Available online at http://jurnal.ahmar.id/index.php/asci

Journal of Applied Science, Engineering, Technology, and Education

ISSN 2685-0591 (Online)

Journal of Applied Science, Engineering, Technology, and Education Vol. 2 No. 1 (2020)
https://doi.org/10.35877/454RI.asci2192

An Empirical Evaluation On Comparative Machine Learning Techniques For Detection Of The Distributed Denial Of Service (DDoS) Attacks

Arnold Adimabua Ojugo[a*], Andrew Okonji Eboka[b]

[a]*Department of Computer Science, Federal University of Petroleum Resources, Effurun 32001, Delta State, Nigeria, ojugo.arnold@fupre.edu.ng*
[b]*Department of Computer Science Edu., Federal College of Education (Technical), Asaba 32001, Delta State, Nigeria, ebokaandrew@gmail.com*

Abstract

The advent of the Internet that aided the efficient sharing of resources. Also, it has introduced adversaries whom are today restlessly in their continued efforts at an effective, non-detectable means to invade secure systems, either for fun or personal gains. They achieve these feats via the use of malware, which is both on the rise, wreaks havoc alongside causing loads of financial losses to users. With the upsurge to counter these escapades, users and businesses today seek means to detect these evolving behaviour and pattern by these adversaries. It is also to worthy of note that adversaries have also evolved, changing their own structure to make signature detection somewhat unreliable and anomaly detection tedious to network administrators. Our study investigates the detection of the distributed denial of service (DDoS) attacks using machine learning techniques. Results shows that though evolutionary models have been successfully implemented in the detection DDoS, the search for optima is an inconclusive and continuous task. That no one method yields a better optima than hybrids. That with hybrids, users must adequately resolve the issues of data conflicts arising from the dataset to be used, conflict from the adapted statistical methods arising from data encoding, and conflicts in parameter selection to avoid model overtraining, over-fitting and over-parameterization.

Keywords: Intrusion, DDoS, network performance, network resources, machine learning, malicious attacks

1. Introduction

The future has always been shaped, refocused and direction set accurse by science and technology. Thus, we see these as the inevitable driving force behind today's society, which also – is heavily dependent on digitally transmitted and processed data. This, is consequent upon the fact that individuals and organizations (inclusive) are seeking better and improve ways at becoming more effective and efficient at their daily (data) processing activities and tasks [1, 2, 3]. However, this situation is currently being menaced by adversaries trying to gain access to such data. Many studies have ensued and systems developed in a bid to ameliorate this menace – with researchers seeking ways and means/methods that will help dissuade these adversaries from such act. This, has led to explosion in the field of informatics, data security and forensics with cutting edge research into data mining. Even with frontier exploration

* Arnold Adimabua Ojugo

E-mail address: ojugo.arnold@fupre.edu.ng (First Author)

and implementation of security tools such as firewalls, application gateways etc – all of which, is geared to ensure data integrity and privacy. This, has since and now become a herculean task for many network administrators, as their adversaries have continued to evolve their intrusion techniques with the direction of technology. This implies that intrusion detection systems that seek to effectively monitor network traffic and thus, identify resource misuse, unauthorized use and abuse on networks must equally advance and evolve [4, 5].

1.1. The Nature of Intrusion and Intrusion Detection Systems

Intrusion (activities) is either initiated externally or internally [6, 7]- resulting in two (2) types of adversaries (or intruders) namely: (a) external adversaries that have unauthorized (or no) access to resources – but, attacks a network via means of penetration. They usually employ the use of malware; and (b) internal adversaries that resides on the said network and thus, have authorized network access to resources [8]. The implementation of security tools have their various demerits and bottlenecks – some of which hampers network performance and even in some cases, compromise the network security also. Thus, an intrusion detection system (IDS) seeks to bridge the gap between the existing network security technologies and provide a system that seeks to minimize the many risks in implementing security measures [9].

Intrusion is a set of action that seeks to compromise integrity, confidentiality, privacy and availability of network resources. An intruder (adversary) is any user(s) who initiates an intrusive action [10] on a network system to exploit its resources. Thus, IDS is designed to generate alert as it observes potentially malicious and abusive traffic. It monitors data packets from network connections and determines if it is an intrusive activity or not. If an intrusive action is detected, the IDS performs one of these: (a) logs a message into the system audit file to be later analysed by a security expert, (b) emails an alert to a network administrator, and (c) ends the adversary's connection (that is as provisioned under Intrusion Prevention System) amongst other functions [1, 11, 12]. Many dataset compilation have successfully classified network connections into genuine and intrusive connections. Intrusive connections are further classified into various attack types namely: (a) distributed denial of service (DoS), (b) user to root (U2R), (c) probe, (d) root to local (R2L) [9].

