Lecture Notes in Electrical Engineering 603

Rayner Alfred
Yuto Lim
Haviluddin Haviluddin
Chin Kim On *Editors*

Computational Science and Technology

6th ICCST 2019, Kota Kinabalu, Malaysia, 29–30 August 2019

Springer

Lecture Notes in Electrical Engineering

Volume 603

The book series *Lecture Notes in Electrical Engineering* (LNEE) publishes the latest developments in Electrical Engineering - quickly, informally and in high quality. While original research reported in proceedings and monographs has traditionally formed the core of LNEE, we also encourage authors to submit books devoted to supporting student education and professional training in the various fields and applications areas of electrical engineering. The series cover classical and emerging topics concerning:

- Communication Engineering, Information Theory and Networks
- Electronics Engineering and Microelectronics
- Signal, Image and Speech Processing
- Wireless and Mobile Communication
- Circuits and Systems
- Energy Systems, Power Electronics and Electrical Machines
- Electro-optical Engineering
- Instrumentation Engineering
- Avionics Engineering
- Control Systems
- Internet-of-Things and Cybersecurity
- Biomedical Devices, MEMS and NEMS

For general information about this book series, comments or suggestions, please contact leontina.dicecco@springer.com.

To submit a proposal or request further information, please contact the Publishing Editor in your country:

China
Jasmine Dou, Associate Editor (jasmine.dou@springer.com)

India
Swati Meherishi, Executive Editor (swati.meherishi@springer.com)
Aninda Bose, Senior Editor (aninda.bose@springer.com)

Japan
Takeyuki Yonezawa, Editorial Director (takeyuki.yonezawa@springer.com)

South Korea
Smith (Ahram) Chae, Editor (smith.chae@springer.com)

Southeast Asia
Ramesh Nath Premnath, Editor (ramesh.premnath@springer.com)

USA, Canada:
Michael Luby, Senior Editor (michael.luby@springer.com)

All other Countries:
Leontina Di Cecco, Senior Editor (leontina.dicecco@springer.com)
Christoph Baumann, Executive Editor (christoph.baumann@springer.com)

**** Indexing: The books of this series are submitted to ISI Proceedings, EI-Compendex, SCOPUS, MetaPress, Web of Science and Springerlink ****

More information about this series at http://www.springer.com/series/7818

Rayner Alfred · Yuto Lim ·
Haviluddin Haviluddin ·
Chin Kim On
Editors

Computational Science and Technology

6th ICCST 2019, Kota Kinabalu, Malaysia,
29–30 August 2019

Springer

Editors
Rayner Alfred
Knowledge Technology Research Unit,
Faculty of Computing and Informatics
Universiti Malaysia Sabah
Kota Kinabalu, Sabah, Malaysia

Haviluddin Haviluddin
Department of Computer Science
Universitas Mulawarman
Samarinda, Indonesia

Yuto Lim
School of Information Science,
Security and Networks Area
Japan Advanced Institute of Science
and Technology
Nomi, Ishikawa, Japan

Chin Kim On
Center of Data and Information Management
Universiti Malaysia Sabah
Kota Kinabalu, Sabah, Malaysia

ISSN 1876-1100 ISSN 1876-1119 (electronic)
Lecture Notes in Electrical Engineering
ISBN 978-981-15-0057-2 ISBN 978-981-15-0058-9 (eBook)
https://doi.org/10.1007/978-981-15-0058-9

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd.
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

Preface

Computational Science and Technology is a rapidly growing multi- and interdisciplinary field that uses advanced computing and data analysis to understand and solve complex problems. The absolute size of many challenges in computational science and technology demands the use of supercomputing, parallel processing, sophisticated algorithms and advanced system software and architecture.

With the recent developments in open-standard hardware and software, augmented reality (AR) and virtual reality (VR), automation, humanized big data, machine learning, internet of everything (IoT) and smart home technology, web-scale information technology, mobility and physical-digital integrations, new and efficient solutions are required in Computational Science and Technology in order to fulfilled the demands from these developments.

The conference was organized and hosted by Spatial Explorer and jointly organized with JAIST and Mulawarman Universiti, Indonesia. Building on the previous FIVE conferences that include Regional Conference on Computational Science and Technology (RCSST 2007), the four International Conference on Computational Science and Technology (ICCST2014, ICCST2016, ICCST2017 and ICCST2018), the Sixth International Conference on Computational Science and Technology 2019 (ICCST2019) offers practitioners and researchers from academia and industry the possibility to share computational techniques and solutions in this area, to identify new issues, and to shape future directions for research, as well as to enable industrial users to apply leading-edge large-scale high-performance computational methods.

This volume presents a theory and practice of ongoing research in computational science and technology. The focuses of this volume is on a broad range of methodological approaches and empirical references points including artificial intelligence, cloud computing, communication and data networks, computational intelligence, data mining and data warehousing, evolutionary computing, high-performance computing, information retrieval, knowledge discovery, knowledge management, machine learning, modeling and simulations, parallel and distributed computing, problem-solving environments, semantic technology, soft computing, system-on-chip design and engineering, text mining, visualization and web-based and service computing. The carefully selected contributions to this

volume were initially accepted for oral presentation during the Sixth International Conference on Computational Science and Technology 2019 (ICCST2019) which is an international scientific conference for research in the field of advanced computational science and technology, that was held during 29–30 August 2019, at Hilton Kota Kinabalu, Sabah, Malaysia. The level of contributions corresponds to that of advanced scientific works, although several of them could be addressed also to non-expert readers. The volume brings together 69 chapters.

