ISSN 2278-3091
Volume 10, No.3, May - June 2021
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse841032021.pdf
https://doi.org/10.30534/ijatcse/2021/851032021

Deep Learning Network Anomaly-Based Intrusion Detection Ensemble For Predictive Intelligence To Curb Malicious Connections: An Empirical Evidence

Arnold Adimabua Ojugo[1], Elohor Ekurume[2]
[1]Department of Computer Science, Federal University of Petroleum Resources Effurun, Delta State, Nigeria
ojugo.arnold@fupre.edu.ng, arnoldojugo@gmail.com, maryarnoldojugo@gmail.com
[2]Department of Computer Science, Delta State University Abraka, Delta State, Nigeria. elohorogaga@gmail.com

ABSTRACT
The future is always been shaped and refocused via Sci-Tech. Info and communication technology – has continued to shape today's society as an inevitable driving force because we are now heavily dependent on digitally transmitted and processed data. This, is consequent upon the fact that individuals and organizations are seeking improve means to process data more effectively and efficiently. We thus, propose hybrid Genetic Algorithm trained Modular Neural Network to detect network anomaly cum malicious packets. GA was used due to its flexibility cum elitist mode. MNN is used as a learning paradigm for modular learning components. Model validation return a confusion matrix with these values: TP = 50, TN = 2, FN = 5, FP = 3. These values were subsequently applied to obtain sensitivity, specificity and accuracy of model. Model portrays a sensitivity value of 93%, specificity value of 25% and an accuracy value of 89%.

Key words: deep learning, anomaly-based networks, IDS, intelligent systems, genetic algorithm, memetic, hybrid

1. INTRODUCTION

Overtime, information has become a crucial element that aids many organizations and businesses for effective decision making. Thus, it determines their survival, navigational skills on the frontiers of service delivery, production efficiency, and effectiveness all-round in other respects. Information has thus, today become the bedrock of our society, as enhanced through the rich faculties and plethora of info and communication tech (ICT) supported-devices [1-2].

Information advanced as knowledge – has continued to foster resources development, growth and productivity [3-4] – and overtime, its availability has been eased with a plethora of ICT devices that allow for an efficient/effective dissemination over network medium. These medium has also from inception being besieged by threats and attacks of varying magnitude and veracity as promulgated by daredevil unauthorized users often referred to as adversaries cum attackers [5-8]. They often employ penetrative means to compromise a user's system, which in turn leads to compromised data integrity, availability, and confidentiality [9-12]. Cyber-attacks have indeed become the largest threat to global resources such as proprietary data, organization networks, and intellectual properties [13] and the extension cum expansion of ICT and technology in general has continued to heighten the fastidious intent of these adversaries as they seek to exploit user resources and data for personal and financial gains [14].

For many reasons, businesses now employ forensic systems to dissuade adversaries from unauthorized access to their data [15]. Although, these schemes had been successful earlier, the brazing efforts to reinvent the wheel has continued to further equip them with the means to evade detection [16-17]. This can simply be a function of the fact that most systems designed to keep such adversaries out are rippled with flaws that these adversaries also seek to exploit [18]. The fundamental issue thus, becomes that of accurate detection and prompt response cum measures designed as outcome in the event the system of detection of an adversary. These challenges and inherent dilemmas have thus, encouraged the adoption of software solutions for predicting network anomalies [19-20]. In recent times, the adoption and usage of software implementation has improved in its monitor capabilities [21-23].

Intrusion are action(s) of an unauthorized users who seeks to gain access to a network so as to compromise integrity, confidentiality, and availability of network resources. An intruder (or adversary) is any user(s) who initiates an intrusive action [2] on a network system to exploit its resources. Thus, an Intrusion Detection System (IDS) is modeled to generate an alert as it observes potentially malicious and abusive traffic. It monitors network connections and determines an intrusive activity. If an intrusive action is detected, the IDS performs one of these: (a) logs an audit data that is later analyzed, (b) alerts administrator, and (c) ends the connection (as measures for the IDS, amongst other functions [11, 12]. Many dataset have successfully classified connection observations into normal and intrusive connections. Intrusive connections are classified into various attack types: (a) denial of service (DoS), (b) user to root (U2R), (c) probe, (d) root to local (R2L) [1, 9].

An Intrusion Detection System (i.e. from a software view) is a component that track system anomalies and reports feedback via an audit trail with measures aimed at stopping the attack [24] so that such anomalies can be further investigated. The IDS monitors and report attacks based on predefined policies and procedures stipulated by the system. The categorization of IDS cut across environmental issues, usage and application, making its implementation cutting across a large range of networked systems [19].

2. LITERATURE REVIEW

Very often, intrusion are large-scale, coordinated attack on provisioned services to network resource(s) or victim system, launched indirectly via a number of compromised computers [25]. 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 [2]. The magnitude of an attack depends on size of the botnet – so that the larger the botnet, the more severe and disastrous such attack [21]. Intrusion attacks seek to exhaust 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 intrusion detection scheme is to detect the attack as soon as possible and stop them as near as possible to their sources [2-8].

2.1. Intrusion and Intrusive Activities

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 connections, analyses it for intrusive acts (as critical component to decide whether or not, a connection is intrusive), and (c) policy control generates measures based on outcome of an analysis. If analysis console flags a connection as intrusion, control met out actions based on policies, set by the network administrator [26-27]. 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 [1, 26] as in Figure 1.

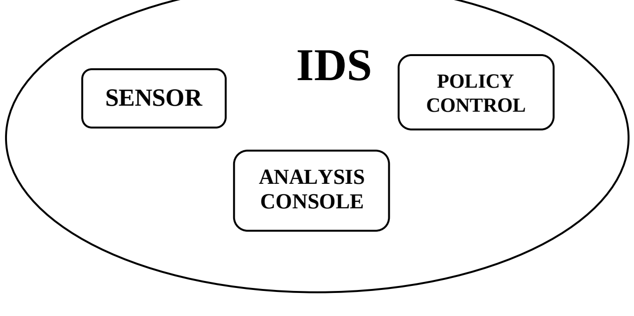

Figure 1. Generalized Framework of the IDS

Intrusion (activities) is either initiated externally or internally [1-2] resulting in two (2) types of adversaries: (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 in some cases, compromise the network security also [9].

2.2. Related Literature

Resources are often regarded as a stream of hardware, software and other events that can be checked on the backdrop of predefined attack rules. With socially-engineered threats and attacks, businesses, individuals and organizations now can formulate policies that can effectively and efficiently detect known attacks occurrences using either of the signature and/or anomaly evaluation schemes, self-organizing maps, and state transition analysis etc.

Thomas et al [28] used a NetBouncer to distinguish between uncompromised and compromised clients via its maintenance of a regularly updated client list that grants the requisite access to network resources. New clients are verified for legitimacy via their requests and data packets. Where a client passes the test, he is added to the legitimacy list and thus, granted access to resources until the window for legitimacy expires. With the list’s expiration, clients are revalidated.

Conversely, Gil and Poletto [29] used a Multi-Level Tree to monitor traffic feats. To detect attacks, the system aggregates packet statistics at various levels to successfully detect attacks via a disproportional difference between data rates in/out of a network. Ring et al. [30] Attackers can evade such detection mode by randomizing the source address IP for such malicious data. They presented a Host-Based IDS using a Blue-Box. The system specifies a set of rules (polices) to regulate resources access via an intuitive language to define security boundaries. The system was designed to minimize kernel changes and performance impact.

Akella et al. [31] used as Adaptive Neuro-Fuzzy Inference System (ANFIS) network-based IDS that sought to address the failure of data uncertainty, vagueness and imprecision in most predictive models. Their study was born-out of adaptation of model tools, target output, solution lacking fuzziness and also the nature of training. With ANFIS, their overall result was improved in detecting attacks. Munivara et al. [32] presented a novel anomaly-IDS via a hybrid behavioral model, Bayes net, recursive log-likelihood and entropy gain with unsupervised learning between current and reference behavior. Entropy and log-likelihood measures performance degradation. The model had improved classification of packets.