IDS has 3-main parts namely: (a) sensors/network probes which tracks data traffic, system behaviour and log files by translating data into events usable as the IDS monitors and taps in to access all network connections, (b) analysis console which takes sensor output as input in form of network connections, analyses it for intrusive acts (as critical component to decide whether or not, a connection is intrusive), and (c) policy control which generates reactions based on analysis' outcome. If analysis console flags a connection as intrusion, control performs several actions depending on policies, set by the network administrator [13, 14, 15, 16]. Such actions include logout of a particular connection, alerting the administrator via e-mail etc. It also handles action(s) to be taken when an intrusion is detected [17, 18, 19] as in Figure 1.

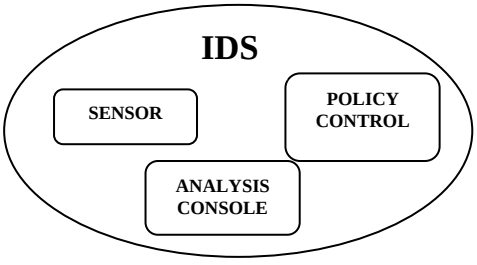

Figure 1. Framework of an IDS

1.2. Distributed Denial of Service (DDoS) Attack: Definition and Overview

DDoS is a large-scale, coordinated attack on provisioned services to network resource(s) or a victim system, launched indirectly via a number of compromised computers [20]. Prior an attack, an adversary holds up the resource of a large number of vulnerable machines under his control. He then exploits their weaknesses by inserting malicious code(s) using technique such as HTTP, SYN and UDP flooding – so that a server is overwhelmed by their service requests [21]. The magnitude of an attack depends on size of the botnet – so that the larger the botnet, the more severe and

disastrous such attack [22]. DDoS attacks aim at exhausting target resources and denying services to legitimate users. When detected, the problem is fixed by manually disconnecting the affected system from the network. DDoS attacks denies users access to network resources such as CPU power, bandwidth, memory, processing time. The goal of any DDoS detection scheme is to detect the attack as soon as possible and stop them as near as possible to their sources [4, 23].

DDoS uses two (2) methods: (a) flooding, and (b) protocol attacks – by sending large volume of malicious traffic to a server. Even with a number of proposed techniques to defeat DDoS, it is still hard to detect/respond to flooding DDoS due to the large number of attacking machines an adversary employs via source-address spoofing, and adoption of similar traffic packets between a legitimate and a malicious user traffic [24, 25]. DDoS detection schemes are grouped based on the locality of deployment as [22, 26, 27]:

a. Source-End Detection are deployed at an attack's source to prevent users from generating DDoS attacks. Here, the source devices identify a malicious packet in outgoing traffic, and filters the traffic. Detecting and stopping a DDoS attack at the source is the best possible defence as minimum damage is done on legitimate traffic.
b. Victim-end detection: Here, a victim system detects, filters malicious incoming traffic at a router (that is, networks providing Web services). The legitimate and attack traffic can clearly be distinguished from either online/offline via either the misuse-intrusion/anomaly-intrusion detection. However, attack traffic reaching the victim may denied or degraded services and bandwidth saturation.
c. Core-end or Intermediate router detection mechanism: In core-end or intermediate network detection scheme, any router in the network can independently attempt to identify the malicious traffic and filter or rate-limit the traffic. It balances trade-offs between detection accuracy and attack bandwidth consumption. Trace-back of attack sources becomes easy, due to collaborative operation. In this point of defence, the traffic is aggregated i.e., both attack and legitimate packets arrive at the router and it is a better place to rate-limit all the traffic.

1.3. Structure of The DDoS Attack(s)

Many companies face a lot of crisis due to intrusion. Many methods have been devised as secure means from such disasters via adoption of intrusion detection systems. In networks, some behaviour exists with an external event. The IDS helps discover malicious events via either the signature of the attack, or an anomaly of the data traffic. Thus, an IDS seeks to secure network resources and grant data confidentiality, integrity, and availability to users [27]. The IDS gathers data within a network, analyses it and detect cum ascertain intrusion affected components using a number of techniques. Such detection depends on three (3) aspects characterized by [4, 9] as seen in figure 2.