In concluding, we would also like to express our deep gratitude and appreciation to all the program committee members, panel reviewers, organizing committees and volunteers for your efforts to make this conference a successful event. It is worth emphasizing that much theoretical and empirical work remains to be done. It is encouraging to find that more research on computational science and technology is still required. We sincerely hope the readers will find this book interesting, useful and informative and it will give then a valuable inspiration for original and innovative research.

Kota Kinabalu, Malaysia — Rayner Alfred
Nomi, Japan — Yuto Lim
Samarinda, Indonesia — Haviluddin Haviluddin
Kota Kinabalu, Malaysia — Chin Kim On

The Sixth International Conference on Computational Science and Technology 2019 (ISSN: 1876-1100)

Keynote Speakers

Prof. Dr. Kukjin Chun—Department of Electrical and Computer Engineering, Seoul National University, Seoul, South Korea

Prof. Dr. Rosni Abdullah—School of Computer Sciences, Universiti Sains Malaysia, Pulau Pinang, Malaysia

Dr. Leandro Soares Indrusiak—Reader in Real-Time Systems, Department of Computer Science, The University of York, UK

Dr. Suresh Manandhar—Reader in Natural Language Processing, Department of Computer Science, The University of York, UK

Contents

Decision Tree with Sensitive Pruning in Network-based Intrusion Detection System

Yee Jian Chew[1], Shih Yin Ooi[1], Kok-Seng Wong[2], and Ying Han Pang[1]

[1] Faculty of Information Science and Technology, Multimedia University, Melaka, Malaysia
[2] Department of Computer Science, Nazarbayev University, Astana, Kazakhstan
chewyeejian@gmail.com, {yhpang,syooi}@mmu.edu.my
kokseng.wong@nu.edu.kz

Abstract. Machine learning techniques have been extensively adopted in the domain of Network-based Intrusion Detection System (NIDS) especially in the task of network traffics classification. A decision tree model with its kinship terminology is very suitable in this application. The merit of its straightforward and simple "*if-else"* rules makes the interpretation of network traffics easier. Despite its powerful classification and interpretation capacities, the visibility of its tree rules is introducing a new privacy risk to NIDS where it reveals the network posture of the owner. In this paper, we propose a sensitive pruning-based decision tree to tackle the privacy issues in this domain. The proposed pruning algorithm is modified based on C4.8 decision tree (better known as J48 in Weka package). The proposed model is tested with the 6 percent GureKDDCup NIDS dataset.

Keywords: Network-based Intrusion Detection System (NIDS); Decision Tree; Weka J48; Sensitive Pruning; Privacy; GureKDDCup

1 Introduction

Trends of coordinated network attacks are rapidly emerging each day with the advancement of technology. To combat the attacks, intrusion detection system (IDS) has been specifically used for flagging the malicious activity and policy violation. Host-based IDS is used to monitor the misuses of an individual user in a single host [1, 4]; while NIDS is used to flag the malicious network traffics [9]. With the proliferation of artificial intelligence, many machine learning techniques are incorporated with NIDS to detect the anomalous traffic patterns automatically.

Decision tree is one of the notable machine learners used in this domain. It was first introduced as ID3 by Quinlan [13], and many variations have been fostered. It is favoured in this domain because the transparency of its decision rules makes the interpretation possible by network administrator. Unlike other works who focus on improving the classification performances, we take a different approach in this work where a pruning method is proposed by considering a privacy preserving mechanism. Each pruning takes place when the splitting attribute disclosing some sensitive values. The sensitive values can be pre-determined by network administrator. In this work, the pro-

R. Alfred et al. (eds.), *Computational Science and Technology*,
Lecture Notes in Electrical Engineering 603,
https://doi.org/10.1007/978-981-15-0058-9_1

posed pruning framework is appended to Weka J48 decision tree [6]. This newly proposed pruned J48 is then tested on the 6 percent version of GureKDDCup IDS dataset [12].

2 Related Work

Many machine learning algorithms have been successfully deployed in this domain. In view of the proposed method in this paper is appended on a decision tree, the literature studies will primarily focus on the various modifications done on decision trees despite of all different kinds of machine learning techniques.

Depren et al. [5] proposed a hybrid NIDS by combining self-organising map as the anomaly detection module and J48 decision tree as the misuse detection module. In order to utilise the results of both modules, a decision support system is employed in such a way that if either one of the modules detects the traffic as an attack, the traffic will be automatically classified as an attack. Testing on 10 percent of KDDCup'99 dataset, the work reported with a detection rate (true positive rate) of 99.9%, missed rate (false negative rate) of 0.1% and classification accuracy of 99.84% respectively. In their work, only 6 basic features from each connection are utilised [16].

Bouzida et al. [2] proposed an improved version of C4.5 decision tree by modifying the post pruned rules. Generally, it classifies the training instances which do not match the decision tree rules into the default class (highest frequency class). In the case of NIDS, the highest proportion of a class would probably be the "normal" class. Thus, the training instances which do not conform to the decision tree model will be directly classified into "normal" class. To improve the detection against unknown or first-seen attacks, the authors modified the default class of a decision tree model to be "new" class. Thus, the enhanced decision tree model will assign the training instances which do not fit the rules of a decision tree to the "new" class instead of the highest frequency class ("normal"). With this method, the successful rates of 67.98% in detecting low-frequency attacks for user-to-root (U2R) was attained. Classification accuracy of 92.30% with standard deviation of 0.57% was reported. The authors adopted the entire 10 percent KDDCup'99 [16] without any feature extraction.

