A Hybrid Nested Genetic-Fuzzy Algorithm Framework for Intrusion Detection and Attacks

RAMY ELHEFNAWY1, (Member, IEEE), HASSAN ABOUNASER1, AND AMR BADR2

1Department of Computer Engineering, Arab Academy for Science, Technology & Maritime Transport–Cairo, Cairo 2033, Egypt
2Department of Computer Science, Faculty of Computers and Artificial Intelligence, Cairo University, Cairo 12613, Egypt
Corresponding author: Ramy Elhefnawy (relhefnawy@outlook.com)

ABSTRACT Intrusion Detection System (IDS) plays a very important role in security systems. Among its different types, Network Intrusion Detection System (NIDS) has an effective role in monitoring computer networks systems for malicious and illegal activities. In the literature, the detection of DoS and Probe attacks were with reasonable accuracy in most of the NIDS researches. However, the detection accuracy of other categories of attacks is still low, such as the R2L and U2R in KDDCUP99 dataset along with the Backdoors and Worms in UNSW-NB15 dataset. Computational Intelligence (CI) techniques have the characteristics to address such imprecision problem. In this research, a Hybrid Nested Genetic-Fuzzy Algorithm (HNGFA) framework has been developed to produce highly optimized outputs for security experts in classifying both major and minor categories of attacks. The adaptive model is evolved using two-nested Genetic-Fuzzy Algorithms (GFA). Each GFA consists of two-nested Genetic Algorithms (GA). The outer is to evolve fuzzy sets and the inner is to evolve fuzzy rules. The outer GFA assists the inner GFA in training phase, where the best individual in outer GFA interacts with the weak individual in inner GFA to generate new solutions that enhance the prediction of mutated attacks. Both GFA interact together to evolve the best rules for normal, major and minor categories of attacks through the optimization process. Several experiments have been conducted with different settings over different datasets. The obtained results show that the developed model has good accuracy and is more efficient compared with several state-of-the-art techniques.

INDEX TERMS Intrusion detection system, evolutionary computation, fuzzy logic systems, genetic algorithms, hybrid intelligent systems, classification algorithms.

I. INTRODUCTION
With the emergence of new technologies in Internet services, such as cloud computing and Internet of Things (IoT), the vast use of communication networks technology has been increased. In this regard, computer networks security has been one of the major concerns in computer societies [1]. Intrusion Detection System (IDS) plays a core function in computer networks security, where it provides proper protection against malicious activities [2]–[4]. Moreover, the IDS goal is not only to detect successful penetrations of the malicious activities from intruders, but also, to monitor any attempts to break security via providing timely information about current security system [5], [6]. In the literature, most of the IDS researches focus on developing accurate and effective techniques to monitor intruders via enhancing methodologies of protection [7].

Based on the detection approach, IDS systems can be categorized into three types. The first type is signature-based, which is designed to detect attacks by comparing incoming traffic with predefined signatures. The second type is anomaly-based, which is designed to focus on the behavior of activities over the normal environment. The data generated from anomaly-based systems can be used to update signature-based systems. The third type can be hybrid from signature-based and anomaly-based [8].

The advantage of signature-based system is that it lowers the number of False Alarm Rate (FAR). However, if the zero-day attacks encountered such that the signature does not exist or specifically modified, the attack cannot be detected which is a major drawback. On the other hand, anomaly-based system has capabilities to handle both features.
However, if not well optimized, it may highly generate FAR. Most of the IDS systems use hybrid systems to get the advantage of signature-based and anomaly-based systems [9].

The continuous evolution of security threats led to the continuous development for Network Intrusion Detection Systems (NIDS), Classical Machine Learning (ML) and Data Mining (DM) techniques met several obstacles in tackling this challenging problem. For instance, the information can be noisy such that it leads to over fitting. In addition, the features used during the training phase can be redundant and irrelevant [10]. Moreover, reports generated from most of the NIDS are very huge such that it needs flexible classifier to mine useful patterns from these reports [11].

Rule-based classifiers use a set of linguistic IF-THEN rules for classification. The rules generated from these classifiers are considered as a knowledge-based system [10]. The classical techniques in building such rule-based systems cannot tolerate imprecision and uncertainty. Therefore, for the big data that is generated from NIDS or Host-based Intrusion Detection System (HIDS), other techniques are needed to build flexible rule-base classifier that can generate robust rules for detecting attack occurrence in the network [2]. Computational Intelligence (CI) techniques are non-classical techniques that function like a human being in learning tasks from data or observations. In other words, CI systems have characteristics that makes it flexible to be utilized in building efficient models in different domains. Some of these characteristics include high computational speed, fault tolerance, adaptation, and ability to error resilience in modeling noisy information [12], [13].

Fuzzy Logic (FL) is one of the CI techniques inspired from how the brain thinks in measuring uncertainty. Fuzzy Logic Systems (FLS) or Fuzzy Rule-Based Systems (FRBS) have robust features that tolerate imprecision and uncertainty, and therefore, perform rule-based classification efficiently and effectively [6]. However, FLS is not adaptive by itself and it is candidate for optimization [14]. In this regard, one of the most popular Evolutionary Computation (EC) algorithms that has strong global optimization capability is Genetic Algorithms (GA). Hence, in this research, a novel Hybrid Nested Genetic-Fuzzy Algorithm (HNGFA) is proposed as a contribution in building flexible rule-based classifier for NIDS. The proposed technique has been tested and proven its capability of evolving an optimized model, with high accuracy and low FAR, that enhances the classification accuracy for specific categories of attacks. In the meanwhile, feature selection methodology and its effectiveness on the classifier output are also considered.

The rest of this paper is organized as follows. Section II introduces the CI techniques utilized in this research. Section III presents the problem statement. Section IV reviews the related work in the NIDS domain. Section V illustrates the proposed framework. Section VI presents the chosen datasets and discusses the obtained results from the conducted experiments. Finally, section VII highlights the conclusion and future work.

II. COMPUTATIONAL INTELLIGENCE (CI) TECHNIQUES

A. GENETIC ALGORITHMS (GA)

EC is a wide range of algorithms inspired by biological evolution and mainly utilized for global optimization. Among of its subsets, Evolutionary Algorithms (EA) are population-based metaheuristic optimization algorithms which utilize mechanisms such as crossover, selection and mutation. EA are not only used to find solutions for optimization problems but also it can be applied successfully for a various range of other domains such as, in control [12], [15], regression [16], clustering [17] and classification [18].

GA is one of the most popular subsets of EA that are widely utilized to generate high quality solutions for optimization problems [19]. GA are numerical adaptive search techniques developed by John Holland (1975) and inspired by Darwin’s theory of natural evolution [13]. In other words, GA mimics the processes of evolution for natural populations, where the process of reproduction of good offspring is generated from the selection of the fittest individual. The best individuals generated from parents are the candidates to survive.

Mainly, GA has four phases to be applied. The first phase is designing the individual (or chromosome) structure, that encodes the candidate solution. The chromosome consists of a string of genes. The gene represents the basic unit that handles the characteristics of the chromosome, where its possible value is called allele [19]. The gene representation can be in different forms such as binary, integer or real values. For example, in binary-coded chromosome, each gene value is either 0 or 1 whereas for real-coded chromosome any value can be assigned to the gene from the current domain. Each chromosome represents one point in the search space and a group of chromosomes is a population [13]. The second phase is selecting the best individuals, based on defined fitness function. The third phase is the reproduction of the next generation. The fourth phase is mutating the selected individuals and the replacement of weak candidates with highest fitness one. However, formulating a good fitness function is one of the most common challenges of GA [20]. In addition, chromosome structure design is one of the biggest challenges in order to design an effective GA [21].

