2023 IEEE Symposium Series on Computational Intelligence (SSCI)
Mexico City, Mexico. December 5-8, 2023

BLB-GAFS: An Efficient, Multi-Objective Genetic Algorithm Based Feature Selection Method for Intrusion Detection Systems

Arihant Singh
The Early College at Guilford
Greensboro, USA
asingh2@guilford.edu

Kaushik Roy
North Carolina Agricultural and
Technical State University
College of Engineering
Computer Science Department
Greensboro, USA
kroy@ncat.edu

***Abstract*—Protecting Internet of Things (IoT) networks from threats is becoming increasingly important as these devices continue to grow in adoption. Modern and unseen attacks that require the analysis of more complex network traffic data for effective identification and mitigation are becoming more prevalent. Traditional machine learning approaches in current intrusion detection systems (IDS) struggle with these volumes of data, prompting exploration into the feature selection space. One class of such feature selection methods is evolutionary algorithms, in which systems mimicking real-life evolution optimize solutions for some problem. In this paper, we propose bag-of-little-bootstraps genetic algorithm feature selection (BLB-GAFS), a novel variant of the genetic algorithm feature selection method that maintains a global search of the solution space while reducing computational cost. This is accomplished with the bag-of-little-bootstraps method for approximating classifier performance. We test the BLB-GAFS technique on three modern intrusion datasets—CCD-INID-V1, detection_of_IoT_botnet_attacks_N_BaIoT, and CIRA-CIC-DoHBrw-2020—that represent updated network patterns and are highly dimensional. We found that the BLB-GAFS method matches or outperforms embedded feature selection methods on the same datasets. Furthermore, the feature sets selected by BLB-GAFS result in significantly improved multiclass precision, recall, and F1-score. Traditionally expensive wrapper feature selection methods like the genetic algorithm can be used on larger datasets through BLB-GAFS, opening the door to other applications with highly dimensional data.**

***Index Terms*—feature selection, genetic algorithm, IDS, bag-of-little-bootstraps**

I. Introduction

As Internet of Things (IoT) devices become more common in consumer and enterprise environments, the frequency and complexity of attacks on these systems have increased [1]. Consequently, solutions that identify and mitigate these attacks have become increasingly important. The primary response intended to fill this niche is the intrusion detection system (IDS) [2].

Both detecting anomalies and identifying specific attacks have been explored through machine learning (ML) IDSs, which have been successful on traditional IoT datasets [3] [4]. However, as more advanced attacks emerge, the volume of network data fed to these IDS implementations has increased in dimensionality. Standard ML approaches struggle to find meaningful relationships in these highly dimensional datasets. These solutions face long training/prediction times and reduced performance [5].

One solution to this “curse of dimensionality” is feature selection techniques. This crucial preprocessing stage is defined as the selection of a subset of the original features that better represent each class than the original set in its entirety [6]. Feature selection for IDSs is a well-explored topic in the current literature. In the past, various works have highlighted that feature selection before training results in less resource consumption and greater interpretability [7] [8].

Feature selection techniques are often split into three categories: wrapper, filter, and embedded methods. Wrappers use the evaluation of a classifier to score different subsets of features in search of the ideal. These methods involve far more computation than the other two varieties depending on the search strategy used. Filters do not rely on a classifier, instead selecting the optimal subset of features based on intrinsic relationships in the data. These methods are usually faster than wrapper selection, but they do not generate subsets that are directly tied to classification performance. Embedded techniques refer to the selection of an optimal feature subset in the training process of the classifier. These methods are often faster than wrapper methods and more performant than filter methods, but they are specific to the classifier at hand [6].

One class of search strategies used in wrapper feature selection is evolutionary algorithms like genetic algorithms (GA), particle swarm optimization (PSO), and ant colony optimization (ACO) [9]. These methods enable a global, random search of the entire solution space while narrowing in on the most promising subsets. This behavior avoids the extreme computational cost of an exhaustive search while still exploring the whole space for ideal subsets. Evolutionary algorithms have been studied extensively in IDS feature selection [10] [11].

GAs are of great interest for this feature selection problem.