Xiang et al. [18] proposed a multiple-level hybrid classifier by leveraging the C4.5 decision tree (supervised) and Bayesian clustering (unsupervised) to detect malicious network traffics. In the first level, C4.5 is used to segregate the traffics into 3 primary classes: denial-of-service (DoS), Probe and "others". "Others" class consisted all instances from normal, U2R and remote-to-local (R2L) class. To segregate the normal traffics from U2R and R2L in this "others" class, Bayesian clustering is adopted. Subsequently, C4.5 is employed again on top of Bayesian clustering to further separate the U2R attacks from R2L attacks. This multiple-level classification module shown the positive sign in reducing the false negative rate to 3.25% while maintaining an acceptable level of false positive rate of 3.2%. With the combined usage of C4.5 and Bayesian clustering, the overall detection rate obtained were 99.19% (DoS), 99.71% (Probe), 66.67% (U2R), 89.14% (R2L) and 96.8% (normal) when testing on 10 percent KDDCup'99 [16].

Kim et al. [10] proposed a hybrid NIDS with C4.5 decision tree (misuse detection) and support vector machines (SVM) (anomaly detection). Initially, a C4.5 decision tree training model is built based on a full set of training data. Avail with the decision tree rules, all leaves which have the "normal" label are further classified with a 1-class SVM. Then, the non-matching traffics will be flagged as "unknown attack". This model greatly improved the detection rate for "unknown attack" (by approximately ~10%) comparing to the conventional techniques when tested on NSL-KDD [17] with the condition of false positive rate below 10% under ROC curve.

Rai et al. [14] modified the standard C4.5 split values by considering the average value of all instances in the selected attribute. The proposed splitting algorithm is able to reduce the time taken for building a model because the process for sorting the attribute values are no longer needed. By selecting the 16 attributes with the highest information gain from NSL-KDD [17], the work reported with the best accuracy of 79.5245%.

Cataltepe et al. [3] proposed a semi supervised anomaly NIDS by integrating online clustering, online feature selection and decision tree. Instead of labelling the instances with human efforts, authors adopted Clustream (a type of online clustering) to group the similar instances together and label each of them. Lastly, the classification is based on decision tree. Detection rate (recall) of 98.38% and precision rate of 96.28% were reported when tested on the 10 percent KDDCup'99 [16].

Goeschel et al. [8] proposed a hybrid classifier encompassing SVM, J48 decision tree, and Naïve Bayes classifiers with the aim of reducing false positives in NIDS. The hybrid model works in 3 phases. In phase 1, binary SVM is adopted to classify the network traffics into "attack" or "normal". If the predicted traffics are classified as "normal", the traffics or instances will not be further processed by the next two classifiers. Following the similar procedure as phase 1, J48 decision tree is employed in phase 2 while Naïve Bayes is utilised in phase 3. Classification accuracy of 99.62% and false positive rate of 1.57% were attainted with this approach.

Though many are working on improving the detection rates for NIDS with decision trees, but none of them are considering the privacy property when adopting the decision tree in NIDS. The issue of privacy related to NIDS, especially IP addresses should not be neglected ever since the IP addresses were perceived as part of personal data under the Court of Justice of the European Union [7]. Thus, we propose a pruning model to integrate into decision tree, where all sensitive information from NIDS will be "pruned" and obscured. In specific, this paper will focus on camouflaging the sensitive IP addresses with the proposed pruning method.

3 Proposed Model

3.1 Sensitive Pruning

To obscure the sensitive information from the network traffic, we proposed a modified pruning method in decision tree. This pruning method is modified based on the existing Weka J48 decision tree algorithm, which is a Weka version of C4.8 decision tree algorithm. Predominantly, the proposed sensitive pruning method is conducted based on the

sensitive IP addresses which are determined by the user (i.e. network engineer who is familiar to the organisation network architecture). Whenever an IP address is selected as the splitting attribute, it will be skimmed through for reviewing whether the value collides with the predetermined sensitive IP addresses.