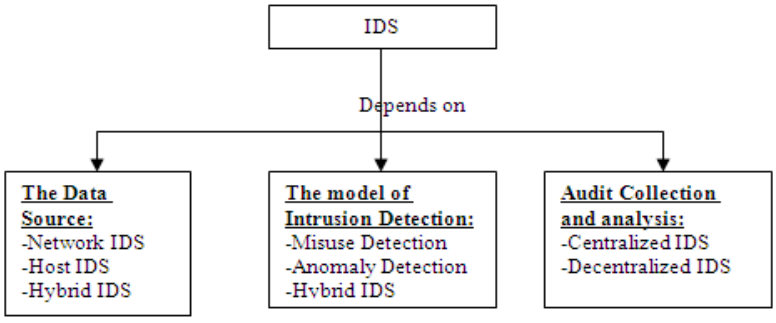

Figure. 2. Structural Architecture and classification of an IDS

1. Data Source – a network IDS examine the network traffic; while the host IDS examines the system components such as the operating system. Also, the hybrid IDS supports both sources of data.
2. The model of intrusion detection deals with misuse detection for which the signature detection mode seeks to verify the signature on data traffic; while, the anomaly detection seeks to verify the behavior of system. Again, the hybrid IDS monitors both the signature and anomaly detection modes.
3. The audit collection/analysis is done via two means: (a) a centralized IDS controlled by a central resource; and the decentralized IDS controlled from a local node with hierarchical reporting to one or more central location(s).

1.4. Motivation of The Study

The study is motivated by the following reasons [9, 27, 30, 31]:

1. The ever-changing intent of malware coders and makers keeps the study of intrusion consequently on the rise and inevitable. DDoS attacks have continued to soar at uneased rate with malicious activities guised to exploit users. They have caused great financial loss. To efficiently tackle malicious data is a continuous, 'inconclusive' task as many models are hampered in performance both by the technique used by adversaries, the selection of parameters and adopted model. These, can result in overfitting, over-training and over-parameterization.
2. With many of the model employing hill-climbing methods – their solutions often gets trapped at local maxima.
3. Independent testing consistently yields a 100% detection, irrespective of the heuristics employed. However, all models have an acceptable rate of false-positive and true-negative results, identifying some genuine connections as malicious and vice-versa.
4. Carefully diagnosis of network is often mundane and time-wasting. It also often yields inconclusive results for ***evolving*** signature strings – all of which will amount to increased false-positive and true-negative results.
5. DDoS attack prevents legitimate user's access to resources – consuming available resources such as bandwidths, memory and processor time by overloading a network with request and makes it hard for a server to respond to other users till a countermeasures are taken. Thus, the need to identify, manage and prevent this attack type.
6. The design of effective detection (prevention) scheme has continued to suffered setback(s) due to the inherent reason that malicious traffics by design, are poised by their architect to evade detection. Thus, the limited size of characters and available dataset continues to cripple its detection. Thus, the impediment in size of feature or parameters to be selected for training has sometimes also, led to poor learning of feats and classification.

To overcome shortfalls in the adoption of machine learning schemes to handle malicious traffics, we compare various machine learning techniques to reduce on traffic packets and enhance adequate classification.

2. Materials And Methods

2.1. Dataset And Its Encoding

A major problem is getting the right *(formatted)* dataset that correlates with the underlying feats to be predicted. For this study, we employ the Hochschule Coburg IDS datasets (CIDDS-2017), which is a set of labelled network flow data for anomaly-based traffic. The dataset is split into training (70%) and testing (30%). We adopt 8-of-the-15 parameters to adjust model weights and coefficients in minimizing errors as in table 1:

Table 1. Parameter List

Features	Format	Data Type
Source IP	a.b.c.d	Object
Source Port	Numeric	Integer
Destination IP	a.b.c.d	Object
Destination Port	Numeric	Float
Protocol	String	Object
Duration	H:M:S	Float
Packets	Numeric	Integer
Attack Name / Type	String	Object

Real-time data are generally incomplete, noisy, imprecise, ambiguous and inconsistent. Encoding helps modulate the original data unto the required format for effective processing that is easily understood by the model. Here, our pre-processing techniques used to encode the selected features will involve transformations applied to our data before feeding it to the algorithm. It is technique that seeks to convert the raw data into a clean dataset – so that whenever the data is gathered from different sources, it is collected in raw format not feasible for analysis. Thus, we employ the ***categorical*** data type in the Pandas Library – whose algorithm is therein displayed as listing 1.