B. FUZZY LOGIC SYSTEMS (FLS)

Knowledge Discovery in Databases (KDD) is a nontrivial process of identifying correct, potentially useful, and understandable patterns in data. FL is one of the most commonly utilized techniques in KDD and is one of strongest CI techniques [22]. The main idea of FL technique originated from the relation between mathematics, certainty, and reality. The basic concept originated from many mathematical sciences works and theories. Lejewski and Lukasiewicz have much fundamental work on multivalued logic (ternary logic) and developed the first alternative to two-value logic, which developed by Aristotelian Logic theory [13]. As an extension of multivalued logic, Lotfi Zadeh (1965) defined FLS as a logical system for approximate reasoning [23].
Generally, statistical uncertainty is based on the laws of probability. In contrast, FL is referred to as a non-statistical uncertainty since it mimics human reasoning, which tolerates uncertainty. In other words, FL defines semantic basics of uncertainty, vagueness, imprecision, and incompleteness. Hence, FLS can be defined as a linguistic computing technique that converts linguistic experience into mathematical information to handle complex issues related to incomplete or noisy data, patterns and others [23]. The strength of FL makes it interactive and efficient in different domains, such as in control, pattern recognition, robotics, mathematics, fuzzy database and fuzzy expert system [24].

When facts are collected measurements or observation, it is possible to have good decisions on data. But when complexity and lack of information exist, uncertainty is produced. However, using correlation and meaningful interpretation of data can produce good decisions [25]. In this regard, the definition of fuzzy sets and FL are the keys to what is referred to as approximate reasoning. The uncertainty degree is determined by fuzzy sets whereas the FL infers new facts from these uncertain facts [13].

The study of FL can be viewed from two perspectives, namely, narrow view and board view. From narrow view perspective, FL is an extension to the multivalued logic system that focuses on approximate reasoning in symbolic logic. From board view perspective, FL is almost equivalent to fuzzy sets theory in which membership is a matter of degree. However, other views also exist, such as fuzzy mathematical programming, fuzzy arithmetic, fuzzy decision analysis and fuzzy topology [26]. Fig. 1 shows the main components of FLS, namely, fuzzification, fuzzy rule base, decision making logic (or inference), and defuzzification [27].

Fuzzification is the process of transforming input data crisp or fuzzy set into the valuation of subjective values. In other words, it maps input data of an observed input space to labels (degrees of membership) of fuzzy sets. Fuzzy rule-base (or knowledge-base) is a container that collects the fuzzy sets along with all the rule-base in the form of IF-THEN rules, mostly offered by the domain expert, to control the FLS. The fuzzy sets (or membership functions) link the degree of truthfulness for the linguistic terms. The main role of inference component is to help FLS to determine the degree of matching between fuzzy inputs and the rules. Based on the percentage of correspondences, it determines which rules are to be implemented for the given input field. After that, applied rules are composited to evolve the control actions. Finally, the process of converting fuzzified data or fuzzy sets into crisp will be performed by the defuzzification [14]. The above model is commonly used in most FLS designs whereas others make few modifications.

C. HYBRID GENETIC-FUZZY ALGORITHMS

Hybridization in algorithms means to combine two or more other algorithms to solve the same problem more efficiently and effectively than the standard algorithms. There are several examples of hybrid algorithms that are used in intrusion detection domain (e.g., [10]). In the literature, hybridization is usually utilized for optimization purposes, such as to improve the accuracy or the performance. One of the many examples is using GA to evolve fuzzy decision trees which leads to improving convergence and reducing excessive tree growth [28]. Another example is using EA to optimize the Artificial Neural Networks (ANN), where EA is utilized to optimize the modeling parameters of ANN such as the weights, learning rules and network architecture for better training [29]. However, the quality of the solution obtained from EA is another factor in the hybridization process [30].

III. PROBLEM STATEMENT

Most of the NIDS researches noticed that common categories of attacks, such as Denial of Service (DoS) and Probe, are detected with reasonable accuracy. However, other categories of attacks, such as Remote to User (R2L) and User to Root (U2R), are detected with very low accuracy [5], [31]. In addition, most of the effort in NIDS researches has been devoted to detecting and classifying major categories of attacks without considering the benefits that can be gained from focusing on minor categories of attacks [32]–[34]. Hence, the purpose of this research is to propose a novel technique for detecting and distinguishing the good connections from the bad connections of such attacks, either major or minor categories, considering the effectiveness of feature selection methodology on the output of the developed predictive model.

IV. RELATED WORK

Since last decades, researches and developments in the NIDS domain are still ongoing because of its transformation. In the literature, a lot of researches utilizing classical ML algorithms have been proposed, such as Decision Tree (DT), Naïve Bayes (NB), Logistic Regression (LR), Expectation-Maximization (EM) clustering, ANN [35], FL [36], GA, Support Vector Machine (SVM) [6], Hidden Markov Model (HMM), and K-Nearest Neighbor (KNN) [4].

Tsang et al. [10] proposed a Multi-Objective Genetic-Fuzzy IDS (MOGFIDS) technique for anomaly detection. This technique can also act as a wrapper feature selection, by finding the optimal set of features. In addition, a Genetic-Fuzzy Rule-Based System (GFRBS) is evolved from an intelligent multiagent-based evolutionary framework. The framework is proposed to construct the GFRBS regarding the
interpretabillity and accuracy for the IDS. The authors utilized the KDDCUP99 dataset for training and testing. Moreover, the model is extracted as fuzzy IF-THEN rules, with Detection Rate (DR) of 92.77% and a precision of 74.74% in classifying normal network traffic. The technique classifies four major categories of attacks, namely, DoS, Probe, U2R, and R2L. However, the low DR and precision for both U2R and R2L make this technique not accurate in the IDS domain.

An intelligent IDS has been proposed by Ganapathy et al. [37] to detect attacks in wireless networks. The authors developed a Weighted Distance Based Outlier Detection (WDBOD) algorithm to enhance Conformal Prediction for K-Nearest Neighbor (CP-KNN) nonconformity calculation. In this model the detection accuracy for DoS and Probe attacks is over 99% on the KDDCUP99 dataset.

For better IDS detection accuracy, other techniques based on Fuzzy rough set algorithms are widely studied in [38], [39]. Intelligent agent-based IDS, using Fuzzy Rough Set based outlier detection and Fuzzy Rough Set based SVM, is proposed by Jaisankar et al. [38]. The authors used the KDDCUP99 dataset for the conducted experiments. The experimental results show that the proposed model achieves high DR, compared with other techniques. Jaisankar et al. [39] proposed an intelligent IDS version that improves the detection accuracy based on Fuzzy Rough Set based C4.5 algorithm. The authors used the KDDCUP99 dataset in conducting experiments simulation. The proposed system has been compared with the SVM. The obtained results show that the detection accuracy is enhanced and the FAR is reduced.

One of the supervised learning techniques is using Fuzzy Rules (FR). In some of the proposed IDS techniques, GA has been used to adapt the generated FR to detect some of unknown attacks. In this regard, Jongsuebsuk et al. [40] introduced a real-time IDS to detect known and unknown types of attacks, using fuzzy-genetic algorithm. The authors utilized the RLD09 dataset for training and testing. The utilized dataset has two major categories of attacks, namely, DoS and Probe. In addition, the dataset has 17 minor categories of attacks, categorized into two major attacks along with normal traffic. The average result of testing accuracy was approximately 97% with a False Positive (FP) of 1.13 and a False Negative (FN) of 4.10.

An Intelligent IDS model for classification and attribute selection was developed by Ganapathy et al. [6]. The classification algorithm is called Intelligent Rule-based Enhanced Multiclass SVM (IREMSVM). The algorithm is a modified version of Intelligent Agent-based Enhanced Multiclass SVM (IAEMSV) algorithm in the methodology of classes sampling. The authors introduced a new technique for attributes’ selection using rules and information gain ratio over the KDDCUP99 dataset. A rule-based approach has been applied for tuples selection. The classification accuracy for DoS and Probe categories using 19 features was very high, compared with other categories of attacks.