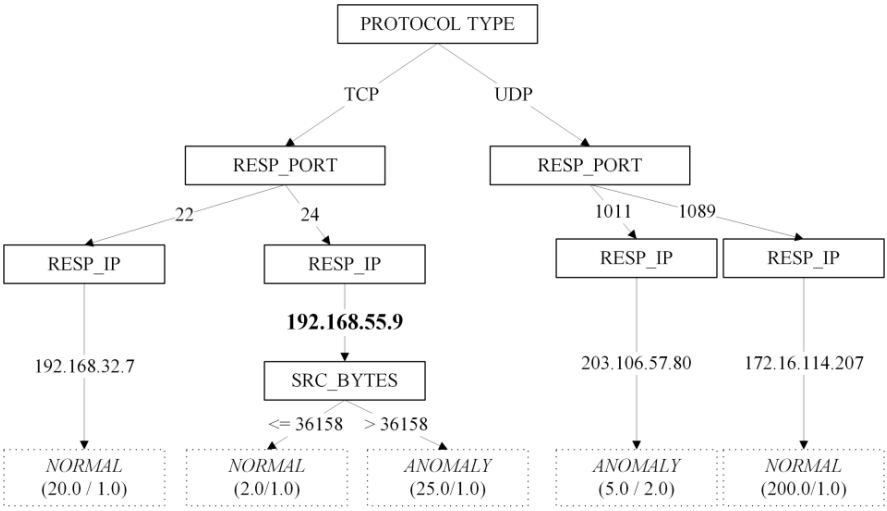

Fig. 1. A model of unpruned decision tree

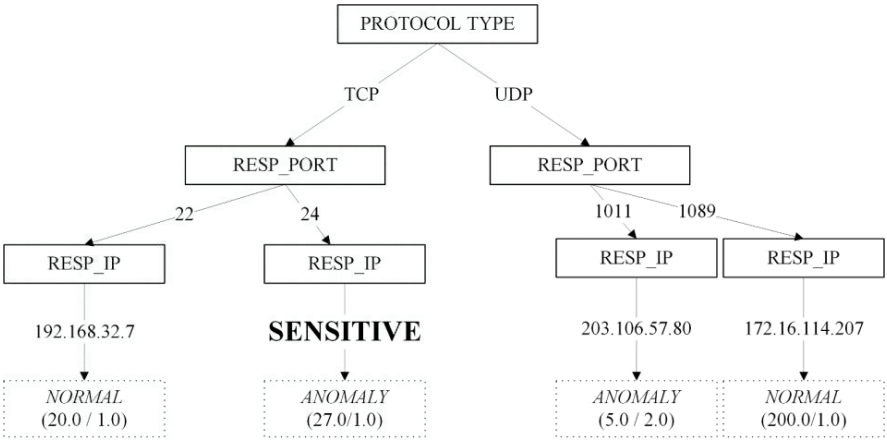

Fig. 2. A model of decision tree after Sensitive Pruning

Referring to the example depicted in Figure 1, assume that 192.168.55.9 is pre-set as one of the sensitive IP addresses, this value will be replaced by the word of "SENSITIVE", thus obscuring its real IP address. Subsequently, all of its child nodes will be pruned into a leaf node as this sensitive branch has been obscured and can no longer being utilised to improve the performance of the model, as depicted in Figure 2. If the sensitive branch is leading to a leaf node, the leaf node is remained as it is without any modification. The sensitive pruning algorithm can be summarised as below:

Sensitive Pruning Algorithm

Input: T_{max} , Unpruned Decision Tree;
Sensitive IP Addresses, A_S

Output: Pruned Sensitive Decision Tree

Procedure:

```
for all node t in T_max, do
Let node t_s be the child node of t
Let Attribute Value A equals to current split IP Addresses attribute value
if (A == A_S):
  Replace A value with "SENSITIVE"
  if (t_s contains any child node):
    Replace t_s with a leaf node
  endif
endif
endif
```

3.2 IP Address Selection and IP Comparison Optimisation

In this proposed pruning algorithm, we supposed the sensitive IP addresses should be determined by the user or network engineer who is familiar to the organisation network architecture itself. Due to the lack of human experts in our case, we take all private IP addresses as the sensitive IP addresses in this paper. It is reasonable to have this assumption because all private IP addresses are part of the organisation, and most of them can reveal a lot of sensitive information [15], including the employees who used the machine, services running on the hosts, activity logs, etc.

Table 1. Private IP Addresses Range

Private IP Addresses Range									No. of Hosts
10.	0.	0.	0	~	10.	255.	255.	255	16,777,216
172.	16.	0.	0	~	172.	31.	255.	255	1,048,576
192.	168.	0.	0	~	192.	168.	255.	255	65,536

According to Table 1, the total number of private IP addresses approximately reaches 17.8 million. By inputting all of them into the pruning algorithm, the required computation cost and complexity will escalate tremendously to compare each of them during

the iteration of pruning procedure. Therefore, an optimised approach is designed to handle such big amount of private IP addresses. Specifically, each IP address is split into 4 subsets in accordance to the 4 octets available in IPv4. Then, only the first two octets of the IP address will be utilised to determine if it is a private IP address. As a result, instead of comparing 17.8 million IP address, we only have to compare 18 of them. The optimise version for comparing private IP addresses is described as below:

Private IP Addresses Comparison Algorithm

Input: *IP* , A Single value of IP Address;

Output: *True*, If IP Address is Private;

False, If IP Address is not Private

Procedure:

```
1.  Split IP into 4 subsets based on dot "." delimiter.
2.  Let IP[1] be the first octet,
    IP[2] be the second  octet,
    IP[3] be the third octet and
    IP[4] be the fourth octet.
3.  if (IP[1] == 10):
4.     return True
5.  elseif (IP[1]=="192" AND IP[2]=="168":
6.     return True
7.  elseif (IP[1]=="192" AND IP[2]="16~31":
8.     return True
9.  else: return False
```

4 Experiment

4.1 Experiment Settings

In this paper, it is important to retrieve a pair of IP address because it is one of the required entities for our proposed pruning as described in Section 3. Thus, GureKDDcup [12] is adopted in this study instead of using the classical IDS dataset such as DARPA'98 [11], KDDCup'99 [16] or NSL-KDD [17]. It was generated according to the same process as KDDCup'99 but it additionally includes each pair of IP addresses. As denoted by the creator of GureKDDcup [12], the full dataset of this dataset is too big to be used in learning process. Thus, the reduced sample of 6 percent GureKDDcup (*gureKddcup6percent.arff*) dataset is employed in this paper. All experiments are evaluated in 10 fold cross-validation. To have a fair comparison with other classifiers, no feature extraction process is conducted. Hence, all of the 47 attributes and 28 class labels in the 6 percent GureKDDCup are not normalized or removed. Additionally, unpruned J48 (*J48U*), pruned J48 (*J48P*), unpruned J48 with sensitive pruning (*J48U-SP*), as well as pruned J48 with sensitive pruning (*J48P-SP*) are tested empirically.