Input: Selected Feature
Output: Converted Feature Data type
1. Select Feature
2. For each Selected Feature
3. If Selected Feature is Non-Numerical then
4. Generate Category Data type
5. End if
6. End For each

Listing 1: Algorithm to Convert Data type to Category

2.2. Rule Generated and Fitness Function Model

Each rule is a solution in space with attributes over a range as encoded via integer decimal with 57-rules. Internet Protocol (IPs) connection addresses are coded as decimal form for simplicity and qualitative representation [1]. Rules are then randomly initialized for selection via the fitness function (weighted sum model) to indicate significance of each feature adopted in the (each) rule template generator. If a connection request (source IP, destination IP, source port, destination port, and connection time) exists, then stop connection establishment. Thus, such IP is recognized as blacklisted by IDS and the service request initiated from it, is rejected. All rules are tested on historical connections, to filter new connections and find suspicious traffics. Source IP originates a connection (genuine/intrusion), Destination IP is target whose port shows running apps. Each If-Then has a condition and outcome part with 6-feats connected via a logical AND to form the **condition** part; while the attack name is the **outcome** to show learning ***classification*** (in training), or connection (at detection) if a rule is matched [1, 4]. An example attack is as follows:

IF (time="0:0:1" **AND** protocol="telnet" **AND** source port=89 **AND** destination port=23 **AND** source IP="9.9.9.9" **AND** destination IP="172.16.12.50") THEN (attack="port-scan"). If telnet/port-scan is represented by integers 1 and 2:
{0, 0, 1, 2, 18982, 79, 9, 9, 9, 9, 172, 16, 012, 50, 1}

Rules are evaluated via fitness function to determine its fit and goodness with detecting attacks. A good rule correctly classifies an attack; Else, it is bad. The support-confidence fitness is used. Changing weights w1/w2 detect intrusions (with w1=0 and w2=1); and precisely classify/detect intrusion cum behaviour anomalies (w1=1 and w2=0) with form:

If w1 = 0.2, w2 = 0.8, N = 10, |A| = 2, |A and B| = 1.
Support = | A & B | / N = 1 / 10 = 0.1:
Confidence = |A & B | / A = 1 /2 = 0.5
Fitness = w1 * support + w2 * confidence =0.42

Table 2 is audit data with sample rules that identifies attack. Rules in lines 3 and 6, to match rsh attack type.

Table 2. Fitness Function Value Framework

Time	Prot	Source Port	Destination Port	Source IP	Dest. IP	Attack
0.0.11	ftp	1892	21	192.168.1.30	192.168.1.20	-
0 .0 .0	Smtp	1900	25	192.168.1.30	192.168.1.20	-
0.0.2	Rsh	1023	102	192.168.1.30	192.168.1.20	Rcp
0.0.23	telnet	1906	23	192.168.1.30	192.168.1.20	Guess
0.0.14	rlogin	1022	513	192.168.1.30	192.168.1.20	rlogin
0.0.2	Rsh	1022	102	192.168.1.30	192.168.1.20	Rsh
0.0. 15	ftp	4354	21	192.168.1.40	192.168.1.20	-

2.3. Experimental Hybrid (Memetic) Genetic Algorithm Trained Neural Network

GA as inspired by Darwinian evolution consists of a dataset chosen for natural selection with potential solutions. Individuals with genes close to its optimal solution, is fit as determined by fitness function [11]. Based on laws of selection, GA generates better rules via 4-operators: initialization, selection, crossover and mutation. Cultural GA is a variants of GA with 4-beliefs: (a) Normative – specific range of values to which an individual is bound, (b) Domain has information about task, (c) Temporal has information about the search space available, and (d) spatial has topographical data about the task with time as a specific feat. In addition, CGA has an influence function that ensures

that individuals (altered or not) conforms to a pool that does not violate its belief space and reduces number of possible individuals generated till an optimum is found (Reynolds, 1994; Hassan and Crossley, 2004). GA's strength is in parallel traversing with solutions from randomly generated initial pool continuously evaluated via its fitness function [13].

For the hybrid memetic algorithm (GANN), we first initialize the artificial neural network (ANN) via its fitness function to select the new pool for the remainder operator(s) crossover and mutation as thus:

a. Crossover – adoptstournament selection (to maintain diversity) in chromosomes randomly chosen. With new offspring generated every iteration, a lesser number is chosen and continues till one is chosen as parents. The goal is not to create best rule (global optimum), but set of rules good enough to detect intrusion (many local maxima). Model chooses two-random cross-over points in chromosome (see table 3) between the parents, to yield two new children in lines 3 and 4 respectively.
b. Mutation: Each gene chromosome may (or not) change depending on probability of mutation rate. Mutation improves population diversity needed.

The generated rules are used to evaluate the remaining dataset. Its testing seeks to gather information of how well the rules created, can detect attacks. Two methods are used for testing namely: (a) use existing rules in rule-based IDS, and (b) build tailored rule IDS. The proposed design requires tailored rules created from traffic and fed back for detection. The rest dataset are used as incoming connection to see if generated rules can distinguish between normal and intrusive connections.