A new IDS model for classifying low-frequent attacks has been structured by Kuang et al. [34]. The model is based on combining Kernel Principal Components Analysis (KPCA) and SVM to achieve higher detection precision and stability. In this model, the GA has been used to optimize the SVM parameters while Gaussian Radial Basis kernel Function (N-RBF) is developed to shorten training time and the performance. The authors developed a multi-layer SVM classifier to evaluate whether a traffic is normal or attack. The KPCA is utilized to reduce the dimensions of features to the classifier as a preprocessor. The DR using the KDDCUP99 for DoS and Probe attacks was reasonable whereas the experimental results for the U2R and R2L attacks are all unsatisfactory.

Ambusaidi et al. [33] proposed an IDS model namely, LSSVM-IDS. In this model, the authors combined a feature selection algorithm called Flexible Mutual Information Feature Selection (FMIFS) along with the proposed Least Square SVM based IDS. The FMIFS algorithm is an evolution of Battiti’s algorithm, with the main objective of reducing features’ redundancy. In this model, three datasets are utilized for model evaluation, namely, KDDCUP99, NSL-KDDCUP99 and Kyoto 2006+. The results obtained, utilizing the KDDCUP99 dataset of corrected labels, demonstrated a low DR for both U2R and R2L attacks with an overall accuracy of 78.86%.

Later, a new model for an IDS based on Fast Learning Network and Particle Swarm Optimization (PSO-FLN) has been developed by Ali et al. [5]. The authors utilized the KDDCUP99 dataset for the conducted experiments. In this model, the authors found that the number of hidden neurons controls the accuracy and affects the total system performance. The results showed that the model outperforms other learning approaches in testing accuracy. In addition, the authors found that the R2L attacks have lower accuracy compared with other categories of attacks.

In more recent studies, a proposed multiclassification model for network anomaly detection using ML was introduced by Nawir et al. [32]. The model is termed online Average One Dependence Estimator (AODE) algorithm, which is an enhanced version of NB algorithm. The AODE averages the attributes of all predictions of multiple 1-dependence classifiers, based on the single parent attribute. In this model, the authors utilized the UNSW-NB15 dataset and reported an accuracy of 83.47% with a FAR of 6.57%. In addition, the model has high accuracy rate in detecting Worms attack compared with other categories of attacks.

P. Nancy et al. [41] proposed a model for feature selection and classification. In feature selection, the authors developed a new model termed Dynamic Recursive Feature Selection Algorithm (DRFSA). This model takes the advantages of both wrapper and filter methods. For classification, an intelligent decision tree has been developed by extending the traditional decision tree algorithm with temporal and fuzzy rules. In this work, KDDCUP99 dataset has been used to evaluate the proposed algorithm. The detection accuracy for both DoS and Probe is acceptable compared with U2R and R2L, which is very low.
Comparing with previous works, the developed technique is different in many ways. First, it classifies both majors and minor categories of attacks. Second, it uses a minimum number of features to speed up the decision-making. Third, the interaction between the two-nested GFA increases classification accuracy. Finally, the generated linguistic “IF-THEN” rules yield to better readability in outputs.

V. PROPOSED SYSTEM

The aim of this research is to enhance the prediction process for real-time NIDS by building a flexible predictive model that tolerates uncertainties, in good and bad connections. To address this issue, a novel HNGFA framework has been designed and developed to improve the accuracy in distinguishing between normal traffic and most of the intrusion’s categories, whether major or minor categories of attacks, particularly the categories that have rare information in datasets. Fig. 2 shows the general structure for the proposed framework. As shown, the proposed framework consists of two main components, namely, the data preprocessing and the Hybrid Nested Genetic-Fuzzy Engine (HNGFE).

A. DATA PREPROCESSING

Most of the NIDS datasets are collected from network sniffers and, therefore, have a lot of features. For example, the KDD-CUP99 dataset has a total of 41 features and the UNSW-NB15 dataset has a total of 49 features [42]. Therefore, if all dataset features are utilized in the training and testing phases without preprocessing, it will affect the classifier’s performance due to high resources consumptions. Fewer features are eligible since it reduces the intricacy of the pattern and makes the training and testing phases simpler and faster for the classifier. However, selecting random features from datasets may decrease the efficiency and increase the overall complexity of classifiers. Hence, data preprocessing is necessary to minimize the number of features, via eliminating irrelevant and redundant features in order to improve classifier performance and maintain high DR for real-time NIDS.

In this research, the number of features has been reduced and unified for all datasets utilized. In this regard, different feature selection methods have been utilized and evaluated. The feature selection is performed based on maximizing cross-validation accuracy. As it can be seen in Fig. 2, two labeled subsets are generated as an output from this stage. The outputs are categorized into two symmetric dataset files. The first dataset is dedicated for major categories of attacks with its relevant and unduplicated features. Similarly, the second dataset is dedicated for minor categories of attacks with its relevant and unduplicated features. The following steps summarize the data preprocessing stage:

1. Check data redundancy to remove duplicates.
2. Select top features from normalized dataset, for both major and minor categories of attacks.
3. Perform intersection between features in the two labeled subsets.
4. Redistribute features such that the features for major categories of attacks consist of common features between the two labeled subsets along with its top ranked features, while the features for minor categories of attacks consist only from its top ranked features rather than the common features.
5. Normalize output for each subset.

B. HYBRID NESTED GENETIC-FUZZY ENGINE (HNGFE)

A classifier is an algorithm utilized to build classification model from an input dataset to classify objects or data. The effectiveness or attitude of FLS classifier is controlled by many parameters such as membership functions, fuzzy sets, structure or technique used to prioritize values and fuzzy rules. Since FLS has no learning ability by itself, EA can be utilized to optimize these parameters. However, optimizing all FLS parameters puts a huge burden. In this research, fuzzy rules, fuzzy sets and membership functions are only optimized.

The set of fuzzy rules in the knowledge-base are represented as linguistic IF-THEN rules. The size of the rule is dependent on the size of the features. The size of the rule-base is controlled by the size of the dataset utilized. However, to avoid ignorance or explosive in the classification process, a limitation is imposed for the number of rules in rule-base, as well as the duplicated rules is ignored. For compactness and simplicity, Virtual Fuzzy Associative Matrix (VFAM) has been utilized for storing the rule-base.

The fuzzy set and its membership functions are feature dependent. For continuous domain, the membership function can be triangle or trapezoidal shapes, for example. In the literature, an overlapping degree between 25% and 50% in membership functions is efficient for a real-time FLS [43]. To reduce computational cost, only three fuzzy sets are recognized for input variables. On the other hand, for discrete domains, the membership function is singleton.

The mapping of fuzzified inputs to rule-base is performed in the inference process to produce fuzzified output for each relevant rule. The firing strength of each rule is determined using the min operator, as in (1).

$$\alpha_{R_i} = \min \{\mu_{D_1}(d_1), \mu_{D_2}(d_2), \ldots, \mu_{D_n}(d_n)\}$$

where $\alpha_{R_i}$ is the firing strength of $R_i$th fuzzy rule, $n$ is the number of features in dataset, $d_1, \ldots, d_n$ are input linguistic
variables, $\mu_{Di}$ membership function of fuzzy set $D_i$, and $\mu_{Di}(d_i)$ is the membership degree of fuzzified input $d_i$ in $\mu_{Di}$. After that, one single fuzzy value is assigned for each output. The final fuzzy value associated with each output is calculated using the max operator, as in (2).

$$\beta_i = \max_{M} \{ \alpha_{Ri} \}$$  \hspace{1cm} (2)

where $\beta_i$ is the max value for each fuzzy rule, $\alpha_{Ri}$ is the firing strength of $R^i$ fuzzy rule and $M$ is the total number of fuzzy rules in rule-base. Finally, the defuzzification process computes the centroid of the composite area using clipped center of gravity method, in order to convert the fuzzy output of fuzzy rules into crisp value, as in (3).