4.2 GureKDDCup dataset

In Table 2, the name and type of the 47 attributes found in GureKDDCup dataset are tabulated. The metadata are extracted directly from the Attribute-Relation File Format – *gureKDDCup6percent.arff* shared by the dataset creator. As highlighted previously in Section 4.1, all the attributes are not modified or altered to provide a fair judgement between the classifiers.

Table 2. Attributes of GureKDDCup Dataset (6 percent)

Attributes	Type	Attributes	Type
connection_number	Num	num_access_files	Num
start_time	Num	num_outbound_cmds	Num
orig_port	Num	is_hot_login	*Bin*
resp_port	Num	is_guest_login	*Bin*
orig_ip	*Nom*	count	Num
resp_ip	*Nom*	srv_count	Num
duration	Num	serror_rate	Num
protocol_type	*Nom*	srv_serror_rate	Num
service	*Nom*	rerror_rate	Num
flag	*Nom*	srv_rerror_rate	Num
src_bytes	Num	same_srv_rate	Num
dst_bytes	Num	diff_srv_rate	Num
land	*Bin*	srv_diff_host_rate	Num
wrong_fragment	Num	dst_host_count	Num
urgent	Num	dst_host_srv_count	Num
hot	Num	dst_host_same_srv_rate	Num
num_failed_logins	Num	dst_host_diff_srv_rate	Num
logged_in	*Bin*	dst_host_same_src_port_rate	Num
num_compromised	Num	dst_host_srv_diff_host_rate	Num
root_shell	*Bin*	dst_host_serror_rate	Num
su_attempted	Num	dst_host_srv_serror_rate	Num
num_root	Num	dst_host_rerror_rate	Num
num_file_creations	Num	dst_host_srv_error_rate	Num
num_shells	Num		

Num: numeric **Nom**: nominal **Bin**: binary

4.3 Experimental Results

In Table 3, the best classification accuracy is attained by the standard unpruned J48 (*J48U*) and pruned J48 (*J48P*). Although our proposed pruning algorithm (*J48U-SP* and *J48P-SP*) degrades the performance approximately by 0.61%, the loss is considered tolerable as it is able to preserve a certain level of privacy and reduces the complexity of the tree at the same time. Referring to Table 4, the number of final leaves is reduced

by 6355 leaves when adopting the unpruned J48 with sensitive pruning (*J48U-SP*) as compared to the unpruned J48 (*J48U*). In general, a smaller decision tree model with satisfactory performance are preferred in contrast to a larger tree as it is easier to understand and interpret the rules. We also observed the number of prune leaves and nodes does not differ abundantly between *J48P* and *J48P-SP*, the scenario can be justified by the order of pruning operation. For the *J48P-SP*, the default J48 pruning procedure (*J48P*) are performed before the proposed sensitive pruning, and these results depicts that most of the sensitive IP values have already been trim away by the default J48 Pruning (*J48P*) ahead of the proposed sensitive pruning.

Table 3. Experimental Results (Classification Accuracy)

Prune Type	Correct Classified Instances	Accuracy (%)	Accuracy Reduction (%)
J48U	178717	99.9480	0.0000
J48P	178710	99.9441	-0.0039
J48U-SP	177626	**99.3378**	**-0.6105**
J48P-SP	177612	**99.3300**	**-0.6183**

Accuracy Reduced = Accuracy (J48U) – Accuracy (J48P / J48U-SP / J48P-SP) * 100 (-) Decrease in Accuracy

Table 4. Experimental Results (Number of Leaves and Nodes)

Prune Type	Number of Prune Leaves	Number of Prune Nodes	Number of Final Leaves	Number of Final Nodes
J48U	0	0	35805	35876
J48P	357	12393	23448	23483
J48U-SP	**6355**	**6368**	29450	29508
J48P-SP	**12621**	**12667**	23184	23209

***Number of prune leaves and prune nodes includes all nodes pruned by sensitive pruning**

4.4 Performance Comparison

To corroborate the robustness of the proposed J48 Sensitive Pruning, performance of the proposed algorithm is compared against a set of benchmarks testing as tabulated in Table 5. Due to the lack of prior art, we compare this proposed technique to 8 notable classifiers from Weka package, including SVM (aka SMO in Weka), Naïve Bayes, Adaboost, Bayesian Network, Decision Stump, ZeroR, Random Tree, and Random Forest. All of them are tested and compared on the 6 percent of GureKDDCup dataset with the similar experimental setup as explained in Section 4.1. As can be seen in Table 5, the results obtained are very encouraging. Although *J48U-SP* and *J48P-SP* are

slightly underperforming when compared to the original *J48U* and *J48P*, but the privacy of sensitive rules is conceived. It is also important to note that they are still outperformed Bayesian Network, Adaboost, Decision Stump, ZeroR and Naïve Bayes.