2.4. Experimental Profile Hidden Markov Model

The Hidden Markov Model is a double embedded chain that models complex stochastic processes as a chain of states with probabilities associated to each transition between states. In *n*-order Markov, its transition probabilities depend on ***current*** and ***n-1 previous*** states. A Hidden Markov model process determines the state generated for each state observation in a series (solution space or output sequence). For intrusion, an instruction not accepted by the trained HMM, yields high probability of being a intrusive connection [33]. Traditional HMM scores data through clustering based on the profile values. The probabilities of initial instruction set are checked to see if such are intrusive connections; while maintaining a log to reduce its true-negative (anomaly-like but genuine connection) and false-positive (unclassified connections). It thus, creates a profile of the connections, classified into low, medium and high profile ranges [34].

The profile HMM as a variant of HMM, which resolves the fundamental problems of HMM as thus: (a) it makes explicit use of positional (alignment) data contained in the observations/sequences, and (b) it allows null transitions, where necessary so that the model can match sequences that includes insertion and deletions. For intrusion detection, O is each code of metamorphic engine denoted as a rule, T is time each code takes to execute, N is number of codes in sequence and obfuscation methods used as etched into HMM, M is number of code access to registers contained in network, π is initial state of network and connection protocols for the various users, A is state transition probability matrix, a_{ij} is probability of a transition from state *i* to another state *j*, B contains *N*-probability distribution codes in knowledgebase from where profiles are been created (one code for each state of the process); while HMM $\lambda = (A,B,\pi)$. Though, parameters for HMM details are incomplete as above; But, the general idea is still intact. We can align multiple data rules as sequence with significant relations. Its output sequence determines if an unknown connection is related to sequence belonging to either genuine profile or not. We then use the profile HMM to score connection criterion and make decision [35]. The circles are ***delete*** state to detects classified DDoS connections in the knowledgebase, ***diamonds*** are insert states that allow us ***sandbox*** connections that are unclassified upon which the knowledgebase is updated for classified false-positive and true-negative errors; while ***rectangles*** are matched states that accurately classifies the various connections into DDoS, R2L, U2R and probe as in figure 4.

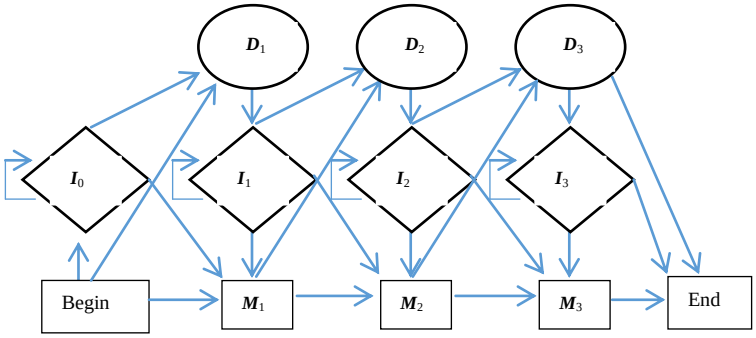

Figure. 3. PHMM with 3-Match States

Match and insert are emission states in which an observation is made as PHMM passes through all the states. Emission probabilities, corresponding to B in standard HMM model is computed based on frequency of symbols (in this case, connections) that can be emitted at a particular state in the model; But, are positional-dependent (in contrast to standard model). Also, the emission probabilities are derived from the Bayesian net, which represents our training phase. Finally, ***delete*** states allow the model to pass through the gaps in the Markov network to reach other emission states. These gaps are necessary to prevent the model from data over-fitting. We compute the probabilities using the forward algorithm, for each possible case recursively by reusing scores calculated for partial sequences [9, 33, 34].

3. Findings and Discussion of Findings

3.1. Model Evaluation

We use misclassification rate and corresponding improvement percentages for the adopted model(s) in comparison in both training and test data given by Equations 1 and 2; While, tables 2 and 3 yields summary of obtained values.

$$Misclassification\ Rate\ (MR) = \frac{No.\ of\ Incorrect\ class}{No.\ of\ Sample\ set} \quad (1)$$

$$Improvement\ Percentage = \frac{MR(A) - MR(B)}{MR(A)}\ x\ 100 \quad (2)$$

Table 3. Misclassification Rate of Each model

	Classification Errors	
Model	*Training Data*	*Testing Data*
PHMM	13.7%	10.2%
GANN	21.3%	19.7%
Benchmark Deep Neural Network	1.29%	1.09%