978-0-7381-4408-5/23/$31.00 ©2023 IEEE

Varieties of GAs are applied to numerous optimization problems across real-world applications [12]. They are popular due to their ability to survey large solution spaces and handle problems with multiple solutions. This capability is particularly important for this application, considering the large volumes of network traffic data needed to identify unseen attacks means large feature sets are becoming more common. In the feature selection case, GAs have produced better results with fewer features than similar techniques [13].

Arguably the most important component of a GA is its fitness function, which describes the performance of each individual—or feature subset—in each generation. Many researchers employ classification accuracy as a component of the fitness function in their work [14] [15]. In some cases, the classification performance is combined with the number of features or another metric to frame the problem as multi-objective optimization. The issue with this approach is that, on larger datasets with many features, training and evaluating a new classifier for each feature subset is an expensive process.

One alternative to training and testing on the entire, large dataset to evaluate classifiers is the bootstrap and other resampling-based methods. Specifically, the bag of little bootstraps (BLB) procedure is tailored for estimating quantities on big data with low storage and computational complexity. By combining features of the bootstrap and subsampling, BLB is suitable for approximating classifier performance quickly—ideal for the fitness function that is evaluated many times during a GA.

In this work, we explore feature selection on modern, highly dimensional IoT attack datasets using a GA with this BLB approximation for classifier performance. Our contributions are:

- developing BLB-GAFS, a novel multi-objective genetic algorithm-based feature selection method that incorporates BLB classifier performance estimation, and
- experimenting on highly-dimensional datasets that include recent IoT attacks—namely the detection_of_IoT_botnet_attacks_N_BaIoT, CIRA-CIC-DoHBrw-2020, and CCD-INID-V1 datasets [16] [17] [18].

The rest of this paper is organized as follows. In Section II, we discuss relevant works in the feature selection and evolutionary algorithms space. In Section III, we detail our proposed feature selection approach. Section IV introduces the three datasets and explains our experimental setup. Section V presents our experimental results. In Section VI, we summarize our work and conclude our discussion of the proposed approach.

II. Related Work

As mentioned in the introduction, feature selection has been explored exhaustively for machine learning applications across many fields. Since the highly diverse traffic observed in most network environments necessitates large feature sets to capture enough information, feature selection is crucial for IDSs. Many previous works have attempted to fill this need with different combinations of feature selection and classification methods.

Evolutionary and nature-inspired feature selection methods have been applied to intrusion detection in the past [19] [20]. Mehmood and Rais use the ant colony optimization (ACO) algorithm for feature selection on the KDD99 dataset before classifying with SVM [21]. Peng et al. test their ACO-based feature selection algorithm on the KDD99 dataset [22]. Li et al. apply the particle swarm optimization (PSO) algorithm in combination with random forest for feature selection on the KDD99 dataset [23].

The GA feature selection approach has also been explored in the current literature. Sindhu et al. apply a GA for feature selection on the KDD Cup 99 dataset using a fitness function that incorporates the sensitivity and specificity of a neural network ensemble as the fitness function [24]. This ensemble approach is then used to classify the selected set of features. Guha et al. use a GA to narrow down the features on the NSL-KDD Cup and UNSW-NB15 datasets before classifying with an artificial neural network [25]. The chromosomes in their GA represented individual feature sets, and they used classification accuracy from a multimodal neural network for the fitness function. Tao et al. apply a GA to optimize the feature set and parameters used for SVM to classify on the KDD Cup 99 dataset [26]. They used chromosomes that included both SVM parameters and feature weights during the evolution, with K-fold cross-validation accuracy serving as the fitness function. Khammassi and Krichen combine the accuracy of a logistic regression classifier with the number of features selected into a single fitness function to select features [27]. They attempted classification with decision trees on the KDD Cup 99 and UNSW-NB15 datasets.

While these two works are examples of single-objective optimization problems, GA feature selection has also been explored with multi-objective fitness functions. Khammassi and Krichen examine the performance of a feature selection method based on the NSGA-II multi-objective optimization algorithm to select the ideal feature subsets from the NSL-KDD, UNSW-NB15, and CIC-IDS2017 datasets [28]. Like the single-objective variants, they included classification accuracy as one component of the fitness function. However, they also included the number of features in the subset as the second objective.