Table 5. Benchmark Comparison against other algorithms from Weka Package

Algorithm	Accuracy (%)
SVM (better known as Weka SMO)	99.96
Random Forest	99.96
J48U	99.95
J48P	99.94
Random Tree	99.91
J48U-SP	**99.34**
J48P-SP	**99.33**
Bayesian Network	98.80
Adaboost	98.08
Decision Stump	98.08
ZeroR	97.80
NaiveBayes	83.71

5 Conclusion and Future Works

In this paper, a sensitive based pruning decision tree is proposed. Through the experimental testing on the 6 percent GureKDDCup dataset, the promising evaluation results spells two advantages: (1) ability to preserve privacy in a fully built decision tree by masking only sensitive values selected, (2) minimal changes on the decision tree structure since the proposed pruning algorithm does not affect the process of attribute selection during tree construction. As the current version of the proposed pruning are not suitable to be applied directly for other domain, further investigation can be conducted on the sensitive pruning to furnish it with flexibility and scalability.

Acknowledgements

This research work was supported by a Fundamental Research Grant Schemes (FRGS) under the Ministry of Education and Multimedia University, Malaysia (Project ID: MMUE/160029), and Korea Foundation of Advanced Studies (ISEF).

References

1. Anderson, J.P.: Computer security threat monitoring and surveillance. Tech. Rep. James P Anderson Co Fort Washingt. Pa. 56 (1980).
2. Bouzida, Y., Cuppens, F.: Neural networks vs. decision trees for intrusion detection. Commun. 2006. ICC '06. IEEE Int. Conf. 2394–2400 (2006).
3. Cataltepe, Z. et al.: Online feature selected semi-supervised decision trees for network intrusion detection. Proc. NOMS 2016 - 2016 IEEE/IFIP Netw. Oper. Manag. Symp. AnNet, 1085–1088 (2016).
4. Denning, D.E.: An Intrusion-Detection Model. IEEE Trans. Softw. Eng. SE-13, 2, 222–232 (1987).
5. Depren, O. et al.: An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks. Expert Syst. Appl. 29, 4, 713–722 (2005).
6. Frank, E. et al.: The WEKA Workbench. Morgan Kaufmann, Fourth Ed. 553–571 (2016).
7. Frederik, Z.B.: Behavioral Targeting: A European Legal Perspective. IEEE Secur. Priv. 11, 82–85 (2013).
8. Goeschel, K.: Reducing false positives in intrusion detection systems using data-mining techniques utilizing support vector machines, decision trees, and naive Bayes for off-line analysis. Conf. Proc. - IEEE SOUTHEASTCON. 2016–July, (2016).
9. Heberlein, L.T. et al.: A network security monitor. In: Proceedings. 1990 IEEE Computer Society Symposium on Research in Security and Privacy. pp. 296–304 IEEE (1990).
10. Kim, G. et al.: A novel hybrid intrusion detection method integrating anomaly detection with misuse detection. Expert Syst. Appl. 41, 4 PART 2, 1690–1700 (2014).
11. Lippmann, R.P. et al.: Evaluating intrusion detection systems: the 1998 DARPA off-line intrusion detection evaluation. In: Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00. pp. 12–26 IEEE Comput. Soc.
12. Perona, I. et al.: Generation of the database gurekddcup. (2016).
13. Quinlan, J.R.: Induction of Decision Trees. Mach. Learn. 1, 1, 81–106 (1986).
14. Rai, K. et al.: Decision Tree Based Algorithm for Intrusion Detection. Int. J. Adv. Netw. Appl. 07, 04, 2828–2834 (2016).
15. Riboni, D. et al.: Obfuscation of sensitive data in network flows. IEEE Conf. Comput. Commun. INFOCOM 2012. 23, 2, 2372–2380 (2015).
16. Stolfo, S.J. et al.: Cost-based modeling for fraud and intrusion detection: results from the JAM project. In: Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00. pp. 130–144 IEEE Comput. Soc (2000).
17. Tavallaee, M. et al.: A detailed analysis of the KDD CUP 99 data set. IEEE Symp. Comput. Intell. Secur. Def. Appl. CISDA 2009. Cisda, 1–6 (2009).
18. Xiang, C. et al.: Design of multiple-level hybrid classifier for intrusion detection system using Bayesian clustering and decision trees. Pattern Recognit. Lett. 29, 7, 918–924 (2008).

A Sequential Approach to Network Intrusion Detection

Nicholas Lee, Shih Yin Ooi and Ying Han Pang

Faculty of Information Science & Technology,
Multimedia University, Malacca, Malaysia
lmz.nicholas@gmail.com, {syooi, yhpang}@mmu.edu.my

Abstract. In this paper, we combine the sequential modeling capability of Recurrent Neural Network (RNN), and the robustness of Random Forest (RF) in detecting network intrusions. Past events are modelled by RNN, capturing informative and sequential properties for the classifier. With the new output vectors being incorporated into the input features, RF is exacted to consider high-level sequential representation when selecting the best candidate to split. The proposed approach is tested and compared on the UNSW-NB15 data set, demonstrating its competence with encouraging results, and achieving an optimal trade-off between detection and false positive rate.

Keywords: Network-based Intrusion Detection System, Machine Learning, Random Forest, Recurrent Neural Network.