$$\text{output} = \frac{\sum_{i=1}^{n} (\alpha_{Ri} \times \mu_{Di}(d_i))}{\sum_{i=1}^{n} \alpha_{Ri}}$$  \hspace{1cm} (3)

where $\alpha_{Ri} \times \mu_{Di}(d_i)$ is the max defuzzification, $n$ total number of fired fuzzy rules and $x$ is element of $X$ universe of discourse for fuzzy sets. As can be also seen in Fig. 2, the HNGFA generates the overall adaptive FLS model. Algorithm 1 represents the main part of the proposed technique. As can be seen, it demonstrates how input parameters initialize the HNGFA framework and how the final model is generated. The model parameters are evolved using two nested Genetic-Fuzzy Algorithms (GFA), namely, the Outer Genetic-Fuzzy Algorithm (OGFA) and the Inner Genetic-Fuzzy Algorithm (IGFA). The local model for OGFA is utilized to classify the major categories of attacks, as demonstrated in Algorithm 2. The parameters of this local model are evolved using two-nested GA, the outer evolves the major classes whereas the inner evolves the fuzzy rules. However, as demonstrated in Algorithm 3, the local model parameters of IGFA depend on the evolved OGFA parameters. More specifically, the OGFA assists the IGFA in training phase such that the best individual in OGFA interacts with the weak individual in IGFA to generate new solutions that enhance the prediction of mutated attacks. Algorithm 4 demonstrates how both OGFA and IGFA interact together to evolve the best rules for normal, major and minor categories of attacks through the optimization process. A threshold value is used to identify the weak IGFA chromosomes to be strengthen by the best OGFA chromosomes. This assistance is done in intelligent way such that it matches the minor category with its major category. In this regard, Fig. 3 shows the chromosomes structures in OGFA and IGFA. As can be seen, there are four GA populations collaborating to evolve the overall best classification model. The first and second populations have been dedicated for the OGFA whereas the third and fourth populations have been dedicated for the IGFA.

Algorithm 1 Hybrid Nested Genetic-Fuzzy Algorithm $HNGFA( )$

**Input:** Dataset, $Pc$, $Pm$, outerGAMaxGen, innerGAMAXGen, outerGAPop, innerGAPop.

**Output:** OGFAResults, IGFAResults.

1. **begin**
2. $(d1, d2, t1, t2) \leftarrow$ dataPreprocessing (Dataset);  
3. initializeFrameworkParameters $(d1, d2, Pc, Pm)$;  
4. $(OP1, OP2, IP1, IP2) \leftarrow$ OGFA( ); // Alg. 2
   // Extract Major and Minor Fuzzy Rules from best chromosomes
5. finalModel$ \leftarrow$ ExtractFuzzyRules $(OP1, OP2, IP1, IP2)$;  
   // Testing the final model results major rules and minor rules
6. (OGFAResults, IGFAResults) $\leftarrow$ EvaluateResults (finalModel, $t1, t2$);
7. **end**

Similarly, the parameters of IGFA local model are evolved using two-nested GA, the outer evolves the fuzzy sets whereas the inner evolves the fuzzy rules. Figure 3 shows the chromosomes structures in OGFA and IGFA showing nested populations for outer and inner GA chromosomes.

**Algorithm 2 Outer Genetic-Fuzzy Algorithm $OGFA( )$**

**Input:** $d1, d2$.

**Output:** $OP1, OP2, IP1, IP2$.

1. **begin**
2. **for** $(outerGAGenOGFA \leftarrow 1$ to outerGAMaxGen) **do**
3. **foreach** $(oChrom \in outerGAPop)$ **do**
4. **for** $(innerGAGenOGFA \leftarrow 1$ to innerGAMaxGen) **do**
5. **foreach** $(iChrom \in innerGAPop)$ **do**
6. $(OAvgBestFit, OP1, OP2) \leftarrow$ EvaluateFRB $(oChrom, iChrom)$;
7. **end**
8. UpdateGAGenomes(innerGAGenOGFA); // Alg. 6
9. **end**
10. **for**
11. $(outerGAGenIGFA \leftarrow$ outerGAGenOGFA$)$;
12. $(IAvgBestFit, IP1, IP2) \leftarrow$ IGFA( ); // Alg. 3
13. UpdateGAGenomes(outerGAGenIGFA); // Alg. 6
14. UpdateGAGenomes(outerGAGenIGFA); // Alg. 6
15. **end**
16. return $(OP1, OP2, IP1, IP2)$
17. **end**
functions selected within the rule-base. Here represents each feature via defining the membership the classification process, as in (4) and (5).

features’size selected from datasets. In fact, the encoding algorithm. This evaluation is used to calculate the accuracy of encoding the fuzzy sets are utilized, as demonstrated in Algorithm 5. The encoding scheme of fuzzy rules is represented in Algorithm 4. This demonstrated in Algorithm 6, which means that the best solution found is used to build the next generation. In other words, Elitism involves replacing the old population by copying the fittest candidates, unchanged, into the next generation. The acronyms and variables used in the algorithms are mentioned in Table 1.

Since the inner GA of both OGFA and IGFA encodes the fuzzy rules, its chromosome is designed to encode a rule-base. The encoding scheme of fuzzy rules is represented in fixed-size integer array, where the array size is equal to the features’ size selected from datasets. In fact, the encoding here represents each feature via defining the membership functions selected within the rule-base.

### Algorithm 3 Inner Genetic-Fuzzy Algorithm IGFA() 

**Input:** innerGAMaxGen, outerGAPop, innerGAPop, OP1, OP2.  
**Output:** IAvgBestFit, IP1, IP2.

1. begin
2. foreach (oChrom in outerGAPop) do
3.   if (outerGAGenIGFA > 1) then
4.     oChrom ← IntegrateChroms (oChrom, OP1); 
   // Alg. 4
5.   end if
6.   for (innerGAGenIGFA ← 1 to innerGAMaxGen) do
7.     foreach (iChrom in innerGAPop) do
8.       if (outerGAGenIGFA > 1 & & innerGAGenIGFA > 1) then
9.         iChrom ← IntegrateChroms (iChrom, OP2); 
       // Alg. 4
10.   end if
11.   (IAvgBestFit, IP1, IP2) ← EvaluateFRB (oChrom, iChrom);
12. end for
13. end for
14. UpdateGAGenomes (innerGAGenIGFA); // Alg. 6
15. end for
16. return (IAvgBestFit, IP1, IP2);
17. end

### Algorithm 4 Integrate Chromosomes IntegrateChroms() 

**Input:** iChromo, oChromo.  
**Output:** iChromo.

1. begin
2. if (fitness < ThresholdValue) then
3.   matchChromosomesOfIGFAwithOGFA();
4. end if
5. return iChromo
6. end

To evaluate and rank the fitness of the chromosomes encoding the rule-base in inner GA, chromosomes of outer GA encoding the fuzzy sets are utilized, as demonstrated in Algorithm 5. This evaluation is used to calculate the accuracy of the classification process, as in (4) and (5).

\[
\text{fitness} = \frac{1}{\text{classification error}} \quad (4)
\]

\[
\text{classification error} = 2E^2 + E + 1 \quad (5)
\]

where E is the percentage of incorrectly classified records. The classification error is represented as a quadratic formula to smoothen the curve and eliminate the division by zero problem. However, for simplicity, the fitness can be calculated using the formula \((1 - E)\). In addition, the classification error is calculated twice in the proposed technique. First, it is calculated in the inner GA of the OGFA to select the best major rules. Second, it is calculated in the inner GA of the IGFA to select the best minor rules, which finally together represent the overall model fitness.