Table 4. Improvement Percentage

	Improvement %	
Model	*Training Data*	*Testing Data*
PHMM	56.03%	64.16%
GANN	42.79%	34.09%
Benchmark Deep Neural Network	75.89%	92.01%

Results in tables 3 and 4 respectively indicates that PHMM outperforms GANN – having a misclassification rate of 13.7% (false-positives and true-negatives error rate). Implying, it has a classification accuracy of about 87.3%; While, promising an improvement of about 56%. In contrast, the memetic algorithm (GANN) has a misclassification rate of 21.3% (false-positives and true-negatives error rate) and promises an improvement of 42.79%. Though, the PHMM outperforms the GANN, it underperforms against the benchmark model (Deep neural network) used as seen in table 3 and 4 respectively.

3.2. Discussion of Findings

Top rules have same fitness range [0.8, 0.8065] and are estimated 80% good to be used in detection – to imply the achievement of generating a bunch of good rules, rather than a single optimum rule – is better in intrusion detection. 10-out-of-22 rules have destination port as -1, so that the rules looks out for connections from any destination port. This increases the chances of detecting intrusion, improves the generality of rules, and provides for new attack types and its corresponding rules to be added to knowledgebase. The rule generator used a population of 400, w1 = 0.2, w2 = 0.8, 5000 epoch-evolutions and 0.05 probability of a gene to be mutated respectively.

After training and testing models, the results are thus:

a. PHMM was run 15 times (to eradicate biasness) and took 89seconds to find optima after 280 iterations (at best). It was able to generate each time, multiple local maxima (good rules) and its time varied between 102seconds and 23minutes. Convergence time depends on how close initial population is to solution and on mutation applied to individuals in the pool.
b. GANN was run 15 times, took 62seconds to reach optima after 319-iterations. It generated at intermittently multiple local maxima (good rules) and its time varied between 62-seconds and 40minutes. Convergence time depends on initial population, the temperature schedule applied and series of random walks applied to pool. It is to be noted that SA is most useful in the generation of best rule (and not set of better rules, goal of this study).

3.3. Rationale For Algorithms Of Choice

Most mathematical, machine learning models are inspired by evolution, biological and behavioural population. They search a space via hill-climbing method which is flexible, easily adapts to changing states and suited for real-time app to guarantee high global convergence in multimodal task. Initialized with random pool, it allocates increasing trials to regions of high fitness to find optima. Once a peak is found, model restarts with another randomly chosen, starting point. Its simplicity, well suited for dynamic feats of many local maxima – makes them appropriate. Each random trial is done in isolation and as search progresses, it allocates its trials evenly over the space and still evaluates as many points in regions found to be of low fitness as in regions found to be of high fitness. Its demerit is its inadequacy for linear model with small regions surrounded by low fitness – making such models, difficult to optimize.

3.4. Implementation Tradeoffs

Result trade-offs are as follows:

a. Result Presentation: Researchers often display flawed and unfounded results, to validate new/modified model rather than re-test limitations, insufficiency, biasness and inabilities of existing ones. This is because negative results are less valuable and most of such models aim to curb the non-linearity and dynamism in the phenomena they are predicting alongside discovering feats and underlying properties of the historic datasets used, to train, cross validate and test such models.
b. Efficiency – modellers (researchers) employ ‘confusing’ result figures to show how well their prediction is quite in tandem with observed values (even with limited dataset used in training the model). Some plots for predicted values are often not easily distinguishable – as such modellers do not provide numerical data to support their system’s claim (though their model is in good agreement with observed values). Some measure of goodness does not provide the relevant data.
c. Insufficient Testing: Validation compares observed on predicted values. Many studies suffer from inadequate dataset. If model aims to predict dynamic state, such ability should not be demonstrated with misleading results of limited dataset; and inconclusive and unclear contributions. Model must be adequately tested with methods laid bare so that process can be repeated to validate the usefulness and authenticity of such models.
d. Model validation is not an undertaking to be carried out by a researcher or research group; but rather, a scientific dialogue. Improper model applications and ambiguous results often impede such dialogue. This study aims to greatly minimize confusion in study of model as well as their corresponding implementation in IDS.