1 Introduction

An intrusion is any set of actions intended to compromise the confidentiality, integrity, or availability of a resource [1]. Network intrusions are prevalent, increasingly sophisticated, and are adept at hiding from detection [2]. To counteract this ever-evolving threat, Network-based Intrusion Detection System (NIDS) has since become a significant topic of research. IDS generally faces limited tolerance on the number of misclassifications [3]. Being unable to detect traces of intrusion could lead to alarming consequences, while having too much false positives undermines the efforts of investigation, rendering an alarm of IDS inconsequential [4].

In this paper, we present an approach in detecting network intrusions through the use of machine learning techniques. In addition to the connection features extracted from the traffic, our approach also factors in a series of past events to classify an observation. Through a trained RNN: Long Short-Term Memory (LSTM) specifically, high-level features can be retrieved as a representation of the past observations. Since Random Forest (RF) is robust in handling high-dimensional data, the learned representation can be combined to allow improved predictive qualities. Our proposed methodology: RNN-RF has shown its competitive performance on UNSW-NB15 data set, further evident by the comparative results of various existing techniques.

The rest of this paper is organized as follows: Section 2 presents the literature study of recent techniques employed for NIDS. The proposed approach is detailed in Section

R. Alfred et al. (eds.), *Computational Science and Technology*,
Lecture Notes in Electrical Engineering 603,
https://doi.org/10.1007/978-981-15-0058-9_2

3. Section 4 discusses on the particulars of data set, implementation, and the experimental results. Finally, the concluding remarks of the study are provided in Section 5.

2 Related Works

In this section, literature study of relevant works and their methodologies are presented. UNSW-NB15 data set is adopted in this study due to it containing modernized network traffic and attacks. Thus, we focus on studies which employed UNSW-NB15 and is by no means exhaustive.

Bamakan et al. [5] proposed Ramp Loss K-Support Vector Classification-Regression (Ramp-KSVCR) under the premise of multi-class classification. Utilizing the proposed methodology, the authors have tackled the issues of imbalanced attacks' distribution, as well as the susceptibility of Support Vector Machine (SVM) to outliers. Improved performance over multi-classification and skewed data set is achieved by adopting K-SVCR as the core of the authors' proposed methodology. Furthermore, the latter issue is solved by implementing Ramp loss function, in which its non-convex property allows for the desired robustness.

Papamartzivanos et al. [6] combined the strengths of both Decision Tree (DT) and Genetic Algorithm (GA) in generating intrusion detection rules. The proposed methodology – Dendron – aims to create linguistically interpretable, yet effective rules when dealing with the detection of attacks and to their corresponding categories. To further increase Dendron's effectiveness over the minority classes, a weighted selection probability function is devised to aid in evolving DT classifiers.

By means of anomaly-based approach, Moustafa et al. [7] introduced Beta Mixture Model (BMM-ADS) to tackle the complexity of intrusion detection. BMM is used to establish normal profile from selected 8 features of legitimate observations. In order to determine the dichotomy of normal and anomaly records, lower-upper Interquartile Range (IQR) baseline is also applied. Any observations outside the baseline are regarded as anomalies, potentially allowing the detection of zero-day attacks.

AL-Hawawreh et al. [8] proposed a model for Internet Industrial Control Systems (IICSs) by employing Deep Autoencoder (DAE) and Deep Feedforward Neural Network (DFFNN). To allow for better convergence properties, initialization parameters are obtained by pre-training DAE on normal traffic instances. Thereafter, pre-trained weights are used in initializing DFFNN before supervised training take place. Emphasis is also given on the importance of IDS components placement in IICS setting.

Yang et al. [9] developed a hybrid technique using Modified Density Peak Clustering Algorithm and Deep Belief Network (MDPCA-DBN). The authors modified DPCA by adopting Gaussian kernel function, this enables the clustering of a more complex and linearly inseparable data set. Each DBN classifier is then independently trained on the now complexity-reduced subset, efficiently extracting abstract features without any heuristic rules.

In view of the recent works, where the trained model makes a prediction independently of previously seen instances, we therefore explore the problem differently by considering the preceding observations. Our approach exploited the modeling

capability of LSTM network in order to extract the representation of previous instances. The acquired output vectors are then additionally incorporated to provide high-level features for the RF classifier.

3 Proposed Methodology

3.1 Random Forest

Random Forest (RF) is an ensemble learning technique developed by Breiman [10], incorporating together several novel ideas from previous studies. RF makes a prediction by combining the votes of multiple decision trees. Each decision tree in RF acts as a classifier and contributes to the overall votes. To grow an individual tree:

1. Let N be the total number of instances from train set, create a bootstrap sample from N instances.
2. m sub-features are randomly selected out of M input features, adhering to the condition where $m \ll M$.
3. Best candidate from the selected m features is chosen to split into daughter nodes.
4. Tree is fully grown, without the need of pruning.

Due to its training procedures, it is more robust to noises and overfitting when compared to its AdaBoost counterpart. Besides, various empirical studies have also shown the performances of RF that rivals some of the popular state-of-the-art methods.

3.2 Recurrent Neural Network

To capture insightful features from a sequence of events, a network specialized in modeling sequential data is required. Hence, a notable architecture of RNN is adopted: Long Short-Term Memory (LSTM). LSTM network is originally proposed by Hochreiter et al. [11] and further improved by Gers et al. [12]. Since then, it has been consistently exploited and found itself useful in many diverse applications.