Santos et al. [28] sought detection and prevention with a rule-based network anomaly and state-wise protocol IDS. The system eliminated false and positive errors – creating rules and alert logs if a rule is violated. Chia-Mei et al. [3] employ the Hidden Markov Model to detect network anomaly emanating from detection errors and sensor input diversity. The study yields a sequence of attack corresponding to states attacks with potential to identify attacks efficiently. Nguyen [33] surveyed intrusion methods, attacks types, varied tools. His result was IDS with sensor (for packet and anomaly signature), backend (to record database), and frontend (for user interface, control and commands). The system addressed denial of services and synchronization.

Ojugo and Yoro [1] presented a deep learning approach to distinguish between benign exchange of data and malicious attacks from data traffic using anomaly-based IDS dataset. Result shows consequent success that effectively differentiates between genuine and malicious packets as well as detection of the Denial of service attacks.

2.3. Study Motivation

Study is motivated by the following reasons [1-2, 7, 34-36]:

1. The inherent gains has continued to ensure such studies are inevitable. These attacks will continue to rise at alarming rate with malicious activities guised to exploit users – causing great financial loss. To efficiently tackle malicious packets, most of such studies are often 'inconclusive' task being hampered in performance both by the technique used by adversaries, feature selection and adopted model – all of which, can result in model overfitting, over-training and over-parameterization.
2. We seek to validate Ojugo and Eboka [2] and Ojugo and Yoro [1] that posits feat selection, their adoption of deep neural network amongst others, as being a focal component in the success of a good fitness function as well as good classification by the model adopted.
3. Even with great results consistently reported irrespective of the heuristic method adopted, many such frameworks also yields a corresponding acceptable rate of false-positive and true-negative results in detecting malicious packets.
4. Careful diagnosis of network is often 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. Design of effective detection scheme has continued to suffer setback(s) due to the fact that malicious packets by design, are poised to evade detection. Its character size and limited available dataset continues to cripple its detection. Thus, we use feat selection in training to resolve impediment in size of parameters to be trained (though this, also can lead to poor learning of feature and in turn, poor 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.

3. METHODOLOGY

We adopt a methodology that first seeks a proposed system broken-down into 2-phases: (a) training, and (b) testing as in figure 2. For our proposed system (see figure 3) as detailed workings of the proposed system in view of figure 2. We thus adopt the methodology as below [37-40]:

3.1. Proposed Experimental Hybrid Framework

For the study, our hybrid framework is divided into three (3) components: (a) a supervised Cultural Genetic Algorithm, (b) an unsupervised modular neural network, and (c) decision support system, and (d) a knowledge-base.

- ✓ Supervised Genetic Algorithm (GA): GA model is inspired by Darwinian evolution of survival of fittest, it consists of a chosen population with potential solutions to specific task. Each potential optimal solution is found via four operators and individuals with genes close to its optimal, are said to be fit. Fitness function determines how close an individual is to optimal solution. The basic operators of GA includes initialize, fitness function and select, crossover and mutation [37]. Our GA is a variant, which has some belief spaces defined thus: (a) ***normative*** belief (has specific value ranges to which an individual is bound), (b) ***domain*** belief (has data about task domain), (c) ***temporal*** belief (has data about events' space is available), and (d) ***spatial*** belief (has topographical data). In addition, an influence function mediates between belief space and the pool – to ensure and alter individuals in the pool to conform to belief space. CGA is chosen to yield a pool that does not violate its belief space and helps reduce number of possible individuals GA generates till an optimum is found [42-43].

- ✓ Unsupervised Modular Neural Network (MNN) in [44] is an improved deep learning net with modular learning of feats of independent series and intermediary components functioning under a particular architecture. It advances a model that receive a module's output as input, to compute final output via tangent activation function. The MNN is divided into potentially, smaller and more manageable network [37] with enhanced efficiency via connected units that exponentially increase, as independent networks are added. This complicates the network structure; But, also improves computational efficiency, reduces convergence time on individual task assigned to segmented modules, and allows tasks to be executed in parallel with module that are re-organized to improve flexibility and adaptability [45]. The network enhances intelligence and increases time efficiency by reducing network's learning time – achieved via an independent training algorithms applied at each module with training dataset implemented independently and more quickly. This makes the model more flexible, adaptable and robust as rules can be re-used independently at various networks. Re-usability of rules is quite a tedious experienced with such large and complex networks. But, with appropriate data encoding and carefully selected feats – network experiences improved performance, improves flexibility via compartmentalizing and removes partitions of interfaces to eliminate redundancy [46]. The MNN is a large network of modularized networks that allows for easy learning and understandability of feats of interest. It grants a model greater flexibility via task execution parallelism via compartmentalization, is flexible, eases code reuse and adaptability [47]. Data is passed via task decomposition and training modules that support effective classification. MNN can be easily implemented using the multilayered perceptron, adaptive resonance theory and self-organizing maps. The network is trained either through the supervised, unsupervised and/or reinforcement learning approach [29].

✓ Knowledgebase – as in figure 1, as explained in the section below for the supervised genetic algorithm and the Kohonen self-organizing map neural network.

3.2. Data Sample / Feature Selection

For the study, we use Australian Defense Force Academy (ADFA 2014) dataset available and retrieved [online] from: http://cloudstor.aarnet.edu.au/plus/index.php/s/f2fgM0lG1AMBKpu. We split into: training (70%) and testing (30%) and use 7-parameters to adjust weights in table 1.

3.3. Training Phase

With feats selected as in table 1 – they are sent to GA-unit as input to perform: Encoder, Assigner, Selector, and Operators (Swapper and Changer).

3.3.1. Encoder

Unformatted data are often rippled with noise, ambiguous and partial truth. Encoding seeks to filter the dataset, mapping unto the required format a model easily understand. To encode the selected feats, we transform our dataset using selected feats as in table 1. We employ data type in Pandas Library displayed by listing 1 algorithm [29, 48-53].

```
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:  End For each
```

Listing 1: Algorithm to Convert Data type to Category

Like Ojugo et al. [26] Each chromosome becomes a rule used for training. Within each individual are seven connection feats and an attack-type encoded. We encode using the direct value. Each rule (If-Then) clause has a “condition” and “outcome”. A rule has 7-feats as in Table 1, connected with logical AND. The feat “Attack name” is for classification (at training); And, for connection (during test) if the “condition” part is matched.

Table 1. Selected Features and Representation

Features	Format	Data Types	Number of Genes
Source IP	a.b.c.d	Object	4
Source Port	Numeric	Integer	1
Destination IP	a.b.c.d	Object	4
Destination Port	Numeric	Float	1
Protocol	String	Object	1
Duration	H:M:S	Float	3
Attack Name / Type	String	Object	1

3.3.2. Assigner / Fitness Function

Model measures each rule’s “goodness” as to how many attacks it can detect. A rule is good if it can correctly classify attacks in the training data; Else, it is bad and is not selected for processing. The more attacks a rule detects, the higher its fitness value assigned. We adopt support-confidence mode as in Eq. (1-3) [26]:

$$\text{support} = |A \text{ and } B| / N \quad (1)$$

$$\text{confidence} = |A \text{ and } B| / |A| \quad (2)$$

$$\text{fitness} = w1 * \text{support} + w2 * \text{confidence} \quad (3)$$

N = Total number of network connection, |A| = Number of net connections matching the condition A, |A and B| = Number of network connections that matches the rule if A then B; and w1, w2 = weights balance between the two terms. A critical merit here is that changing the weights w1/w2, can be used to either identify intrusions with w1 = 1 and w2 = 0; Or, to classify the types of intrusions with w1 = 0 and w2 = 1 for the latter case.