### Algorithm 5 Evaluate Fuzzy Rules Based EvaluateFRB() 

**Input:** oChrom, iChrom  
**Output:** xAvgBestFit, OP1, OP2

1. begin
2. weightArr[] ← Empty;
3. crispOutput ← 0;
4. foreach (rec record in training dataset) do
5.   weightArr[] ← Fuzzification(rec, oChrom, iChrom);
6.   crispOutput ← defuzzification (weightArr, rec, oChrom, iChrom);
7. xAvgBestFit ← computeClassificationError(crispOutput, rec, oChrom, iChrom);
8. end for
9. return (xAvgBestFit, OP1, OP2);
10. end

For Simplicity, roulette wheel selection is utilized to select the best parents for reproduction. A single-point crossover is used randomly for every two selected pairs of chromosomes for reproduction. However, the outer GA layer chromosomes for OGFA and IGFA should remain in fixed length whereas the inner GA layer for OGFA and IGFA can be in variable size but under the limitations mentioned earlier. A random mutation is performed on a chromosome based on certain selected mutation probability. Elitism is employed, as demonstrated in Algorithm 6, which means that the best solution found is used to build the next generation. In other words, Elitism involves replacing the old population by copying the fittest candidates, unchanged, into the next generation. The acronyms and variables used in the algorithms are mentioned in Table 1.

### Algorithm 6 Update GA Genomes UpdateGAGenomes() 

**Input:** GAGeneration.  
**Output:** newGAGeneration.

1. begin
2. sortPopBasedOnFitness();
3. selectionAndCrossOver();
4. mutation();
5. replacement();
6. return newGAGeneration;
7. end

Fig. 4 shows the detailed structure of the proposed HNGFA framework. As can be seen, the n dataset features \((F_1, \ldots, F_n)\) undergo to normalization and selection processing. Both the
FIGURE 4. Detailed structure for the HNGFA framework proposed.

TABLE 1. Algorithms notations.

| Symbol       | Description                                |
|--------------|--------------------------------------------|
| \(d_1, d_2\) | Training datasets of selected features, \(d_1\) denotes major and \(d_2\) denotes minor. |
| \(t_1, t_2\) | Testing datasets of selected features, \(t_1\) denotes major and \(t_2\) denotes minor. |
| \(P_c\)     | Probability of crossover                    |
| \(P_m\)     | Probability of mutation                     |
| outerGAMaxGen | Outer maximum generation                    |
| innerGAMaxGen | Inner maximum generation                    |
| OuterGApop   | Outer Population Size.                     |
| InnerGApop   | Inner Population Size.                     |
| OAvgBestFit  | Outer Average Best Fitness                  |
| IAvgBestFit  | Inner Average Best Fitness                  |
| OP1, OP2     | Best OGFA chromosomes, 1 denotes outer GA and 2 denotes inner GA. |
| IP1, IP2     | Best IGFA chromosomes, 1 denotes outer GA and 2 denotes inner GA. |
| \(n, m\)     | Size of features in major and minor datasets. |
| \(N, M\)     | Size of rules in major and minor chromosomes. |
| \(C_k, L_i\) | \(j^{th}\) condition in \(R_i\) rule, \(i^{th}\) Label in \(R_i\) rule. |
| \(S_j\)      | \(j^{th}\) fuzzy set for the \(j^{th}\) feature. |

OFGA and IGFA are composed of two GAs. The role of the Integration Engine (IE) is to integrate between the OGFA and IGFA by combining the best outer chromosome with the best inner one and find the relation between outer and inner features. As can also be seen, the final output is collaboratively composed of \(K\) rules for detecting major categories of attacks and \(k\) rules for the detecting minor categories of attacks.

VI. RESULTS AND DISCUSSION

Several experiments have been conducted to validate and evaluate the proposed technique. All the experiments for the proposed technique have been performed on an Intel Core i7-4720HQ CPU, running windows 10 (64-bit) with 16GB RAM using C# on Microsoft Visual Studio 2017.

A. DATASETS DESCRIPTION AND PREPROCESSING

The experiments have been conducted on two publicly available benchmark datasets for NIDS researches, namely, KDDCUP99, and UNSW-NB15. Although these datasets are common in NIDS researches, they do not represent complete real-world network traffic [44].

The KDDCUP99 dataset is a subset of a larger dataset provided by the Defense Advanced Research Projects Agency (DARPA) (1998), as an operational traffic simulation for US Air Force base on Local Area Network (LAN). This dataset contains normal traffic along with multiple attacks, which classified into four major categories, namely, Probe, DoS, U2R, and R2L. These four major categories of attacks have 24 minor categories for training, and additionally, 17 other minor categories for testing. As mentioned earlier, the KDDCUP99 dataset contains 41 features, however, these features whether continuous or discrete, are classified into three groups, namely, basic features, traffic features, and content features (or host-related) [45]. In addition, this dataset contains a lot of redundant records that consequently affect classification accuracy [42]. Moreover, this dataset is outdated, and the low difficulty of its records is misleading [46], [47].

The UNSW-NB15 dataset is a recent dataset provided by Australian Centre for Cyber Security (ACCS), as a simulation to modern networks traffic. This dataset is created by a legitimate traffic tool, known as the IXIA PerfectStorm network traffic generator. This recent dataset is composed of real normal traffic along with multiple synthetic contemporary attacks, which is classified into nine categories, namely, Analysis, Fuzzers, Exploits, Backdoors, DoS, Reconnaissance, Generic, Worms, and Shellcode.

As mentioned earlier, the UNSW-NB15 dataset has 49 features, including the class label [48]. These features are classified into six groups, namely, basic features, content features, time features, flow features, labeled features, and additional generated features. The additional generated features are divided into two subgroups, namely, general-purpose features and connection features. The general-purpose features assist some features to protect the protocol service protocol whereas the connection features are used to track time features.

The feature selection stage has been performed utilizing Waikato Environment for Knowledge Analysis (WEKA) [49] on the datasets obtained after removing the redundancy from the KDDCUP99 and UNSW-NB15 training datasets. These datasets are converted into WEKA ARFF file format. In addition, feature selection has been evaluated utilizing three different approaches and different methods to obtain the best features for classification purposes.

The first approach considers common features utilized in previous researches (e.g., [50]–[53]). In this approach, the testing results have been unsatisfactory for the features extracted. The second approach considers unifying the output from some entropy and correlation-based methods such as Gain Ratio, Chi-Square test, Symmetrical Uncertainty and Correlation to find common features. In this approach, for
each feature selection algorithm, the top ranked features have been selected for the datasets of both major and minor categories of attacks, separately, along with normal traffic. Specifically, the top 10 ranked features have been utilized for the KDDCUP99 dataset whereas the top 12 ranked features have been utilized for the UNSW-NB15 dataset. The first half of the top features have been dedicated for the dataset of major categories of attacks whereas the second half of the top features have been dedicated for the dataset of minor categories of attacks, considering reparation. This approach demonstrated higher results than the first approach, but not in all selected datasets.

Finally, the third approach considers the embedded methods, which demonstrated the best results in all selected datasets. Specifically, the Elastic Net method demonstrated the best results as opposed to the LASSO method. In this method, the feature sets have been evaluated utilizing the Naive Bayes learning scheme. In addition, utilizing the selected features, cross-validation has been performed to estimate the accuracy of the learning scheme. Furthermore, the top ranked features have been selected similarly to the procedure employed earlier in the second approach.

B. PERFORMANCE METRICS

Several metrics and criteria have been adopted to measure and evaluate the performance of the proposed technique. These measures and metrics are commonly used in the literature to evaluate techniques in NIDS domain, such as the Accuracy (Acc.), Precision, Recall (or DR), FAR, F-score and the confusion matrix, as in (6), (7), (8), (9) and (10), respectively.

\[
\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}
\]

\[
\text{Precision} = \frac{TP}{TP + FP} \tag{7}
\]

\[
\text{Recall} = \frac{TP}{TP + FN} \tag{8}
\]

\[
\text{FAR} = \frac{FP}{FP + TN} \tag{9}
\]

\[
F_{\text{score}} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{10}
\]

where \(TP\), \(FP\), \(FN\), and \(TN\) are true positive, false positive, false negative, and true negative, respectively.