De La Hoz et al. also experimented with NSGA-II feature selection, instead using the Jaccard's coefficient between the ground truth and predicted labels for each class as the objectives [29]. Rather than including an additional objective focused on the reduction of the subset size like the previous work, this approach focused on optimizing the subset for each of the classes in the NSL-KDD dataset before classifying with growing hierarchical self-organizing maps.

In this paper, we suggest a multi-objective, NSGA-II feature selection approach with a fitness function consisting of a BLB estimate of classification performance and the number of features represented by the individual.

III. Proposed Approach

The proposed feature selection solution implements the fast NSGA-II optimization algorithm for selecting the ideal feature subset [30]. The fitness function for this algorithm includes an estimate of classification performance using BLB and the total number of features represented by the individual. For each dataset, the optimal feature set is fed into various traditional ML classifiers—namely, naïve Bayes (NB), random forest (RF), logistic regression (LR), support vector machine (SVM), and k-nearest neighbors (KNN)—and the classification performance is evaluated.

The first step in BLB-GAFS is to raise a population representing a group of random feature subsets. Each of these "individuals" represents a feature subset with an array of binary values that has a length equivalent to the original number of features in the dataset. The binary values themselves indicate whether the corresponding feature is included in the subset. Much like actual chromosomes, different expressions of these "genes" can result in wildly different feature sets. For BLB-GAFS, we raised an initial population of 10 and created 20 "children" in each generation.

Then, fitness values are found for each individual in the initial population. The fitness function in this stage returns two objectives. The first objective approximates the classification performance of a naïve Bayes classifier on the included features. This approximation is calculated using the BLB method which is similar to the bootstrap and other resampling techniques. First presented by Kleiner et al., this method averages some performance metric found on bootstrapped samples from multiple small subsets of the original dataset [31]. Specifically, BLB samples s subsets of size b from the original dataset with size n then resamples n points from each subset. A classifier is then trained and evaluated on r of these resamples, yielding an estimate of the chosen performance metric through Monte Carlo approximation. Algorithm 1 details this approximation process. The performance metric of choice for BLB-GAFS is the ROC-AUC score since it measures the quality of model predictions regardless of the classification threshold.

The computational benefits of this method come from the fact that each resample can only contain b distinct points while having a size of n. Thus, each resample can be generated by drawing an array of counts from an n-trial uniform multinomial distribution of b objects. Then, the resamples can be represented as the b distinct items along with these sampled counts—meaning that each resample only needs $O(b)$ storage space rather than $O(n)$. This weighted representation can be passed directly to the naïve Bayes classifier (as well as other common estimators).

Algorithm 1 BLB score approximation

```
Require: Data $X_i, \ldots, X_n$
Ensure: An estimate of classifier performance on $X$
  for $j = 1$ to $s$ do
    Randomly sample a set $\mathcal{I} = i_1, \ldots, i_b$ of $b$ indices from $1, \ldots, n$
    for $k = 1$ to r do
      Sample $(n_1, \ldots, n_b) \sim$ Multinomial$(n, 1_b/b)$
      Train classifier with $(X_{n_1}, \ldots, X_{n_b})$
      $Q_k \leftarrow$ classifier score
    end for
    $P_j \leftarrow r^{-1} \sum_{k=1}^{r} Q_k$
  end for
  return $s^{-1} \sum_{j=1}^{s} P_j$
```

The original paper presents some techniques for selecting the hyperparameters b, s, and r in BLB [31]. They found that a value of $b = n^{\gamma}$ where γ is 0.7 was suitable for most problems. They also suggested an adaptive technique for selecting the values of s and r. r is to be increased until the computed values from the resamples have ceased to fluctuate, as determined with a convergence assessment algorithm presented in the paper. Similarly, s will be increased until the computed values from each subset seem to converge. This adaptive hyperparameter selection is shown to reduce unnecessary computation done by BLB after reaching the solution and is implemented in BLB-GAFS as a result.

By avoiding training and predicting on large sections of the original dataset, the BLB approximation method is very storage and compute efficient compared to cross-validation or even a single train-test split. Since the subsample and resample sizes in BLB are relatively small, the method lends itself to more straightforward parallel computing–which, although not explored in this study, could further reduce how long the approximation takes. When realized over the course of many individuals across many generations of the GA, the performance gain attached to BLB enables the usage of BLB-GAFS on large datasets with many features.