The forward pass of an LSTM cell is defined as follows:

$$h_t = \tanh(c_t) \odot o_t \tag{1}$$

$$c_t = c_{t-1} \odot f_t + z_t \odot i_t \tag{2}$$

$$z_t = \tanh(W^z x_t + R^z h_{t-1} + b^z) \tag{3}$$

$$f_t = \sigma(W^f x_t + R^f h_{t-1} + b^f) \tag{4}$$

$$i_t = \sigma(W^i x_t + R^i h_{t-1} + b^i) \tag{5}$$

$$o_t = \sigma(W^o x_t + R^o h_{t-1} + b^o) \tag{6}$$

Where x_t is the input, and h_t is the output of LSTM cell at time step *t*. Input weights, recurrent weights, and biases are denoted by W, R, and b respectively. The information

flow in an LSTM cell is regulated by forget f, input i, and output o gating units. Essentially, it is the component of state unit c that allows past information to be transferred over a long distance, and conveniently discard it when necessary.

3.3 Proposed RNN-RF

The training phase of the proposed methodology consists of two stages. During the first stage, LSTM is trained with a supervised criterion, learning a good representation for the fully-connected layer. On top of being able to account for the previous traffic events, it also eliminates the need for manual feature engineering. The network is also trained in a many-to-one manner. Having a look-back window of length s, it seeks to predict the normality of data at current timestep.

The latter part of training phase begins once LSTM has completed its training. Since the network is now able to yield properties that make a classification task easier, new representation of the data can be obtained by taking the output vectors from each layer of the trained model. Finally, the output vectors are concatenated altogether with the original features, serving as the input data for training RF classifier. Original data with n-dimensional features, will consequently have a representation $x' \in \mathbb{R}^{n+u\cdot\ell}$, where u and ℓ denote the number of LSTM hidden units and layers respectively.

In the testing phase, data are passed through the trained model in the same manner. The newly represented test set is then used by RF classifier to make prediction choices. A summarization of the proposed methodology is illustrated in Fig. 1.

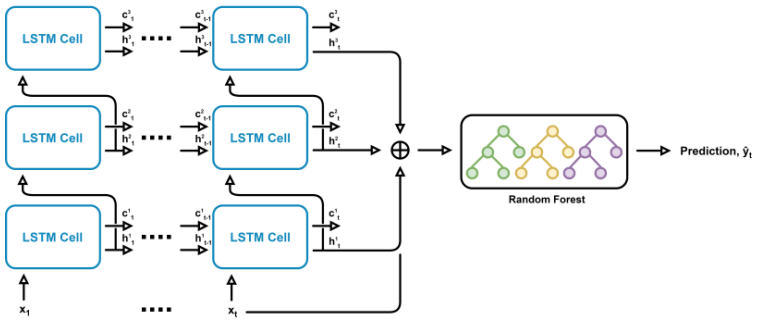

Fig. 1. Architectural summary of the proposed methodology

4 Experimental Setup

4.1 Data set

UNSW-NB15 is created by Moustafa et al. [13-14] in order to improve the complexity of NIDS data set, while addressing some of the major shortcomings found in KDDCup99 [15] and NSL-KDD [16]. Despite NSL-KDD being an improved derivative from KDDCup99 [17], it still carries over few issues found in its preceding counterpart.

Moreover, NSL-KDD does not represent modern low footprint attacks, as is being inherent in KDDCup99 [13,18].

To create UNSW-NB15 data set, the authors deployed IXIA PerfectStorm tool in generating both realistic normal and malicious network traffic. Different techniques were utilized for the extraction of features, capturing a total of 2,540,044 instances in comma-separated values (CSV) format. A partitioned version of the data set is also created by the authors, containing both train and test set, further aiding the evaluation of NIDS.

As hyperparameter optimization and early stopping will be carried out, a validation set is necessary. By observing validation error, it allows for fair selection of best hyperparameter, and hereafter avoiding the pitfall of overfitting to the test set [19]. In this study, validation set is created by extracting 10% of the instances (17,534) from the original train set, while preserving its sequential order to the best extent possible. However, to account for all attack types available, half of 10% had to be split at the leading train set, while the other 5% is extracted at the end.

Table 1 shows the number of instances for each attack types, both before and after the validation split. In this study however, we focus on the detection of attacks, neglecting the concern of correctly classifying an attack type. Thus, C_1 will represent normal class, while C_2 will be the class indicating the presence of attack.

Table 1. Total instances of data set

Class	Original Data Set		Split Data Set		
	Train	Test	Train	Validation	Test
Normal	56,000	37,000	47,233	8,767	37,000
Analysis	2,000	677	1,946	54	677
Backdoor	1,746	583	1,705	41	583
DoS	12,264	4,089	11,965	299	4,089
Exploits	33,393	11,132	32,513	880	11,132
Fuzzers	18,184	6,062	17,662	522	6,062
Generic	40,000	18,871	33,356	6,644	18,871
Reconnaissance	10,491	3,496	10,197	294	3,496
Shellcode	1,133	378	1,102	31	378
Worms	130	44	128	2	44
Total	175,341	82,332	157,807	17,534	82,332

The data set contains a total of 42 features, with the exclusion of *id* and ground truths. Features are extracted from varying sources, and are comprised of both categorical and numerical (float, integer or binary) types.

Data Transformation. In order to better represent categorical data, 3 features: *protocol*, *service* and *state* are encoded using one-hot encoding scheme. A dimension is created for each new feature value found in the train set. The now extended dimensions