3.3.3. Selector

Accepts input from assigner and apply tournament selection to distinctly separate high-from-low optimal ruleset feats. We compute probability to determine the chance of selecting a rule from the pool; We compute the cumulative probability that yields the probability of accuracy on the pool range to ensure that all values assigned are accurate; And thus, we perform the tourney selection to obtain optimal feats. We used tournament method for its flexibility to deal with premature convergence. The tournament algorithm is given as in listing 2.

```
Input: Population of chromosome
Output: Selected Chromosome to carry out crossover
1.  Randomly select 3 chromosomes from given population
2.  Pick out top 2-chromosomes based on their fitness value
3.  Return the selected two chromosome
```

Listing 2. Tournament Selection Algorithm

3.3.4. Operator (Swapper / Changer)

The operator routine consist of swapper and changer. With the tourney selection, it chooses 3 or more rules from current pool cum generation; And, afterwards – selects the best 2-rules from the selected three (3) as parents to create new offspring. This scheme was specifically selected based on its reputation of maintaining population diversity. The goal here is to create elitists rules and not just one best rule (global optimum); But, to create a set of rules that are good enough to detect intrusion (local optimums). Thus, to maintain pool diversity – we used the tournament selection. This in furtherance will also grant us greater flexibility to adopt cum adapt the two (2) swapper (or crossover).

3.3.5. Modular Neural Network

Our MNN is constructed from the feedforward multi-layer perceptron network Ghale [44]. It receives optimized rule(s) as in Figure 2 [1-2] and propagates the If-Then (selected) rules into the varying classes for detection. Rules are modeled as a production system of 4-components: (i) rule-set pattern of how rule(s) and operation(s) are applied, (b) databank of (If-Then) rules as selected data feats, (c) control strategy to specify how the rules are compared to those in the knowledgebase to find a match and ways to resolve conflicts if several rules match at the same time, and (d) rule applier. MNN yields self-learning ability and acts as the principal component analyzer with rules optimized by GA's swapper and changer (mutation) so that the trained model can effectively, autonomously classify dataset into either the normal or attack classes [54-55].

3.3.6. Training Result

Table 2 shows the top 22-rules generated at training phase having almost same fitness value of between 0.80-to-0.8065. Thus, top rules are above 80% good enough to be used for intrusion detection in testing phase. Results shows we have a set of good rules. For example, rule 14 (bold and italics) states that any connection with any number of hours, 0 minutes, any number of seconds, any protocol, with source port of 1023, with any destination port, a source IP address of 192.168.1.30, and destination IP address of 192.168.0.-1 (i.e. which means the last octet of this address could be anything from 0 to 255) is regarded as an intrusion. We can interpret other rules used thus. Also, we note that 10-of-the-22 rules have destination port as -1. This infers that most of the rules looks out for network connection from any destination port (i.e. -1). Thus, increasing the chances of detecting intrusion on any port in the network as well as improves generality of rules [56-58].

Table 2. Top 22-Generated Rules with Fitness Function Value

Time	Prot	Source Port	Dest Port	Source IP	Dest. IP	Attack	FF
-1,0,23	telnet	-1	23	192.168.1.30	192.168.0.20	PG	0.8063
-1,0,23	-1	-1	-1	192.168.1.30	192.-1.0.20	PC	0.8063
0,0,5	-1	-1	-1	192.168.1.30	192.168.0.20	PS	0.8063
0,0,5	-1	-1	-1	192.168.1.30	192.-1.0.20	PS	0.8063
-1,0,23	telnet	-1	23	192.-1.1.30	192.168.0.20	PC	0.8063
0,0,5	-1	-1	-1	192.168.1.30	192.168.0.20	ARS	0.8063
-1,0,23	telnet	-1	23	192.168.1.30	192.168.0.20	ICMP	0.8063
0,0,5	-1	-1	-1	192.168.1.30	192.168.0.20	NP	0.8063
0,0,23	telnet	-1	-1	192.168.1.30	192.168.0.20	PA	0.8063
-1,0,23	telnet	-1	23	192.168.1.30	192.168.0.20	FA	0.8063
0,0,5	-1	-1	-1	192.168.1.30	192.-1.0.20	FA	0.8063
-1,0,23	telnet	-1	23	192.168.1.30	192.168.0.20	ARS	0.8063
0,0,-1	-1	1023	1021	192.-1.1.30	-1.168.0.20	PODA	0.8031
-1,0,-1	***-1***	***1023***	***-1***	***192.168.1.30***	***192.168.0.-1***	***PODA***	***0.8031***
0,0,14	-1	-1	513	192.168.1.30	192.168.0.-1	PA	0.8031
0,0,14	-1	-1	513	192.168.1.30	192.168.0.20	SR	0.8031
0,0,14	-1	-1	513	-1.168.1.30	192.168.0.20	SH	0.8031
0,0,14	-1	-1	513	192.168.1.30	192.168.0.-1	RA	0.8031
-1,0,-1	-1	1023	-1	192.168.1.30	192.168.0.-1	DN	0.8031
0,0,5	-1	-1	23	192.168.1.30	192.168.0.20	IPS	0.8031
-1,0,-1	-1	1023	-1	192.168.1.30	192.168.-1.20	PODA	0.8031
0,0,14	-1	-1	513	192.168.1.30	192.168.0.-1	ICMP	0.8031

The rule generator used a population size of 400, and went through 5000 evolutions, with 0.05 probability of a gene to be mutated. The weight parameters (i.e. w1 and w2) used for this run was 0.2 and 0.8 respectively. Taking the first rule from table 2 as a case study, we have that:

If(duration="-1:0:23" and protocol="telnet" and sourceport=-1 and destinationport=23 and source IP="192.168.1.30" and destination IP ="192.168.0.20) then {log network connection as an Intrusion}

Furthermore, figure 4 shows the training phase scatter-graph in support of the Table 3 below – which shows the training result for the hybrid Genetic Algorithm trained Modular Neural Network framework. The key for the table 3 includes: ICMP – Internet Control Protocol Packet Internet Groper (ICMP PING), NP – Network Ping, PS – Port Scanning Utility, PAS – Packet Sniffer, PA – Protocol Analyser, PG – Password Guessing Attack, PC – Password Cracking Program, SH – Session Hijack, SR – Session Replay, IPS – IP Spoofing, DN – Domain Name Attack, RA – Rerouting Attack, FA – Flood Attack, ARS – Address Resolution Spoofing, and PODA – Ping of Death respectively.

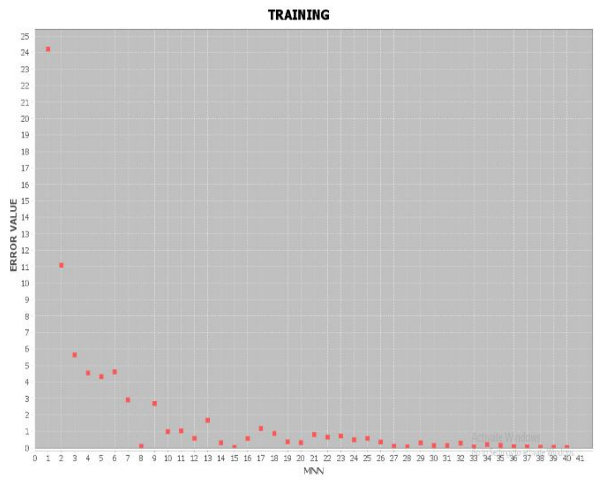

Figure 4. Training Phase Result

4. FINDINGS / DISCUSSION

4.1. Model Performance

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

$$M_r = \frac{No\ of\ Incorrect\ Classification}{No\ of\ Sample(s)\ in\ Dataset} \quad (4)$$

$$I_p = \frac{M_r(A) - M_r(B)}{M_r(A)} \quad (5)$$

Table 3. Misclassification Rate of Each model

Model	Classification Errors	
	Training Data	*Testing Data*
PHMM	13.7%	10.2%
GANN	21.3%	19.7%
Hybrid Genetic Algorithm Modular Neural Network	1.29%	1.09%

Table 4. Improvement Percentage

Model	Improvement %	
	Training Data	*Testing Data*
PHMM	56.03%	64.16%
GANN	42.79%	34.09%
Hybrid Genetic Algorithm Modular Neural Network	75.89%	92.01%

Tables 3 and 4 indicates that PHMM outperforms GANN – with a misclassification rate of 13.7% (false-positives and true-negatives). It also has a classification accuracy of 87.3%; While, promising an improvement of about 56%. In contrast, GANN has a misclassification rate of 21.3% (false-positives and true-negatives error rate) and promises an improvement of 42.79%. Both PHMM and GANN underperformed against the proposed hybrid Genetic algorithm trained Modular neural network as seen in tables 3 and 4 respectively.