After evaluating each generation and selecting the top individuals with NSGA-II, the mutation and two-point crossover operations are carried out. In mutation, each bit in the individual has a 20% probability of being flipped, turning the feature on or off. In two-point crossover, two points are randomly chosen from the parent chromosomes and the bits between them are flipped. These techniques are used to create the next generation, which then has its fitness evaluated. The entire BLB-GAFS process is described by the flowchart in Fig. 1.

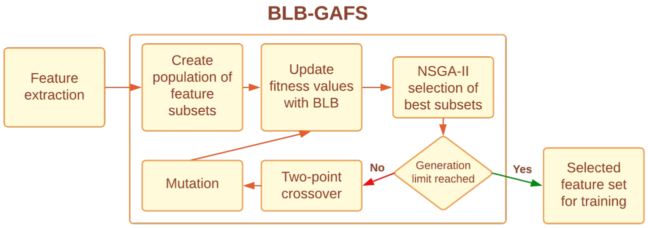
BLB-GAFS
Feature extraction
Create population of feature subsets
Update fitness values with BLB
NSGA-II selection of best subsets
Generation limit reached
No
Yes
Two-point crossover
Mutation
Selected feature set for training

Fig. 1. Description of BLB-GAFS system

IV. Datasets and Setup

A. CCD-INID-V1

This dataset was developed by the researchers in [18]. It focuses on transmissions between edge devices and cloud servers, like in the smart lab and smart home settings. The NFStream library was used to engineer features. The dataset contains 91,665 instances and 81 features. It includes five frequently used attacks: ARP poisoning, ARP DoS, UDP flood, Hydra bruteforce with Asterisk protocol, and SlowLoris.

B. detection_of_IoT_botnet_attacks_N_BaIoT (BaIoT)

This dataset was created by the group in [16]. They collected real traffic data from nine commercial IoT devices infected with botnets from the Mirai and BASHLITE families. The features were extracted from this captured data using autoencoders. The dataset includes 7,062,606 instances and 115 features. There are ten attacks from two families: BASHLITE Scan, Junk, UDP, TCP, and COMBO along with Mirai Scan, Ack, Syn, UDP, and UDPplain.

C. CIRA-CIC-DoHBrw-2020 (DoHBrw)

This dataset was generated by the researchers in [17]. There are two layers of traffic flows: DoH and non-DoH HTTPS at the first layer, and benign/malicious DoH at the second. Mozilla Firefox and Google Chrome are used to simulate the non-DoH HTTPS and benign DoH traffic. dns2tcp, DNSCat2, and Iodine are used to simulate the malicious DoH traffic. The DoHMeter tool is used to extract statistical features from this captured traffic. The dataset includes 1,159,241 instances and 34 features. The three tools used for the malicious DoH traffic make up the attack classes in this dataset.

D. Experimental Setup

Our experiments were conducted on a machine equipped with an AMD Ryzen 5 3600 CPU, 16 GB of RAM, and an NVIDIA RTX 3070 through the Windows Subsystem for Linux platform. Since all of our selected datasets had some measure of class imbalance, we applied the imbalanced-learn library to each for better performance [32]. We used the scikit-learn library to scale the datasets, implement our GA feature selection as a Transformer, and train/evaluate our machine learning models [33]. To speed up training and inference for our classifiers, we used the RAPIDS cuML library to run them on the GPU.

For each dataset, we first split them into training and testing sets with an 80-20 ratio (80% training, 20% testing). We then used the edited nearest neighbors approach in the imbalanced-learn library to balance the classes, creating separate binary and multiclass datasets in the process.

To preprocess the CCD-INID-V1 dataset, we first used cuML's target encoding on the categorical features in the dataset. The last step was to encode the target labels—0 or 1 for binary classification, and 0 through 5 for multiclass attack detection. In the BaIoT dataset, several of the nine devices are missing multiple attack types. To ensure that we could test on as many attack types as possible, we selected the Danmini Doorbell (that contains all 10 attacks). After filtering down to this device, there were 1,018,298 rows left. We wrapped up preparation by encoding the targets—again, 0 or 1 for binary classification and 0 through 10 for multiclass attack detection. The DoHBrw dataset also did not take long to preprocess. We first used target encoding on the categorical features. Then, we labeled the targets 0 or 1 for binary classification and 0 through 3 for multiclass attack detection.

