for each pattern discovered. Using a semi-supervised training process, the design of each classifier is optimised using GAs and the *Backward Propagation of Errors* (back-propagation) algorithm [10].

The best scoring TWEANN classifiers for each pattern are gathered for each feature subset and evaluated against predetermined criterion. In this experiment, the average recall accuracy achieved by the classifiers should exceed 0.75, for both the training and test datasets. If the classifiers meet or exceed the defined minima, they are accepted and exit the training process. A failure to reach adequate approval triggers further generations of the feature subset selection GA, where the adaptation process is repeated until the success criterion is met, or another stop condition is triggered.

The APIC method was tested as a victim-end *Network Based Intrusion Detection* (NBAD) system, using the pre-labelled *Information Security Centre of Excellence* (ISCX) 2012 IDS data set [11][1]. The experiments followed the design set forth by Yassin et al. [15] and Juma et al. [7], where a subset of the ISCX data set was used to evaluate each respective method. The training and testing data sets comprised 77,526 and 56,421 flow samples respectively, containing summary information for service transactions describing web (*Hypertext Transfer Protocol* (HTTP)), E-Mail (*Post Office Protocol* (POP)/*Simple Mail Transport Protocol* (SMTP)), *Domain Name Services* (DNS) and *Secure Shell* (SSH) flows. The accuracy, detection and false positive rates achieved by the APIC method and comparable methods, for this task, are outlined in table 1.

Results in table 1 demonstrate that the APIC method achieves lower false positive rates for all but one of the comparable methods, designed specifically as a victim-end NBAD system. Contrary to the KMC+NBC and XM-RF methods, proposed by Yassin et al. [15] and Juma et al. [7] respectively, the APIC method is a general method, applicable to a broad range classification tasks. It scores well without encoding domain-specific knowledge, or customization to support specific tasks. Using GA-controlled, hyper-parameter optimization processes, the method is capable of identifying distinct groups of data (clusters) and modelling classifiers more accurately for each problem than methods using a heuristic approach guided by human experts. To date, specially designed NBAD systems are required to detect anomalous traffic in these data sets. The results of this work, however, indicate that ML-pipeline methods, such as APIC, yield comparable task performance to manually-designed NBAD methods, exhibiting lower false positive rates than these comparable methods. The application of automated ML-pipeline methods for anomaly detection in various tasks provides exciting opportunities for future research.

[1]http://www.unb.ca/research/iscx/dataset/

REFERENCES

[1] T. Bäck, D. Fogel, and Z. Michalewicz. 1997. Handbook of Evolutionary Computation. (1997).

[2] M. Ester, H. Kriegel, J. Sander, and X. Xu. 1996. A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Knowledge Discovery in Databases*, Vol. 96. 226–231.

[3] R. Goss and G. Nitschke. 2013. Automated Network Application Classification: A Competitive Learning Approach. In *In Proceedings of the IEEE Symposium Series on Computational Intelligence (IEEE SSCI 2013)*. 45 – 52.

[4] R. Goss and G. Nitschke. 2013. Network Protocol Identification Ensemble with EA Optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO2013)*. 1735–1736.

[5] R. Goss and G. Nitschke. 2014. Automating Network Protocol Identification. In *Case Studies in Intelligent Computing: Achievements and Trends*, B. Issac and N. Israr (Eds.). CRC Press, Cleveland, Ohio.

[6] C. Huang and C. Wang. 2006. A GA-based Feature Selection and Parameters Optimization for Support Vector Machines. *Expert Systems with applications* 31, 2 (2006), 231–240.

[7] S. Juma, Z. Muda, and W. Yassin. 2014. Reducing False Alarm Using Hybrid Intrusion Detection Based on X-Means Clustering and Random Forest Classification. *Journal of Theoretical & Applied Information Technology* 68, 2 (2014).

[8] J. MacQueen and others. 1967. Some Methods for Classification and Analysis of Multivariate Observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Vol. 1. California, USA, 14.

[9] Z. Muda, W. Yassin, M. Sulaiman, and N. Udzir. 2011. Intrusion Detection Based on K-means Clustering and OneR Classification. In *Information Assurance and Security (IAS), 2011 7th International Conference on.* IEEE, 192–197.

[10] D. Rumelhart, G. Hinton, and R. Williams. 1988. *Learning Representations by Back-propagating Errors.* MIT Press, Cambridge, MA, USA.

[11] A. Shiravi, H. Shiravi, M. Tavallaee, and A. Ghorbani. 2012. Toward Developing a Systematic Approach to Generate Benchmark Datasets for Intrusion Detection. *Computers & Security* 31, 3 (2012), 357–374.

[12] C. Tsai and C. Lin. 2010. A Triangle Area Based Nearest Neighbors Approach to Intrusion Detection. *Pattern Recognition* 43, 1 (2010), 222–229.

[13] H. Vafaie and K. De Jong. 1992. Genetic Algorithms as a Tool for Feature Selection in Machine Learning. In *Tools with Artificial Intelligence, 1992. TAI'92, Proceedings., Fourth International Conference on.* IEEE, 200–203.

[14] X. Yao. 1999. Evolving Artificial Neural Networks. *Proc. IEEE* 87, 9 (1999), 1423–1447.

[15] W. Yassin, N. Udzir, Z. Muda, and M. Sulaiman. 2013. Anomaly-Based Intrusion Detection through K-Means Clustering and Naives Bayes Classification. In *Proceedings of the 4th International Conference on Computing and Informatics (ICOCI), Sarawak, Malaysia.* 298–303.