4. Summary, Recommendation and Conclusions

The comparative solutions employed a total of 56-rules for each solution approach. Top rules were found to have fitness range [0.8, 0.865] and estimated 80% good for classification of intrusion dataset. This implies that achieving a set of good rules – is much better than single optimum rule, which in turn is better for such clustered dataset. 22-of-56 rules have profile for candidate rules, which in turn increases chances of detecting intrusion and the generality of rules, providing the ability for new rules to be generated rules and added to the knowledgebase. The rule generator used a population of 400, w1 = 0.2, w2 = 0.8, 5000 epoch-evolutions and 0.05 probability of a gene to be mutated respectively. The consequences of DDoS attacks to users requires a concerted effort by all to detect intrusion. Detection schemes work by first scanning the request, analysing them in some way to decide which class (i.e. legitimate or otherwise) and then delivering to the intended module for further actions. Their performance therefore, can be measured by the number of false-positives (incorrectly marked) and false-negatives (unidentified data) that it generates. An ideal scheme will correctly classify all request and packets with almost zero error rates of false positive and true-negatives – through trade-offs between the number of false positives and false negatives.

References

[1] A.A. Ojugo., A. Eboka., E. Okonta., R. Yoro., F. Aghware., *Genetic algorithm rule-based intrusion detection system*, J. of Emerging Trends in Comp. Info. Sys., 2012. 3(8): pp1182-1194

[2] S.S. Kandeeban, R.S. Rajesh, *GA for framing rules for intrusion detection*, J. Comp. Sci and Security, 2007. 7(11), p285-290.

[3] R. Gong, M. Zulkernine, P. Abolmaesumi, *A software implementation of GA based approach to network intrusion detection*, 2005, www.cse.msu.edu/~cse848/Studentpapers/Tavon_Pourboghrat.pdf

[4] A.A. Ojugo., E. Ben-Iwhiwhu, O. Kekeje., M. Yerokun., I. Iyawah., *Malware propagation on social time varying networks: a comparative study of machine learning frameworks*, Int. J. of Modern Education Comp. Sci., 2014, 6(8): pp25-33, doi: 10.5815/ijmecs.2014.08.04, [web]: http://www.mecs-press.org/ijmecs/v6n8.html

[5] I.P. Okobah., A.A. Ojugo., *Evolutionary memetic models for malware intrusion detection: a comparative quest for computational solution and convergence,* IJCAOnline Int. J. Comp. Appl. 2018. 179(39): pp34-43

[6] S.S. Kandeeban, R.S. Rajesh, *Integrated intrusion detection system via soft computing*, J. Network Security, 2010. 10(2), p87

[7] A.A. Ojugo., A. Eboka., R. Yoro., M. Yerokun., F.N. Efozia., *Hybrid model for early diabetes diagnosis*, Mathematics and Computers in Science & Industry, 2015, 50, pp207-217, [web] *www.semanticscholar.org/paper/Hybrid-Model-for-Early-Diabetes-Diagnosis-Ojugo-Eboka/662ce32a1f353eca02391a4a0cfe684499ad4448*

[8] A.A. Ojugo., F.O. Aghware., R.E. Yoro., M.O. Yerokun., A.O. Eboka., C.N. Anujeonye., F. Efozia., *Evolutionary model for virus propagation on networks*, Automation, Control & Intelligent Systems, 2015, 3(4): pp56-62.

[9] A.A. Ojugo., R.E. Yoro., *Forging a machine learning intrusion detection model as tools to curb the distributed denial of service (DDoS) attacks*, Accepted publication in Int. J. Elect. & Comp. Engr., 2020. 10(1): pp126-132

[10] F. Olusegun, O.A. Oluwatobi, O. O. Adewale, *ID-SOMGA: self-organising migrating GA-based solution for Intrusion Detection*, Computer and Information Science, 2010. 3(4), p80

[11] M. Perez, T. Marwala, *Stochastic optimization for solving Sudoku*, Proc. of IEEE on Evol. Computing, 2011. p256 – 279.

[12] B. Shanmugam, N.B. Idris, *Hybrid intrusion detection systems using fuzzy logic*, 2011. www.intechopen.com/download/pdf/14361.

[13] P. Diaz-Gomez, D. Hougen, *Improved off-line intrusion detection using a GA*, 2005. cameron.edu/~pdiaz-go/Art_ICEIS.pdf

[14] A.A. Ojugo., A.O. Eboka., *An intelligent hunting profile for evolvable metamorphic malware*, African J. of Computing and ICT, 2015, 8(1-2): pp181 –190, [web]: www.afrjcict.net

[15] A.A. Ojugo., I.P. Okobah., *Hybrid fuzzy-genetic algorithm trained neural network model for diabetes diagnosis*, Digital Inno. & Contemporary Res. in Sci., Eng. & Tech., 2017, 5(4): pp69-90, [web]: ww.isteams.net/digital-innovations-journal

[16] A.A. Ojugo., D. Allenotor, *Text mining identification and detection using the exact string matching algorithm: a comparative analysis*, Digital Innovations & Contemporary Research In Science, Engineering & Technology, 2018, 6(1): pp169 – 180

[17] M. Perez, T. Marwala, *Microarray data feature selection using hybrid genetic algorithm simulated annealing*, IEEE conference on Electrical and Electronics Engineers, 2012, pp 1 – 5, doi: 10.1109/EEEI.2010.6377146

[18] Schafer, J.D., (1985): *Multiple objective optimization with vector evaluated Genetic Algorithm*, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.122.5689&rep=rep1&type=pdf.