To evaluate the different approaches on each dataset, we will examine classification metrics including precision, recall, and F1-score. Additionally, we will review the ROC-AUC score for the binary case.

V. Results

Fig. 2, Fig. 3, and Fig. 4 show the evolution of our GA in the feature selection stage prior to classification on all three of the datasets. These plots are from the multiclass attack detection case. They demonstrate how the highest ROC-AUC score in each generation rises while the number of features selected falls. While the feature-space gains are mostly gradual, the performance gains show the GA escaping local minima during the search.

Following the feature selection process, we were left with 42/115 features on BaIoT, 4/34 features on DoHBrw, and 16/81 features on CCD-INID-V1 in the binary case. In the multiclass case, the GA selected 33/115 features on BaIoT, 17/34 features on DoHBrw, and 17/81 features from CCD-INID-V1. The differences between the anomaly detection and attack identification behaviors in the GA were interesting. While selecting more features on the DoHBrw and CCD-INID-V1 datasets for the more difficult multiclass task, the GA selected fewer features for the BaIoT dataset.

In Table I, we present the binary classification metrics for our various ML models evaluated following feature selection. We include a comparison to the XCNN and RCNN approaches presented in Liu et al., as that work also attempted feature selection on these three datasets [18]. For the XCNN/RCNN

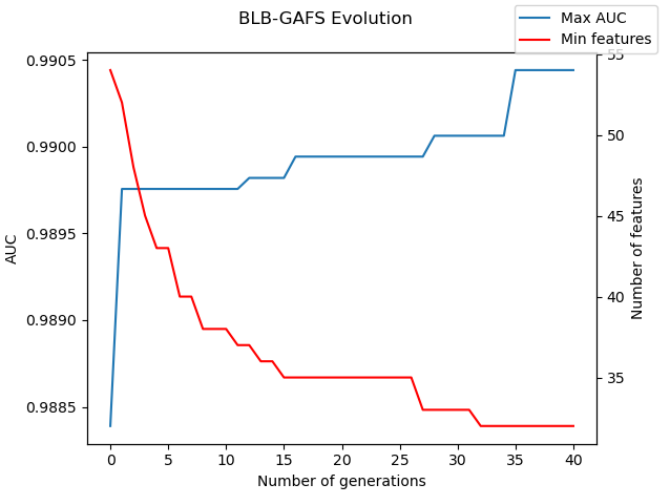

Fig. 2. Evolution of BLB-GAFS during feature selection on BaIoT

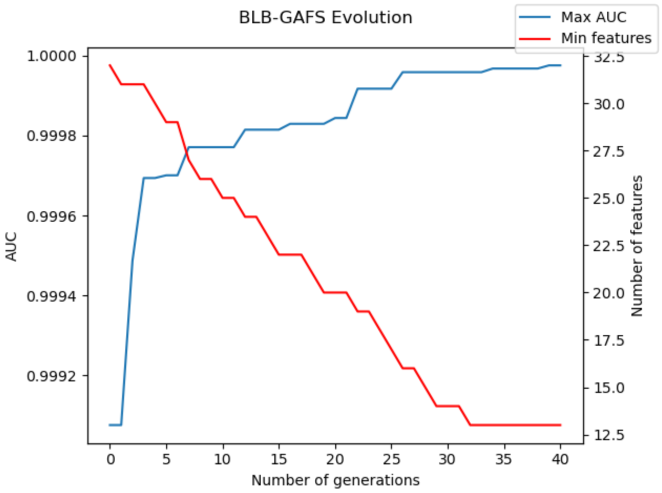

Fig. 3. Evolution of BLB-GAFS during feature selection on CCD-INID-V1

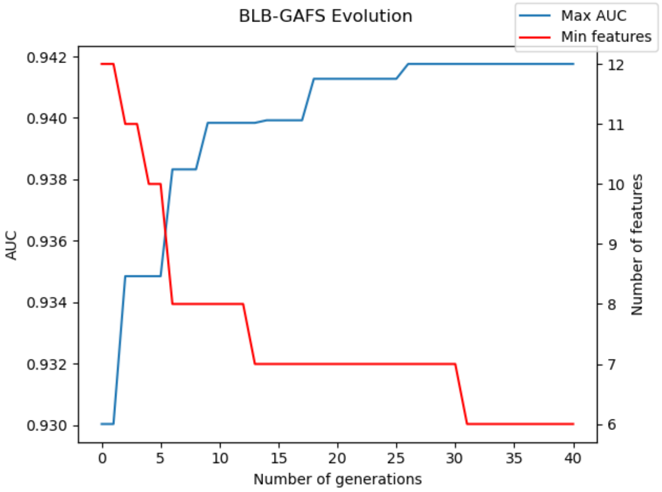

Fig. 4. Evolution of BLB-GAFS during feature selection on DoHBrw

models, this work used embedded feature selection with XGBoost and Random Forest. In the anomaly detection task, our feature selection technique paired with traditional ML models resulted in excellent performance across all three datasets. Our classifiers match or exceed the XCNN and RCNN in ROC-AUC score for all cases except for the naïve Bayes classifier on the BaIoT dataset. There were minute differences in ROC-AUC score between the classifiers besides the RCNN on the CCD-INID-V1 dataset. In general, every classifier was successful in anomaly detection on these datasets.