4.2. Findings and Discussion

Model distinctively predicted the various attacks namely: RA – Reconnaissance Attack, EA – Eavesdropping Attacks, AA – Access Attacks, DAM – Data Manipulation Attack, SA – Session Attack, and DOSA – Denial of Service Attack. Our test dataset consists of 60 samples to determine the veracity in predicting the attacks. The confusion matrix (TP, TN, FP and FN) was computed in that of-the-60-cases, model accurately predicted Fifty-Two (52) cases (TP = 50, TN = 2); While, eight (8) cases of (FN = 5, FP = 3) were inaccurately predicted.

Figure 5 shows model sensitivity, specificity and accuracy. Sensitivity (93%) shows the capability of HGAMNN to detect occurrence of all attacks when exhibited. Also, Specificity (25%) shows the capability of HGAMNN to detect occurrence of all attacks when not exhibited or present sample case; While, Accuracy (71%) yields the degree of truth for which the model HGAMNN detects the presence and absence of any and all attacks.

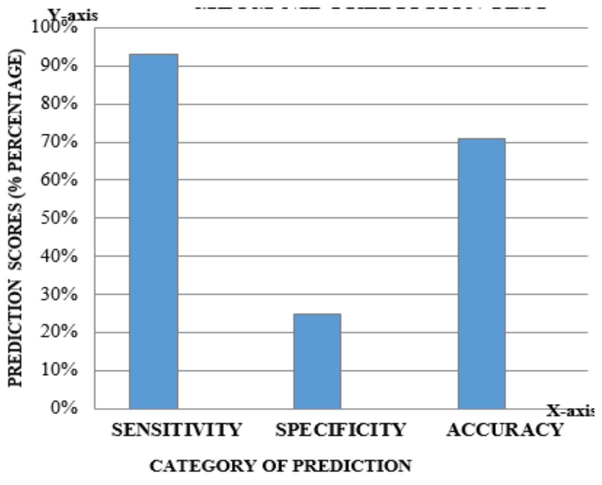

Figure 5. Categories of Prediction

4.3. Result Trade-offs

Several trade-off were noticed during result compilation and they fall under these [46-51]:

a. Result Presentation – researchers often display flawed results, modify and/or build new models rather than re-test limitations, biasness and inabilities of existing ones. Also, some researchers fail to report negative results thinking they are less valuable. We employ such data driven model to curb the non-linearity and dynamism in observed datasets used to train and test model, unlike knowledge models.

b. Efficiency – modelers use figure to show how good and well their simulations are, in agreement with observed data (even with their limited and squeezed data) with graphs that are often not easily distinguishable. Some researchers do not even provide the dataset used. Yet, their model is in 'agreement'. Some measure of goodness does not provide the relevant knowledge for the task at hand.

c. Insufficient Test – Validation compares simulated versus observed values, and many studies suffer from inadequate data. If a model seeks to simulate results, such capability cannot be demonstrated with unfounded/misleading result from limited data and misleading conclusions.

d. Model validation is a scientific dialogue – impeded by improper applications and ambiguous results.

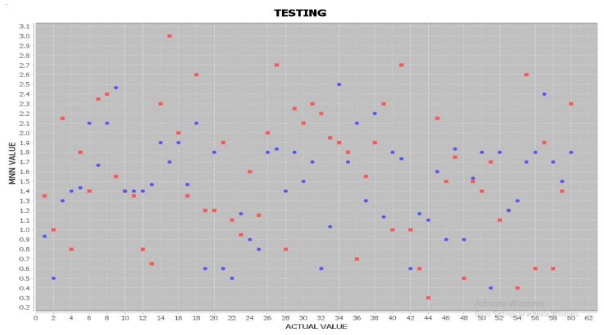

Figure 6. Test Phase Scatter-Graph

5. CONCLUSION

The unpredictability of attack patterns and the noisy nature of its many features, will continue to thrust researchers into the adoption of deep learning models to address cyber-attacks. The variance associated with machine learning dataset has also made the possibility of optimization of training sample a must if heighten predictability is to be achieved. We propose hybrid Genetic Algorithm trained Modular Neural Network to detect network anomaly cum malicious packets. The use of GA was due to its flexibility cum elitist mode; While, MNN is used as a learning paradigm for modular learning components. Model validation return a confusion matrix with these values: TP = 50, TN = 2, FN = 5, FP = 3. These values were subsequently applied to obtain sensitivity, specificity and accuracy of the model. HGAMNN shows sensitivity value of 93%, specificity value of 25% and an accuracy value of 89%.