C. EXPERIMENTAL RESULTS AND COMPARISONS

Since the KDDCUP99 and UNSW-NB15 datasets are huge in size, only part of each dataset has been utilized in the conducted experiments. Specifically, only 10% of the KDDCUP99 dataset has been utilized whereas 20% of the UNSW-NB15 dataset has been utilized. Each of these partial datasets has been partitioned into two separate datasets, one for training phase and the other for testing phase. Since the Elastic Net method demonstrated best results in feature selection, the dataset features utilized are the features given by the third feature selection approach.

The results have been obtained from a series of experiments conducted using the developed HNGFA framework described above. The series of trial runs have been performed utilizing 8 different parameter settings, categorized into 2 configurations with respect to population size (Pop. size). Each experiment has been conducted 10 times per setting and per dataset, and the best fitness outputs have been averaged to obtain more accurate results. In the first set of experiments, a population size of 20 chromosomes with maximum generation of 5 has been used. Hence, the system has been allowed to run for 10000 generations. However, for simplicity, the results have been recorded each 8 generations to result in 1250 readings. In the second set of experiments, a similar approach has been employed but with a population size of 40 chromosomes and the results have been recorded each 16 generations to result in 2500 readings.

Since the KDDCUP99 dataset has normal traffic along with major and minor categories of attacks, the OGFA has been responsible to predict and classify normal traffic along with major categories of attacks whereas the IGFA has been responsible to predict and classify normal traffic along with minor categories of attacks. On the other hand, since the UNSW-NB15 dataset has no minor categories of attacks, the OGFA has been responsible to distinguish between normal and abnormal (i.e., attacks) traffic whereas the IGFA has been responsible to distinguish between normal traffic and major categories of attacks.

Mainly, the results obtained have been compared with two state-of-the-art techniques, namely, Fuzzy Hybrid Genetics-Based ML (FH-GBML) [54] and Genetic-Fuzzy System based on Genetic Cooperative-Competitive Learning (GFS-GCCL) [55]. All the previously mentioned techniques have been implemented in KEEL [56], which is a well-known ML tool. Finally, for further evaluation, the results have been compared with other state-of-the-art techniques. Table 2 illustrates the different parameter settings and configurations (Config.) for the conducted experiments.

![Fig. 5](image-url)

*Fig. 5 shows the results when the C1 configuration settings have been utilized in OGFA on the KDDCUP99 major categories of attacks dataset. As can be seen, the best fitness curve for S3 setting almost reached a value of 0.99 after 1100 generations. As can also be seen, the best fitness curve for S4 setting converges slower than other setting curves. This*

| Config. | Settings | Inner/Outer GA Crossover | Inner/Outer GA Mutation Rate | Pop. size |
|---------|----------|--------------------------|-----------------------------|-----------|
| C1      | S1       | 35.00%                   | 10.00%                      | 20        |
|        | S2       | 35.00%                   | 20.00%                      | 20        |
|        | S3       | 70.00%                   | 10.00%                      | 20        |
|        | S4       | 70.00%                   | 20.00%                      | 20        |
|        | S5       | 35.00%                   | 10.00%                      | 40        |
|        | S6       | 35.00%                   | 20.00%                      | 40        |
|        | S7       | 70.00%                   | 10.00%                      | 40        |
|        | S8       | 70.00%                   | 20.00%                      | 40        |
FIGURE 5. Convergence of C1 configuration settings in OGFA on the KDDCUP99 major categories of attacks dataset.

FIGURE 6. Convergence of averaged C1 configuration settings in OGFA against FH-GBML and GFS-GCCL using KDDCUP99 dataset.

FIGURE 7. Convergence of C2 configuration settings in OGFA on the KDDCUP99 major categories of attacks dataset.

is due to high rate of crossover and mutation which slow down the good solutions.

Fig. 6 shows the results of the averaged C1 configuration above against other techniques. As can be seen, the FH-GBML technique converges faster than others. However, the best fitness curve for the proposed technique almost reached the same value of 0.98 after 1100 generations. In addition, it is evident that the proposed technique outperforms the GFS-GCCL technique and has better exploration as illustrated from the fast gradually increasingly convergence.

Fig. 7 shows the results when the C2 configuration settings have been utilized in OGFA on the KDDCUP99 major categories of attacks dataset. As can be seen, the best fitness curve for S7 setting outperforms other settings and almost reached a value of 0.99 after 1250 generations. In addition, as can be shown, this curve has a fast convergence between the generations of 1000 and 1250. This is due to the high rate of crossover and low rate of mutation that result in more exploration along with maintaining good solutions, respectively.

Fig. 8 shows the results of the averaged C2 configuration settings in OGFA against other techniques using KDDCUP99 dataset. As can be seen, the best fitness curve for the HNGFA outperforms other techniques and almost reached a value of 0.98 after 1450 generations. It can also be seen that the proposed technique is better in exploration, from the fast gradually increasingly convergence.

Fig. 9 shows the results of the averaged C1 configuration settings in IGFA against FH-GBML and GFS-GCCL using KDDCUP99 dataset. As can be seen, the best fitness curve for the HNGFA outperforms other techniques after 1350 generation and almost reached a value of 0.985 at 2500 generations.

Fig. 10 shows the results of the averaged C2 configuration settings in IGFA against FH-GBML and GFS-GCCL using KDDCUP99 dataset. As can be seen, the best fitness curve of HNGFA outperforms other techniques after 1350 generation and almost reached a value of 0.985 at 2500 generations.

Fig. 11 shows the results when the C1 configuration settings
have been utilized in OGFA on the UNSW-NB19 major categories of attacks dataset. As shown, the best fitness curve for S1 setting outperforms other settings and almost reached a value of 0.92 after 950 generations. Comparing with the results from Fig. 7, the accuracy is lower at approximately the same number of generations. This is since the features utilized in UNSW-NB19 dataset are increased and most of these features are continuous, which require low rates in crossover and mutation to maintain the good solutions.

Fig. 12 shows the results of the averaged C1 configuration settings against other techniques using UNSW-NB19 dataset. As can be seen, it is evident that the HNGFA outperforms other techniques and almost reached a value of 0.91 at 1000 generations. Similar results can be obtained when comparing the HNGFA with other techniques, when utilizing averaged C2 configuration settings in OGFA.

Fig. 13 shows the results when the C1 configuration settings have been utilized in IGFA on the UNSW-NB19 minor categories of attacks dataset. As can be seen, the proposed technique outperforms other techniques and almost reached a value of 0.92 at 1150 generations. This is evident that the IGFA has been assisted by the OGFA such that the HNGFA performs better in complex datasets. Similar results can be concluded in Fig. 14, when C2 configuration settings utilized.

For more details of the performance, Table 3 shows a sample of the summarized average results, as a confusion matrix, for the experiments conducted utilizing best fitness settings in C1 and C2 configurations in OGFA on the KDDCUP99 dataset. In this table, the confusion matrix shows the number of correctly classified records when the records have been labeled only as normal traffic and attacks.

Table 4 shows the confusion matrix for normal traffic and major categories of attacks in OGFA on the KDDCUP99 dataset. As can be seen, the total number of correctly classified records is 143,752 out of 146,399, which is an accuracy rate of 98.19%.

Table 5 shows the confusion matrix for normal traffic and major categories of attacks in OGFA on the UNSW-NB15 dataset. As it is shown, the total number of correctly classified records is 50,821 out of 63,098 records, which is an accuracy rate of 80.45%. However, the developed technique has achieved the highest accuracy rate compared with FH-GBML and GFS-GCCL techniques where their accuracy rate has been 77.95% and 63.05%, respectively.