We present the multiclass classification results in Table II. In multiclass detection, there is a far more obvious discrepancy between our classifiers and the XCNN/RCNN. In every dataset, the GA feature selection-based models outperform the deep learning techniques in precision, recall, and F1-score. Especially in the BaIoT dataset with the largest original feature space, BLB-GAFS selected features that resulted in higher performance than the embedded feature selection methods used for RCNN and XCNN, respectively. This could be attributed to the more model-agnostic nature of the evolutionary search approach relative to embedded methods. While those techniques select features that are important for the Random Forest and XGBoost models specifically, BLB-GAFS can find a feature set that is optimal regardless of the classifier.

TABLE I
Binary Results

Dataset	Model	Precision	Recall	F1-score	ROC-AUC
CCD-INID-V1	RCNN	0.96	0.96	0.96	0.956
	XCNN	0.99	0.99	0.99	0.998
	GA-NB	1.00	1.00	1.00	0.999
	GA-RF	1.00	1.00	1.00	0.999
	GA-LR	1.00	1.00	1.00	0.999
	GA-SVM	1.00	1.00	1.00	0.999
	GA-KNN	1.00	1.00	1.00	0.999
BaIoT	RCNN	1.00	1.00	1.00	0.999
	XCNN	0.99	0.99	0.99	0.999
	GA-NB	1.00	1.00	1.00	0.998
	GA-RF	1.00	1.00	1.00	0.999
	GA-LR	1.00	1.00	1.00	0.999
	GA-SVM	1.00	1.00	1.00	0.999
	GA-KNN	1.00	1.00	1.00	0.999
DoHBrw	RCNN	0.99	0.99	0.99	0.986
	XCNN	1.00	1.00	1.00	0.999
	GA-NB	1.00	1.00	1.00	0.999
	GA-RF	1.00	1.00	1.00	0.999
	GA-LR	1.00	1.00	1.00	0.999
	GA-SVM	1.00	1.00	1.00	0.999
	GA-KNN	1.00	1.00	1.00	0.999

TABLE II
Multiclass Results

Dataset	Model	Precision	Recall	F1-score
CCD-INID-V1	RCNN	0.09	0.21	0.11
	XCNN	0.77	0.35	0.29
	GA-NB	1.00	1.00	1.00
	GA-RF	0.99	0.99	0.99
	GA-LR	0.96	0.96	0.96
	GA-SVM	0.79	0.77	0.77
	GA-KNN	0.85	0.83	0.83
BaIoT	RCNN	0.01	0.09	0.02
	XCNN	0.02	0.09	0.03
	GA-NB	0.76	0.70	0.64
	GA-RF	0.92	0.89	0.86
	GA-LR	0.82	0.83	0.79
	GA-SVM	0.78	0.79	0.76
	GA-KNN	0.88	0.88	0.83
DoHBrw	RCNN	0.16	0.25	0.19
	XCNN	0.65	0.47	0.51
	GA-NB	0.69	0.73	0.69
	GA-RF	0.95	0.96	0.96
	GA-LR	0.79	0.81	0.80
	GA-SVM	0.73	0.74	0.73
	GA-KNN	0.94	0.95	0.94

VI. Conclusions

In this work, we experimented with a novel, BLB-based fitness function for NSGA-II feature selection for intrusion detection. We found that BLB-GAFS's performance approximation approach avoided repeatedly training and testing classifiers on large datasets, saving computationally. We also observed that our feature selection system achieved high precision, recall, and F1-score on various intrusion detection tasks, including multiclass classification. This GA-based feature selection could have applications in other cases where big data needs preprocessing for traditional ML algorithms to succeed.