REFERENCES

1. A.A. Ojugo., R.E. Yoro., **Forging a deep learning neural network intrusion detection framework to curb distributed denial of service attack**, Int. J. Elect. Computer Engr., Vol. 11, No. 2, pp 128-138, 2021
2. A.A. Ojugo., A.O. Eboka., **Empirical evaluation on comparative study of machine learning techniques in detection of DDoS**, J. Applied Sci. Eng. Tech. & Edu., Vol. 2, No. 1, pp18–27, 2020, doi: 10.35877/454RI.asci2192
3. C. Chia-Mei., G. Dah-Jyh., H. Yu-Zhi., O. Ya-Hui., **Anomaly Network Intrusion Detection Using Hidden Markov Model**, *Int. J. Innovative Computing, Info. and Control International*, Vol. 12, No. 2, pp.569-580
4. 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. Application*. Vol.179, No. 39. pp34–43, 2018
5. A.A. Ojugo., A.O., Eboka., E.O. Okonta., R.E. Yoro., F.O. Aghware., **Genetic algorithm rule-based intrusion detection system**, *J. of Emerging Trends in Computing Information System*, Vol. 3, No. 8, pp.1182-1194, 2012
6. P.J. Criscuolo., **Distributed Denial of Service, Tribe Flood Network**, Stacheldraht CIAC-2319. *Department of Energy Computer Incident Advisory Capability (CIAC)*, 2010
7. H. Monowar., H. Bhuyan, H. Kashyap, D. K. Bhattacharyya., J. K. Kalita., **Detecting Distributed Denial of Service Attacks: Methods, Tools and Future Directions**, *The Computer Journal*, pp. 3-19, 2012
8. A.A. Ojugo., A.O. Eboka., **Signature-based malware detection using approximate Boyer Moore string matching algorithm**, *Int. J. of Math. Sciences & Computing*, 3(5): pp49-62, doi: 10.5815/ijmsc.2019.03.05, 2019
9. M. Dadkhah, T. Sutikno., **Phishing or hijacking? Forgers hijacked DU journal by copying content of another authenticate journal**. *Indonesian J. of Elect. Engr., & Info*., Vol. 3, No. 3. Pp. 119-120, 2015
10. A.A. Ojugo., E. Ben-Iwhiwhu, O.D. Kekeje., M. Yerokun., I. Iyawah., **Malware propagation on time varying networks: comparative study**, Int. J. Modern Edu. Comp. Sci., Vol. 6, No. 8, pp. 25-33, doi: 10.5815/ijmecs.2014.08.04, 2014
11. V. Paxson., **An Analysis of Using Reflectors for Distributed Denial-of-Service Attacks**. Vol. 31, No. 3, pp. 38-47, 2001.
12. 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*., Vol. 9, No. 1, pp. 13 – 27, doi: 10.11591/ijict.v9i1.pp9-18, 2020
13. E. Ahmed., A. Clark, G. Mohay, **A novel sliding window based change detection algorithm for asymmetric traffic**, *IEEE Computer Society*. 2010, Washington, DC, USA.
14. A.A. Ojugo, A.O. Eboka., **Comparative evaluation for high intelligent performance adaptive model for spam phishing detection**, Digital Tech., Vol. 3, No.1: pp. 9-15, doi: 10.1269/dt-3-1-1, 2018
15. K.K. Vasan., B. Surendiran., **Dimensionality reduction using principal component analysis for network intrusion detection**. *Perspectives in Science*, Vol. 8, pp. 510-512, 2016
16. P.R. Mirkovic., **A Taxonomy of DDoS Attack and DDoS Defense Mechanisms**, Vol. 34, No. 2, pp. 39-53, 2004
17. S. Tobiyama, Y. Yamaguchi., et al., **Malware detection with deep neural network using process behaviour**, *IEEE 40th Annual Computer Software and Applications Conf.*, Vol. 2, pp. 577-582, 2016
18. M. Rhode., P. Burnap., K. Jones., **Early-stage malware prediction using recurrent neural networks**, *Computers & Security*, Vol. 77: 578-594, 2018
19. G. Loukas., T. Vuong et al, **Cloud-based cyber-physical intrusion detection for vehicles using deep learning**. *IEEE Access*, Vol. 6: 3491-3508, 2018
20. M. Al-Qatf., Y. Lasheng et al, **Deep learning approach combining sparse auto-encoder with SVM for network intrusion detection**. *IEEE Access*, Vol. 6: 52843-52856, 2018
21. Y. Zhang., P. Li., X. Wang., **Intrusion detection for IoT based on improved genetic algorithm and deep belief network**. *IEEE Access*, 7: 31711-31722, 2019
22. A.A. Ojugo, D.O. Otakore., **Improved early detection of gestational diabetes via intelligent classification models: a case of Niger Delta**, J. of Computer Sci. & Application, Vol. 6, No. 2, pp. 82-90, doi: 10.12691/jcsa-6-2-5, 2018
23. X.F. Wang, M.K. Reiter, **Defending against denial of service attacks with puzzle auctions**, 2003 Symposium on Security and Privacy, 2003, 78-92
24. T. Ma, F. Weng, J. Cheng, Y. Yu, X. Chen, **A hybrid spectral clustering and deep neural network ensemble algorithm for intrusion detection in sensor networks**, *Sensors*, 16: 1701, 2016, doi: 10.3390/s16101701
25. M.S. Mehdi, A.A. Zair, M.A. Bensebti, **Bayesian Networks in Intrusion Detection Systems**, *Journal of Computer Science*, 3 (5): Pp.259-265, 2007.
26. A.A. Ojugo., A.O. Eboka., E.O. Okonta., R.E. Yoro., F.O Aghware., **Genetic algorithm rule-based intrusion detection system**, *Journal of Emerging Trends in Computer and Information Systems*, 3(8): pp1182-1194, 2012.
27. K.B. Santos S.P. Chandra, M. Ratnakar, B. Dawood, Sudhakar N. **Intrusion detection system: types and prevention**, *International Journal of Computer Science and Information Technologies*, Vol. 4 (1), Pp. 77 – 82, 2013
28. R. Thomas, R., Mark, B., Johnson, T., and Croall, J. **NetBouncer: Client-legitimacy-based high performance DDoS filtering**. *Proc. of 3rd DARPA Information Survivability Conf. and Exposition*, pp. 111-113, Washington, DC, 2013

29. T. Gil, and M. Poletto, **MULTOPS: a datastructure structure for bandwidth attack detection**. *Proc. of 10th conf. on USENIX Security Symposium. Vol. 10*, pp. 13-17. Berkeley, CA, USA. USENIX Association Berkeley, 2011
30. M. Ring, S. Wunderlich, Grudl D., Landes D., A. Hotho, **Flow-based benchmark data sets for intrusion detection**. *Proceedings of the 16th European Conference on Cyber Warfare and Security (ECCWS), to appear.* ACPI, 2017
31. A. Akella, A., Bharambe, A., Reiter, M., S. Seshan, **Detecting DDoS attacks on ISP networks**. *Proceedings of the Workshop on Management and Processing of Data Streams* (pp. 1-2). San Diego, CA: ACM, 2013
32. 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, 14* (7), 15-31, 2014
33. H. Nguyen, **Proactive detection of DDoS attacks utilizing k-NN classifier in Anti-DDoS framework**. *Int. J. of Electrical, Computer, and Systems Engineering , 4*, 247–252. 2010
34. A. Angel., S. Ramamoorthy, **Intrusion Detection System by Combining Fuzzy Logic with Genetic Algorithm**, Global Journal of Pure and Applied Mathematics (GJPAM), Volume 11, No. 1, pp105 – 110, 2015.
35. P. Garcia-Teodoro, J. Diaz-Verdejo, G. Macia-Fernandez and E. Vazquez, **Anomaly-based network intrusion detection: Techniques, systems and challenges**. *Computers & Security , 28*, 18-28, 2012.
36. A.J. Deepa and D. Kavitha, **A comprehensive Survey on Approaches to Intrusion Detection System**, *Int. Conference on modelling Optimization and Computing, Procedia Engineering,* 2012, 38, Pp. 2063 – 2069.
37. R.E. Yoro., **A network intrusion detection system using hybrid genetic algorithm trained fuzzy modular neural network model,** PhD Thesis to Department of Computer, Babcock University, Ileshan-Remo, Ogun State, Nigeria, 2021.
38. R. Karimazad, A. Faraahi, **An anomaly based method for DDoS attacks detection using rbf neural networks**. *Proc. of Int. Conference on Network and Electronics Engineering* (pp. 44–48.). Singapore: IACSIT Press, 2012
39. R. Jalili, R., Imani-Mehr, F., Amini, M., Shahriari, **Detection of distributed denial of service attacks using statistical pre-processor and unsupervised networks**. *Proc of Conf. Info. Security Practice* (pp. 192–203). Singapore, 2015
40. Z. Chen, A. Delis, **An inline detection and prevention framework for distributed denial of service attack**. *The Computer Journal , 50*, 7–40, 2017
41. A.A. Ojugo., O.D. Otakore., **Forging an optimized Bayesian network model with selected parameter for detection of Coronavirus in Delta State of Nigeria**, *J. App. Sci. Eng. Tech. Edu*., 3(1): pp37–45, doi: 10.35877/454RI.asci2163, 2021
42. A.A. Ojugo, A.O. Eboka, **Signature-based malware detection using approximate Boyer Moore string matching algorithm**, *Int. J. of Mathematical Sciences and Computing*, 3(5): pp49-62, doi: 10.5815/ijmsc.2019.03.05, 2019
43. A.A. Ojugo., D.O. Otakore., **Intelligent cluster connectionist recommender system using implicit graph friendship algorithm for social networks**, *Int. J. Artificial Intelligence*, 9(3): pp497~506, doi: 10.11591/ijai.v9.i3.pp497~506, 2020.
44. B. Ghazale, **Reasoning Using Modular Neural Network – An innovative solution to address question answering AI tasks**, 2020. Available [online] and retrieved from https://towardsdatascience.com/reasoning-using-modular-neural-networks-f003cb6109a2?gi=7dbcd12eb7c, July 18, 2020
45. M.S. Gayathri Shivaraj, **Using Hidden Markov Model to detect rogue access points**. *3:394–407* . (S. C. Networks, Ed.) Nashville, Tennessee State University, U.S.A. 2010
46. Y.C. Wu, H.R. Tseng, W. Yang, R.H. Jan, **DDoS detection and traceback with decision tree and grey relational analysis**. *International Journal of Ad Hoc and Ubiquitous Computing , 7*, 121-136, 2011
47. K. Lee, J. Kim, K.H. Kwon, Y. Han, S. Kim, **DDoS attack detection method using cluster analysis**. *Expert Systems with Applications , 34*, 1659–1665, 2011
48. K. Hwang, P. Dave, S. Tanachaiwiwat, **NetShield: Protocol anomaly detection with data-mining against DDoS attacks**. *Proc of 6th Recent Advances in Intrusion Detection* (pp. 8-10). Pittsburgh, PA: Springer-verlag, 2003
49. S. Alexander, **An anomaly intrusion detection system based on intelligent user recognition**. Ph.D Thesis, University of Jyväskylä, Faculty of Information Technology, Finland. 2012.
50. E. Ahmed, A. Clark, G. Mohay, **A novel sliding window based change detection algorithm for asymmetric traffic**. *IEEE Computer Society*. Washington, DC, USA. 2010
51. 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, 2020
52. R. Bone, M. Crucianu, **Multi-step-ahead Prediction with Neural Networks**. *A review publication de l'equipe RFAI*, 2016
53. A.A. Ojugo, E. Ben-Iwhiwhu, O. Kekeje., M. Yerokun., I. Iyawah., **Malware propagation on time varying networks: comparative study**, Int. J. Modern Edu. Comp. Sci., 6(8): pp25-33, doi: 10.5815/ijmecs.2014.08.04, 2014
54. A.A. Ojugo, A.O. Eboka, **Comparative evaluation for high intelligent performance adaptive model for spam phishing detection**, *Digital Technologies*, Vol. 3, No. 1, pp. 9-15, doi: 10.1269/dt-3-1-1, 2018
55. K. Apoorv, **How to deal with IP addresses in Machine Learning algorithms**. 2016, Retrieved October 6, 2018, www.quora.com/how-can-IP-addresses-in-machine-learning-algorithms-in-traffic-analysis-and-anomaly-detection
56. W. Eddy, **TCP SYN flooding Attacks and Common Mitigations**. 2017. Retrieved June 16, 2018, from [web]: http://tools.ietf.org/html/rfc4987.
57. A.A. Ojugo., D.A. Oyemade., **Boyer Moore string-match framework for a hybrid short messaging service spam filtering technique**, International Journal Artificial Intelligence, 10(3): pp1~8, doi: 10.11591/ijai.v10.i3.pp25, 2021
58. A.A. Ojugo., A.O. Eboka., **Empirical Bayesian network to improve service delivery and performance dependability on a campus network**, International Journal of Artificial Intelligence, 10(3), pp31-43, 2021