Table 6 shows the metrics evaluated for the developed technique in testing phase, namely, Precision, Recall, F-Score, FAR, and Accuracy along with their Weighted Average (W. Avg.), when the records have been labeled only as normal traffic and attacks in OGFA on the KDDCUP99 dataset. As shown, the developed technique achieved consistently a very low FAR with an accuracy rate above 98%.

Table 7 shows the metrics evaluated for normal traffic and major categories of attacks in OGFA on the KDDCUP99 dataset. As can be seen, the developed technique
achieved a very low FAR for all categories of attacks with an accuracy rate of 98.19%.

Table 8 shows the metrics evaluated for normal traffic and minor categories of attacks in IGFA on the KDDCUP99 dataset. As can be seen, for example, the weighted average FAR achieved is decreased and the precision for the normal traffic is increased, compared with Table 7. The reason is that the IGFA has been assisted by the OGFA with more features, including the candidate target class, which bring down the search space.

Table 9 shows the metrics evaluated for normal traffic and attacks in OGFA on the UNSW-NB15 dataset. As shown, the developed technique achieved an accuracy rate of 80.54%. However, as can be seen in Table 10, the IGFA enhanced the accuracy rate to reach a value of 90.24% due to the new features involved. In addition, for example, the IGFA enhanced the weighted average precision from 0.816 to 0.927. Although all minor categories of attacks have been classified with reasonable precision, Worms and Shellcode attacks have the lower precision due to the low number of their records in dataset compared with other categories.

Table 11 shows the summarized average results for the fuzzy rules evolved for normal traffic and major categories of attacks utilizing best fitness settings in C1 and C2 configurations in OGFA on the KDDCUP99 dataset, compared with other techniques. As can be seen, the developed technique has been capable of evolving rules in both configurations for detecting the U2R and R2L attacks whereas other techniques have been failed.

On the other hand, Table 12 shows the summarized average results for the fuzzy rules evolved for normal traffic and major categories of attacks utilizing best fitness settings in C1 and C2 configurations in IGFA on the KDDCUP99 dataset, compared with other techniques. As shown, the developed technique has been capable of evolving rules in both configurations for detecting all minor categories of attacks whereas other techniques have been failed.

Table 13 shows the summarized average results for the fuzzy rules evolved for normal traffic and attacks utilizing best fitness settings in C1 and C2 configurations in OGFA on the UNSW-NB15 dataset, compared with other techniques.
TABLE 11. Average results of fuzzy rules for KDDCUP99 normal traffic and major categories of attacks utilizing C1 and C2 configurations.

| Major     | HNGFA C1 | FH-GBML C1 | GFS-GCCL C1 | HNGFA C2 | FH-GBML C2 | GFS-GCCL C2 |
|-----------|----------|------------|-------------|----------|------------|-------------|
| Normal    | 7.5      | 8          | 3           | 2        | 3          | 3           |
| DoS       | 13       | 11.75      | 3           | 1        | 2          | 1           |
| Probe     | 7        | 10.75      | 2           | 2        | 1          | 3           |
| U2R       | 6        | 10.25      | 0           | 0        | 0          | 0           |
| R2L       | 9        | 8          | 0           | 1        | 0          | 0           |

TABLE 12. Average results of fuzzy rules for KDDCUP99 normal traffic and minor categories of attacks utilizing C1 and C2 configurations.

| Minor     | HNGFA C1 | FH-GBML C1 | GFS-GCCL C1 | HNGFA C2 | FH-GBML C2 | GFS-GCCL C2 |
|-----------|----------|------------|-------------|----------|------------|-------------|
| Normal    | 4.25     | 2.25       | 4           | 3        | 2          | 22          |
| back      | 2.75     | 4.5        | 0           | 0        | 0          | 0           |
| land      | 3.5      | 1.75       | 0           | 0        | 0          | 0           |
| neptune   | 1.75     | 2.5        | 2           | 1        | 2          | 1           |
| pod       | 1.5      | 3          | 0           | 0        | 0          | 0           |
| smurf     | 3        | 3.25       | 2           | 2        | 0          | 1           |
| teardrop  | 1.5      | 4.25       | 0           | 0        | 0          | 0           |
| nmap      | 3.25     | 3.25       | 1           | 2        | 2          | 1           |
| portmap   | 2.5      | 2.5        | 0           | 0        | 0          | 0           |
| satan     | 4.75     | 2.5        | 2           | 2        | 0          | 0           |
| buffer_overflow | 3.5 | 2.75 | 0 | 0 | 0 | 0 |
| loadmodule | 2.75 | 2.5 | 1 | 1.5 | 0 | 1 |
| perl      | 2.75     | 1          | 0           | 0        | 0          | 0           |
| rootkit   | 2        | 2.25       | 0           | 0        | 0          | 0           |
| ftp_write | 1.25     | 2.75       | 0           | 0        | 0          | 0           |
| guess_passwd | 3 | 2.5 | 1 | 2 | 1 | 1 |
| imap      | 2        | 2          | 0           | 0        | 0          | 0           |
| multihop  | 2.75     | 1.5        | 0           | 0        | 0          | 0           |
| phf       | 4        | 0.75       | 2           | 2.5      | 0          | 0           |
| spy       | 2.25     | 4          | 0           | 0        | 0          | 0           |
| warezclient | 4.5 | 5 | 1 | 2 | 1 | 1 |
| warezmaster | 2.75 | 2.25 | 0 | 0 | 0 | 0 |

TABLE 13. Average results of fuzzy rules for UNSW-NB15 normal traffic and attacks utilizing C1 and C2 configurations.

| Major        | HNGFA C1 | FH-GBML C1 | GFS-GCCL C1 | HNGFA C2 | FH-GBML C2 | GFS-GCCL C2 |
|--------------|----------|------------|-------------|----------|------------|-------------|
| Normal       | 21.75    | 18.5       | 4           | 4        | 4          | 5           |
| Attacks      | 24.5     | 18         | 3           | 2        | 2          | 2           |

As shown, the HNGFA has been able to analyze the dataset features deeply such that it has been capable of evolving more rules for detecting attacks, regarding these features.

Table 14 shows the summarized average results for the fuzzy rules evolved for normal traffic and minor categories of attacks utilizing best fitness settings in C1 and C2 configurations in IGFA on the UNSW-NB15 dataset, compared with other techniques.

As shown, the developed technique has been capable of evolving rules in both configurations for detecting the Shellcode and Worms attacks whereas other techniques have been failed. Table 15 shows the Accuracy and FAR metrics evaluated for normal traffic and attacks utilizing best fitness settings and same features selected on the KDDCUP99 and UNSW-NB15 datasets, compared with other techniques.

As can be seen, the HNGFA achieved higher accuracy and lower FAR compared with both FH-GBML and GFS-GCCL.

To further validate the results related to averaged best fitness values in C1 and C2 configurations, the 95% confidence interval test has been employed for each dataset and for all techniques. Table 16 shows the statistics of applying this test. As can be seen, the HNGFA produced a good confidence value with good arithmetic means compared with other techniques. Moreover, as visualized in Fig. 15, the confidence interval for testing the developed technique utilizing different datasets yielded good results.