[19] A.A. Ojugo., A. Eboka., R. Yoro., M. Yerokun., F. Efozia., *Framework design for statistical fraud detection*, Mathematics and Computers in Science & Industry, 2015, 50, pp176-182

[20] P.J. Criscuolo, *Distributed Denial of Service, Tribe Flood Network, and Stacheldraht CIAC-2319*. Department of Energy Computer Incident Advisory Capability (CIAC), 2010.

[21] H. Monowar, H. Bhuyan, D.K. Kashyap et al. *Detecting Distributed Denial of Service Attacks: Methods, Tools and Future Directions*. The Computer Journal, 2012, 3-19.
[22] P.K. Munivara, M. Rama, R.A. Mohan, R.K. Venugopal, *DoS and DDoS Attacks: Defense, Detection and Traceback - A Survey*, Global J. of Computer Science and Technology: E Network, Web & Security, 2014. 14 (7), 15-31
[23] S. Alexander, *An anomaly intrusion detection system based on intelligent user recognition.* 2012. Ph.D Thesis, University of Jyväskylä, Faculty of Information Technology, Finland
[24] K. Apoorv, *How to deal with IP addresses in Machine Learning algorithms.* 2016. Retrieved October 6, 2018, from Quora: www.quora.com/How-can-deal-with-IP-addresses-in-machine-learning-algorithms-in-traffic-analysis-and-anomaly-detection
[25] W. Eddy, *TCP SYN flooding Attacks and Common Mitigation*. 2017. Retrieved June 16, 2018, http://tools.ietf.org/html/rfc4987.
[26] P. Garcia-Teodoro, Diaz-Verdejo J., Macia-Fernandez G., E. Vazquez, *Anomaly-based network intrusion detection: techniques, systems and challenges*. 2012. Computers and Security, 28, 18-28
[27] A.A. Ojugo, T.O. Eduvie, *An anomaly-based intrusion detection system using the profile hidden markov chain model.* 2018, B.Sc Thesis, Department of Computer Sci, Federal University of Petroleum Resources Effurun, Nigeria.
[28] A.A. Ojugo., A.O. Eboka., *Memetic algorithm for short messaging service spam filter text normalization and semantic approach*, Int. J. of Info. & Comm. Tech., 9(1): pp13 – 27, doi: 10.11591/ijict.v9i1.pp9-18, [web]: http://ijict.iaescore.com/index.php/IJICT/article/view/20241
[29] A.A. Ojugo., A.O. Eboka., (2019). *Signature-based malware detection using approximate Boyer Moore string matching algorithm*, Int. J. of Math. Sciences and Computing, 3(5): pp49-62, doi: 10.5815/ijmsc.2019.03.05
[30] B. Lavender, *Implementation of GA into IDS and integration into nprobe*, 2010. Retrieved from [web] brie.com/brian/netga/Lavender_Report.pdf.
[31] W. Li, *GA approach to network IDS*, 2004, Retrieved from security.cse.msstate.edu/docs/Publication/wli/DOECSG2004.pdf
[32] T. Vollmer, J. Alves-Foss, M. Manic, *Autonomous rule creation for intrusion detection*, 2011, Retrieved from [web]: inl.gov/technicalpublications/Documents/5025964.pdf
[33] A.A. Ojugo, and A.O. Eboka, "Comparative Evaluation for High Intelligent Performance Adaptive Model for Spam Phishing Detection." *Digital Technologies*, vol. 3, no. 1 (2018): 915. doi: 10.12691/dt-3-1-2.
[34] A.A. Ojugo, and D. Otakore, "Improved Early Detection of Gestational Diabetes via Intelligent Classification Models: A Case of the Niger Delta Region in Nigeria." *Journal of Computer Sciences and Applications*, vol. 6, no. 2 (2018): 82-90. doi: 10.12691/jcsa-6-2-5.
[35] A.A. Ojugo & A.O. Eboka, *An Intelligent Hunting Profile for Evolvable Metamorphic Malware*. Afr J. of Comp & ICTs. 2015. Vol 7, No. 3. Pp 181-190,