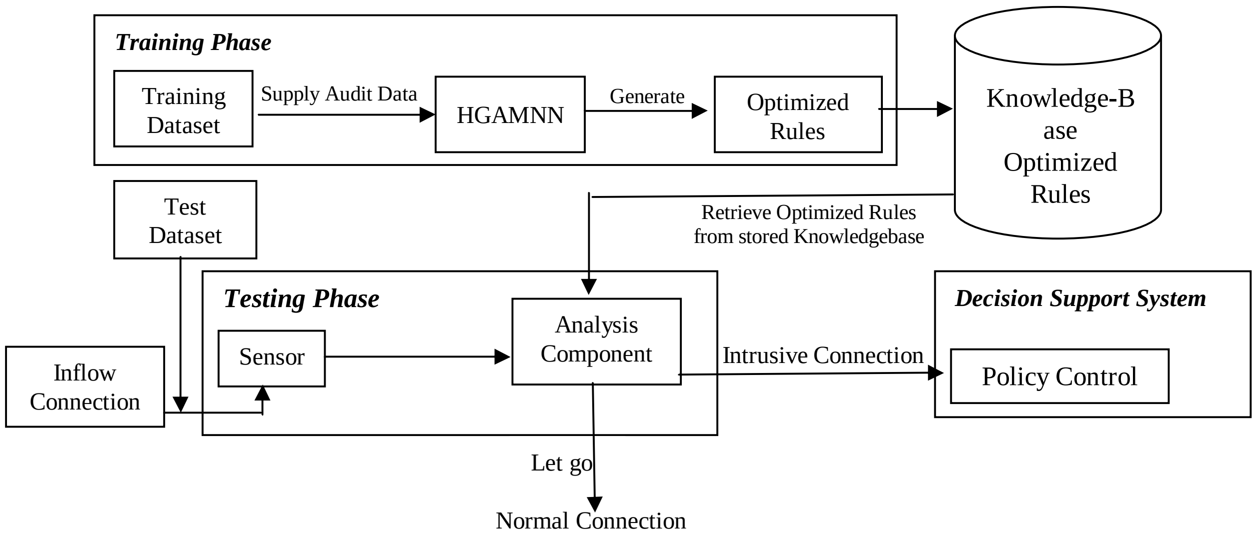

Figure 2. Framework Diagram of Proposed System for the Hybrid Genetic Algorithm Trained Modular Neural Network

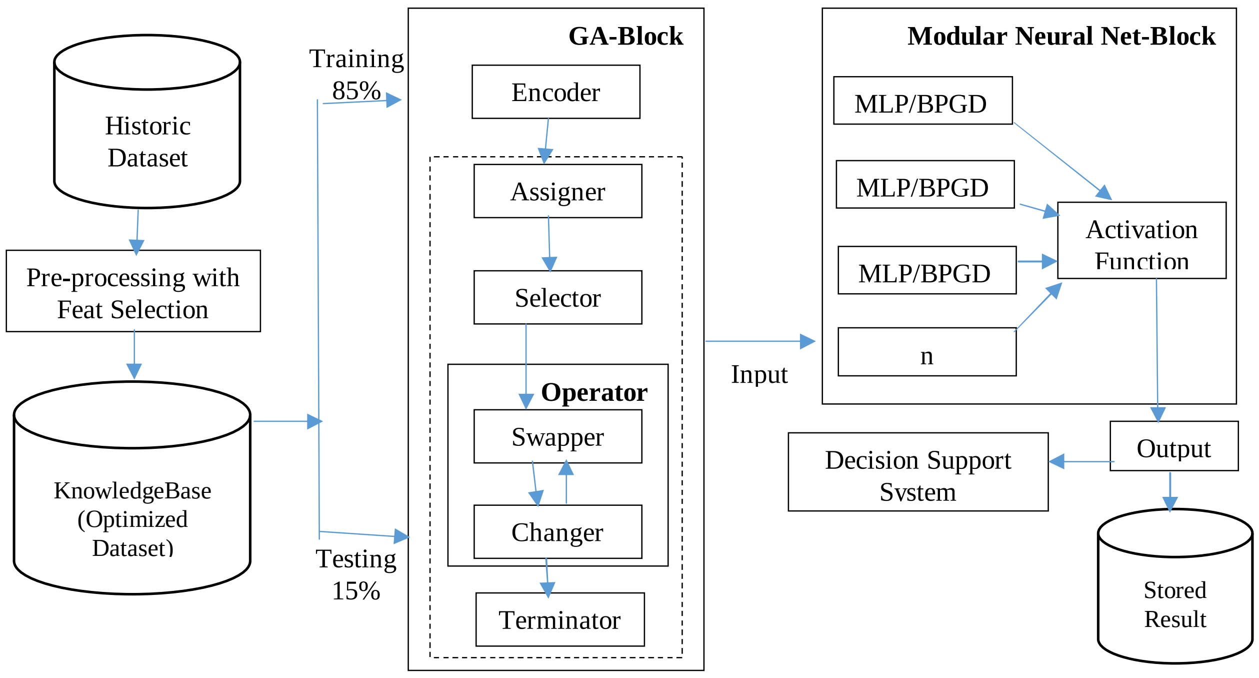

Figure 3. Schematics Diagram of Genetic Algorithm Trained Modular Neural Network

Table 4. Training Result of the HGAMNN with Attack Types Classified into Attack Groups