Table 17 shows the average execution time in training and testing per instance compared with other techniques. As can be seen, the developed technique consumes more time due to the interaction between the two-nested GFA. However, the good decision-making, shown by the achieved accuracy and FAR results, justifies this increase.
For further evaluation, Table 18 shows the performance analysis of HNGFA compared with other state-of-the-art techniques in classifying major categories of attacks in KDDCUP99 dataset.

| Technique | FL [38] | RS [38] | FRS_IDS [38] | IDT [41] | FGA [40] | SVM (N-RBF) [34] | LSSVM-IDS + | FMIFS [33] | PH-GBML | HNGFA |
|-----------|---------|---------|--------------|---------|---------|----------------|------------|---------|---------|---------|
| Normal    | 0.903   | 0.998   | 0.997        | 0.99   | 0.988   | 0.997`        | 0.993`     | 0.986   | 0.981   | 0.991   |
| DoS       | 0.949   | 0.959   | 0.995        | 0.999  | 0.974   | 0.999         | 0.999`     | 0.961   | 0.949   | 0.952   |
| Probe     | 0.909   | 0.999   | 0.969        | 0.999  | 0.974   | 0.999         | 0.999`     | 0.962   | 0.949   | 0.958   |
| R2L       | 0.992   | 0.847   | 0.345        | 0.354  | 0.222   | 0.2488        | 0.2488`    | 0.307   | 0.234   | 0.999   |
| U2R       | 0.932   | 0.69    | 0.41         | 0.513  | 0.221   | 0.68          | 0.68       | 0.357   | 0.999   | 0.221   |

VII. CONCLUSION AND FUTURE WORK

In this paper, a novel Hybrid Nested Genetic-Fuzzy Algorithm (HNGFA) framework has been developed for detecting normal traffic and most of the intrusions’ categories, whether major or minor categories of attacks, particularly the categories that have rare information in datasets. Two important issues for NIDS have been considered in this work, namely, feature selection methods and building interpretable and accurate NIDS to facilitate data analysis and human understanding. The developed technique has been compared with many state-of-the-art techniques. The experimental results show that the developed technique has been able to extract accurate multilevel rule-based knowledge from network traffic, due to the effective assistance of OGFA to IGFA. In addition, in terms of the performance metrics evaluated, the results show that the HNGFA outperforms other techniques in exploration, detection and evolving rules for all minor categories of attacks with high accuracy and low FAR in different configurations on complex datasets. Moreover, the 95% confidence interval test has been applied for further validation.

The successful detection and classification of sophisticated intrusion attacks and normal network traffic provide much scope for future work. In this regard, the developed approach can be applied to other complex problem domains such as DNA computing. In addition, regarding the domain, other optimization techniques are candidate to be utilized to achieve more accurate and interpretable FLS.

REFERENCES

[1] A. Verma and V. Ranga, “Security of RPL based 6LoWPAN networks in the Internet of Things: A review,” IEEE Sensors J., vol. 20, no. 11, pp. 5666–5690, Jun. 2020.
[2] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, “Survey of intrusion detection systems: Techniques, datasets and challenges,” Cybersecurity, vol. 2, no. 1, p. 20, Dec. 2019.
[3] Q. Liu, P. Li, W. Zhao, W. Cai, S. Yu, and V. C. M. Leung, “A survey on security threats and defensive techniques of machine learning: A data driven view,” IEEE Access, vol. 6, pp. 12103–12117, 2018.
[4] E. Hodo, X. Bellekens, A. Hamilton, C. Tachtatzis, and R. Atkinson, “Shallow and deep networks intrusion detection system: A taxonomy and survey,” 2017, arXiv:1701.02145. [Online]. Available: http://arxiv.org/abs/1701.02145
[5] M. H. Ali, B. A. D. Al Mohammed, A. Ismail, and M. F. Zolkipli, “A new intrusion detection system based on fast learning network and particle swarm optimization,” IEEE Access, vol. 6, pp. 20255–20261, 2018.
[6] S. Ganapathy, K. Kulthubung, S. Muthurajkumar, M. Vijayalakshmi, P. Yogesh, and A. Kanman, “Intelligent feature selection and classification techniques for intrusion detection in networks: A survey,” EURASIP J. Wireless Commun. Netw., vol. 2013, no. 1, p. 271, Dec. 2013.
[7] A. Verma and V. Ranga, “Machine learning based intrusion detection systems for IoT applications,” Wireless Pers. Commun., vol. 111, pp. 1–24, Nov. 2019.
[8] A. L. Buczak and E. Guven, “A survey of data mining and machine learning methods for cyber security intrusion detection,” IEEE Commun. Surveys Tuts., vol. 18, no. 2, pp. 1153–1176, 2nd Quart., 2016.
[9] S. KishorWagh, V. K. Pachghare, and S. R. Kolhe, “Survey on intrusion detection system using machine learning techniques,” Int. J. Comput. Appl., vol. 78, no. 16, pp. 30–37, Sep. 2013.
[10] C.-H. Tsang, I. Gondal, P. Vamplew, and J. Kamruzzaman, “Survey of intrusion detection systems: Techniques, datasets and challenges,” IEEE Access, vol. 6, pp. 12103–12117, 2018.
[11] E. R. Hruschka, R. J. G. B. Campello, A. A. Freitas, and A. C. P. Leon, “A survey of evolutionary algorithms for clustering,” Trans. Syst., Man, Cybern. C, Appl. Rev., vol. 39, no. 2, pp. 133–155, 2nd Quart., 2016.
[12] A. Portilla-Figueras, and L. Prieto, “Short term wind speed prediction in the Internet of Things: A review,” IEEE Sensors J., vol. 20, no. 11, 2020.
[13] A. Verma and V. Ranga, “Classification of power system faults using evolutionary algorithms,” in Proc. IEEE 15th Int. Symp. Comput. Intell. Inform. (CINTI), Nov. 2014, pp. 351–368.
[14] A. P. Engelbrecht, Computational Intelligence: An Introduction, 2nd ed. Hoboken, NJ, USA: Wiley, 2007, pp. 401–479.
[15] F. Herrera, “Genetic fuzzy systems: Taxonomy, current research trends and prospects,” Evol. Intel., vol. 1, no. 1, pp. 27–46, Mar. 2008.
[16] P. J. Fleming and R. C. Purshouse, “Evolutionary algorithms in control systems engineering: A survey,” Control Eng. Pract., vol. 10, no. 11, pp. 1223–1241, Nov. 2002.
[17] S. Salcedo-Sanz, E. G. Ortiz-Garcı, Á. M. Pérez-Bellido, A. Portilla-Figueras, and L. Prieto, “Short term wind speed prediction based on evolutionary support vector regression algorithms,” Expert Syst. Appl., vol. 38, no. 4, pp. 4052–4057, Apr. 2011.
[18] E. R. Hruschka, R. J. G. B. Campello, A. A. Freitas, and A. C. P. Leon F. de Carvalho, “A survey of evolutionary algorithms for clustering,” IEEE Trans. Syst., Man, Cybern. C, Appl. Rev., vol. 39, no. 2, pp. 133–155, Mar. 2009.
[19] D. A. Van Veldhuizen, “Multiobjective evolutionary algorithms: Classifications, analyses, and new innovations,” Ph.D. dissertation, Dept. Elect. Comput. Eng., Graduate School Eng., Air Force Ins. Tech., Wright-Patterson AFB, OH, USA, May 1999.
[20] D. A. Coley, An Introduction to Genetic Algorithms for Scientists and Engineers. Singapore: World Scientific, 1999, pp. 76–91.
HASSAN ABOUNASER received the B.Sc. and M.Sc. degrees (Hons.) in computer engineering from the Arab Academy for Science, Technology & Maritime Transport (AASTMT), Cairo, Egypt, in 2002 and 2008, respectively, and the Ph.D. degree in computer engineering from Cairo University, Cairo, in 2017. He is currently an Assistant Professor with the Department of Computer Engineering, AASTMT. His M.Sc. thesis research focuses on soft-computing and autonomous vehicle robot navigation. His Ph.D. thesis research focuses on distributed data mining techniques in classification and prediction. His current research interests include soft-computing, big data, intelligent systems, network security, and image and video processing.

AMR BADR received the B.Sc. degree (Hons.) in engineering, in 1986, and the M.Sc. and Ph.D. degrees in computer science from Cairo University, in 1995 and 1998, respectively. He is currently a Professor with the Department of Computer Science, Faculty of Computers and Artificial Intelligence, Cairo University. His research interests are intelligent systems, bioinformatics, medical imaging, and P-systems. He has published more than 170 journal research articles in these areas.

* * *