Attack	RA			EA		AA		SA					DMA	DOSA	
SN	ICMP	NP	PS	PAS	PA	PG	PC	SH	SR	IPS	DNS	ARS	RA	FA	PODA
1	1.8	2	1.4	1.6	0.7	0.6	2.1	2.4	0.8	0.4	1	1.4	0.1	2.4	1.5
2	0.4	1	0.4	2.9	2	1.8	1.1	0.5	0.4	0.9	2.7	0.1	1.6	1.7	1.4
3	2.8	2.8	0.6	0.3	0.7	1.7	0.1	1.2	0.1	1.5	1	2.4	1.5	0.7	1.4
4	1.8	1.1	0.5	1.1	0.1	2.5	0.6	0.8	1	0.9	2.5	0	1.4	0.2	2.4
5	1.6	0.7	0.6	0.4	1	0.8	1.8	2.4	0.8	0.4	1	1.4	0.1	2.4	1.6
6	1	0.2	1.	0.2	1.9	0	1.	0.2	0.	0.6	0.2	2	0.9	1.2	1.4

			6				2		7						
7	0.6	2.1	2.9	1.2	2.1	2.8	1	2.2	2	2.9	2.9	2	2.3	0.4	2.2
8	0.9	0.9	1	1.1	0.6	2.8	0.5	3	1.4	2.5	1.4	2.7	0.1	0.2	0.8
9	1	0.2	1.6	0.2	1.9	0	1.2	1	0.1	1	1.8	1.8	2.5	1.9	1.7
10	2.2	0.2	2.9	2.7	0.9	0.1	0.5	1.9	2.8	0.4	2.7	0.3	1.5	1.1	2.3
11	0.1	0.4	0.6	2.6	0.6	0.3	1.7	0.6	1.1	1.6	2	2.2	1.9	2.6	2.1
12	2.4	2	0.9	0	1.7	2.8	1.5	1	1	2.6	0.1	1.1	0.5	0.6	2
13	1.9	0.1	1.7	2.4	0.1	2.9	2.5	2.4	1.3	0.9	2	0.9	2.1	1.3	2
14	0.3	1.5	2.2	0.1	0.7	1.9	0.1	0.6	0	2.8	2.1	2.8	0.4	0.3	2.6
15	2.9	0.4	1.8	1	0.2	2.3	2.6	1.6	2.1	1.9	0.5	1.4	2.8	2.9	2.3
16	2.1	1.9	1.5	1.7	0.8	3	2.9	2.5	0.7	0.5	1.1	2.7	1.4	0.7	1.2
17	2.2	2.8	2.3	1	0.4	1	1.8	1.8	1.2	1.1	0.8	2.2	1.3	2.5	0.6
18	1.5	1.2	2.3	0.6	1.8	2.5	2.8	1.3	0.1	0.9	0.8	1	0.8	0	1.1
19	1.4	1.5	1.5	2.9	2.8	0.4	1.4	2.6	1.2	1.5	1.8	1.9	2.3	0.3	1.5
20	1.7	1.4	2.4	2.3	1.9	0.2	1.3	1.1	2.7	2.4	2.4	1.7	1.4	2.5	1.6
21	0.9	0.4	1	1.1	1.4	0.9	2	2.8	1.8	2.4	1.6	0.6	2.6	2.8	0.5
22	2.7	1.2	1.6	1.4	2.8	0.8	1.2	1.2	0.1	1.5	1	2.4	1.5	0.7	2
23	1	3	2.2	1.3	1.8	0.5	2.7	1	2.2	1.5	1.2	1.5	2.7	0.7	1.8
24	1.2	0.5	2.6	2.4	0.5	2.2	2.8	1	0.4	2.1	2	2.3	2.8	1.8	0.8
25	1.4	1.1	0.7	2.5	0.9	2.6	0.2	1.6	2.3	1.6	2.4	0.2	1.3	1.1	1.6
26	0.3	2.2	0.2	2.1	2.5	0.9	2.5	2.8	2.4	2.9	2.3	2.9	2.6	0.6	2.1
27	0.5	1.5	2.3	2.4	2.7	1.9	1.1	1	0.1	1	1.8	1.8	2.5	1.9	1.8
28	2.2	2.8	0.4	2.4	1.5	0	2.6	0.3	0.1	1.4	1.8	2.2	0.9	0.9	2.2
29	2	0.5	2.4	0.5	2.6	0.4	1.5	2.5	0.1	1.5	0.1	0	1.4	2.3	2.7
30	2.1	0.2	2.9	2.8	0.7	2.5	2.4	2.4	0.8	0.4	1	1.4	0.1	2.4	1.9
31	2.6	0.4	2.3	1.5	0.1	0.5	0.5	2.2	2	2.9	2.9	2	2.3	0.4	0.2
32	1.2	2.8	2.9	0.7	2.9	1.7	0.3	2.5	2.2	2.3	0.3	2.9	0.3	2.1	0.7
33	1.9	2.5	2.2	2.1	1.7	3	2.2	2.7	2.8	0.1	2.5	1.4	1.7	2.7	2.2
34	0.8	2.4	2.7	0.1	1.6	2.1	1.8	1.2	1.6	0.6	0.4	0.3	2.3	2.2	1.9
35	2.4	2.1	1.2	2.6	1.2	0.2	0.1	2.8	1.6	0.1	1.2	0.4	1.4	0.9	1.5
36	0.6	2.1	2.9	1.2	2.1	2.8	1	0.7	2.1	2.8	0.8	1.8	1.3	2.2	1
37	2.6	1.5	3	2.2	0.1	0.1	0.	2.4	1.	0.9	0.8	0.7	1.2	1.3	1.5

							8		1						
38	0.2	2.9	0.7	1.5	0.9	0.7	2.1	2	1.7	0.5	0.8	1	1.6	2.9	1
39	1.2	0.6	1.4	1	2.2	1.9	1.5	2.7	1.5	1.6	2.2	0.1	2.7	0.1	1.7
40	2.5	1.7	0.8	0.7	2.8	2.1	0.4	2.6	2.5	1.6	2.1	0.7	1.8	1.4	0.3
41	0.3	0.6	0.9	1.2	0.9	1.9	2.5	1.4	2.9	2.3	2.9	2.5	0.6	1.7	1.2
42	0.8	1.1	2.6	1.4	2.8	2.6	2.5	0	0.8	0.3	2.6	0.5	1.7	2	1.5
43	0.3	2.2	0.1	1.9	2.1	1.2	2.9	0.2	2.1	2.7	2.3	1.3	2.5	1.6	1
44	1.8	1.9	1.8	1.3	1	0.3	0.8	1.2	0.4	2.8	0.8	2.3	1.8	0.7	0.5
45	1.8	1.2	0.8	0.9	0.3	0.2	0.4	1.5	1	0.2	2.5	1	0.5	2	2.2
46	0.6	0.1	0.1	1.7	0.3	0.2	0	3	2.6	1.1	1.1	1.6	2.9	2	1.4
47	2.7	2.1	1.5	2.1	0.7	0.8	2.8	0.5	1.3	1.7	0.2	1.7	2.3	1.1	1.7
48	1.5	1	2.6	1.3	0.6	2.1	1.5	2.2	2.3	0.6	2.8	0.3	0.8	0.7	1.1
49	2.7	0.5	2.5	2.2	0.8	0	0.8	1.7	2.6	1.5	1.6	2.2	2	2.4	1.2
50	1.6	0.3	0	0.6	2.9	2	1.4	0.8	0.7	2.3	2.2	1.8	2.4	2.2	0.4
51	0.3	1	1.6	2.2	1.6	2.9	2.9	0.9	2.7	1.1	1.7	0.1	2.6	2.4	2.6
52	1.9	0.6	0.6	2.7	1	0.8	1.8	2.1	2.9	0	0.6	1.1	3	0.9	0.6
53	1.6	0.7	0.6	0.4	1	0.8	1.8	0.9	1.5	1.6	1.6	2.6	2.7	1.6	1.9
54	1	0.8	3	0	2.5	1.5	2.5	2.5	0.1	1.2	1.4	2.5	0.9	1.1	0.6
55	0.2	0.3	0.8	0.8	0.9	0.4	2.4	1.5	1.4	2.4	0	2.1	3	3	1.4
56	2.5	0.8	2.1	1.9	1.4	1.6	1.2	2.2	1.6	1.8	1.4	1.1	2.3	1.8	2.3
57	1.8	2	1.4	1.6	0.7	0.6	2.1	2.6	1.4	0.1	1.5	2.1	2.1	0.8	2.8
58	2.1	2.2	1.8	0.4	2	0.4	0.8	1.8	0.3	0.8	2.7	1.6	2.9	2.3	0.7
59	0.4	1	0.9	2.2	2.5	1.3	3	1.5	1.6	0.7	1.6	2.9	3	1.4	0.9
60	2.5	1.5	0.7	0.8	0.5	0.5	2.7	1	1.6	2.7	1.1	0.9	2.5	1.1	1.5

Table 5. Result of Efficiency of HGAMNN at Testing Phase

S/N	RA	EA	AA	SA	DAM	DOS	AA	AA	TP
1.	1.145221	0.740639	1.238624	0.825188	0.912576	0.24069543	AA	AA	TP
2.	1.298007	0.986543	1.791807	1.186744	1.161781	0.92057455	AA	AA	TP
3.	1.462402	1.188071	2.674925	1.279731	1.671345	1.19477387	AA	AA	TP
4.	1.779682	1.308777	2.666281	1.353374	1.213471	0.54475628	AA	AA	TP
5.	1.096717	1.140673	2.625089	0.993437	0.773264	0.5475417	AA	AA	TP
6.	1.362148	1.378951	2.475307	1.397318	1.571091	1.49257306	AA	AA	TP
7.	1.654572	1.723601	2.341845	1.11437	0.906337	1.68077918	AA	AA	TP

8.	2.088441	1.899401	2.450093	1.853578	1.775133	1.46754675	AA	AA	TP
9.	1.168651	1.858298	1.618963	1.643482	1.590895	0.98409124	AA	EA	FN
10.	0.608126	1.265237	1.786271	2.02092	1.616803	1.58973958	AA	SA	FN
11.	1.036154	2.178259	2.323616	1.144753	0.669079	1.19001043	AA	AA	TP
12.	1.385181	2.065562	1.87516	1.752237	1.79577	0.73513175	AA	EA	FN
13.	1.125601	1.093824	1.508817	1.395502	1.348271	1.47307977	AA	AA	TP
14.	1.163178	1.806082	2.872712	1.363866	1.820904	1.9412663	AA	AA	TP
15.	0.537211	1.41855	1.694458	1.624039	1.080302	0.68066651	AA	AA	TP
16.	1.60784	1.706436	1.387475	1.477592	1.722393	0.78385333	DMA	DMA	TP
17.	1.436983	2.406047	1.537661	2.577004	2.546922	0.95404663	DMA	SA	FN
18.	1.450412	0.795849	0.473358	2.005779	2.291248	0.76097431	DMA	DMA	TP
19.	0.868102	1.6953	1.520961	1.67332	2.01163	1.25818485	DMA	DMA	TP
20.	1.483137	1.934566	1.08047	1.973112	1.974995	1.34559804	DMA	DMA	TP
21.	0.758749	1.025694	0.092034	1.591964	2.305973	0.9708285	DMA	DMA	TP
22.	1.63458	1.915358	1.18654	1.259972	1.913305	1.42120613	DMA	EA	FN
23.	1.528723	1.928169	1.161088	1.500427	1.79483	1.41576289	DMA	EA	FN
24.	1.383908	0.790485	1.291135	1.219047	1.745576	1.25585408	DMA	DMA	TP
25.	0.787106	0.83199	1.065448	1.100579	0.83516	1.44015847	DOSA	DOSA	TP
26.	0.400954	1.295232	0.47968	0.872047	0.534323	1.20401244	DOSA	EA	FN
27.	0.462054	1.008137	0.58825	1.275179	1.141453	1.67491842	DOSA	DOSA	TP
28.	1.019359	1.544497	0.975073	1.356528	1.286769	1.61675307	DOSA	DOSA	TP
29.	1.249457	0.717916	1.998781	1.208508	1.891255	2.08888464	DOSA	DOSA	TP
30.	1.634014	2.322682	0.648019	1.956776	1.765493	1.95249323	DOSA	EA	FN
31.	0.745678	1.319641	1.301297	1.723985	2.275666	2.52417574	DOSA	DOSA	TP
32.	0.908548	1.711734	0.82181	1.091148	1.278484	2.17205165	DOSA	DOSA	TP
33.	1.53547	1.754345	0.96574	2.186631	1.797608	2.4745849	DOSA	DOSA	TP
34.	0.990058	0.95513	1.664435	0.910087	1.315444	2.01445495	DOSA	DOSA	TP
35.	1.531673	1.137577	1.390784	1.162937	1.448822	2.69561644	DOSA	DOSA	TP
36.	1.242974	0.607483	0.279377	1.071257	0.773388	2.10512873	DOSA	DOSA	TP
37.	1.575336	1.780247	0.328159	1.635776	1.761581	2.18301009	DOSA	DOSA	TP
38.	0.786443	1.809597	1.553314	1.751552	1.483655	2.06529269	DOSA	DOSA	TP
39.	0.89827	0.664382	1.289824	1.549923	1.672583	1.94438136	DOSA	DOSA	TP
40.	0.398665	0.917867	1.233003	1.340682	1.947955	2.65383471	DOSA	DOSA	TP
41.	1.228751	3.036853	1.155614	1.02652	0.755642	0.76015573	EA	EA	TP
42.	1.556566	1.960189	0.262416	1.450106	1.543339	1.18606119	EA	EA	TP
43.	1.660613	3.517836	0.719227	1.837238	1.691638	1.083169	EA	EA	TP
44.	1.683847	2.13692	0.926975	1.480859	1.036756	0.60533973	EA	EA	TP
45.	1.258429	3.154816	1.196924	1.307103	1.023035	1.36194814	EA	EA	TP
46.	1.099156	2.557461	1.881641	1.740465	1.590554	0.70322263	EA	EA	TP
47.	1.430446	2.101897	1.11758	1.400688	1.512653	1.08198506	EA	EA	FN
48.	1.752706	1.131862	1.291279	1.114632	0.906651	1.68192345	RA	RA	TP

49.	1.887932	0.503047	0.843047	1.139533	1.042396	0.6117324	RA	RA	TP
50.	1.826242	0.766047	1.219812	1.206673	1.260082	1.61894798	RA	RA	TP
51.	2.114548	2.417205	1.036036	1.13538	0.600058	0.40006945	RA	EA	FN
52.	2.100956	2.191988	0.125554	1.157001	1.347212	0.42027154	RA	EA	FN
53.	2.015753	1.157165	0.42147	0.990854	1.35745	1.69381217	RA	RA	TP
54.	1.592386	1.165703	0.511642	1.400343	1.497606	1.21498618	RA	RA	TP
55.	2.283751	1.338653	1.772224	1.024978	1.442303	1.43468776	RA	RA	TP
56.	1.549626	1.412527	1.082289	1.30949	1.382474	1.08664489	RA	RA	TP
57.	1.394832	1.376607	0.50429	1.460414	1.327358	1.22384431	RA	SA	FN
58.	1.249294	1.657261	1.782562	2.3299	2.280578	0.75408993	SA	SA	TP
59.	0.91905	0.868423	1.523947	1.906193	1.684543	0.46394474	SA	SA	TP
60.	0.568968	0.451055	0.851288	1.435549	0.875701	0.29922454	SA	SA	TP