References

[1] S. Arisdakessian, O. A. Wahab, A. Mourad, H. Otrok, and M. Guizani, "A Survey on IoT Intrusion Detection: Federated Learning, Game Theory, Social Psychology, and Explainable AI as Future Directions," *IEEE Internet of Things Journal*, vol. 10, pp. 4059–4092, Mar. 2023.

[2] C. Amali Pushpam and J. Gnana Jayanthi, "Systematic Literature Survey on IDS Based on Data Mining," in *Second International Conference on Computer Networks and Communication Technologies* (S. Smys, T. Senjyu, and P. Lafata, eds.), vol. 44, pp. 850–860, Cham: Springer International Publishing, 2020. Series Title: Lecture Notes on Data Engineering and Communications Technologies.

[3] A. Nisioti, A. Mylonas, P. D. Yoo, and V. Katos, "From Intrusion Detection to Attacker Attribution: A Comprehensive Survey of Unsupervised Methods," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3369–3388, 2018.

[4] Y. Xin, L. Kong, Z. Liu, Y. Chen, Y. Li, H. Zhu, M. Gao, H. Hou, and C. Wang, "Machine Learning and Deep Learning Methods for Cybersecurity," *IEEE Access*, vol. 6, pp. 35365–35381, 2018.

[5] A. Lehavi and S. Kim, "Feature Reduction Method Comparison Towards Explainability and Efficiency in Cybersecurity Intrusion Detection Systems," in *2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA)*, (Nassau, Bahamas), pp. 1326–1333, IEEE, Dec. 2022.

[6] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of machine learning research*, vol. 3, no. Mar, pp. 1157–1182, 2003.

[7] Y. Liu, Z. Xu, J. Yang, L. Wang, C. Song, and K. Chen, "A Novel Meta-Heuristic-Based Sequential Forward Feature Selection Approach for Anomaly Detection Systems," in *2016 International Conference on Network and Information Systems for Computers (ICNISC)*, (Wuhan, China), pp. 218–227, IEEE, Apr. 2016.

[8] R. B and G. S, "An Intelligent Fuzzy Rule based Feature Selection for Effective Intrusion Detection," in *2018 International Conference on Recent Trends in Advance Computing (ICRTAC)*, (Chennai, India), pp. 206–211, IEEE, Sept. 2018.

[9] B. Xue, M. Zhang, W. N. Browne, and X. Yao, "A Survey on Evolutionary Computation Approaches to Feature Selection," *IEEE Transactions on Evolutionary Computation*, vol. 20, pp. 606–626, Aug. 2016.

[10] H. Gharaee and H. Hosseinvand, "A new feature selection IDS based on genetic algorithm and SVM," in *2016 8th International Symposium on Telecommunications (IST)*, (Tehran, Iran), pp. 139–144, IEEE, Sept. 2016.

[11] M. Di Mauro, G. Galatro, G. Fortino, and A. Liotta, "Supervised feature selection techniques in network intrusion detection: A critical review," *Engineering Applications of Artificial Intelligence*, vol. 101, p. 104216, May 2021.

[12] C. A. Coello, "An updated survey of GA-based multiobjective optimization techniques," *ACM Computing Surveys*, vol. 32, pp. 109–143, June 2000.

[13] R. Leardi, R. Boggia, and M. Terrile, "Genetic algorithms as a strategy for feature selection," *Journal of chemometrics*, vol. 6, no. 5, pp. 267–281, 1992. Publisher: Wiley Online Library.

[14] A. Fatima, R. Maurya, M. K. Dutta, R. Burget, and J. Masek, "Android Malware Detection Using Genetic Algorithm based Optimized Feature Selection and Machine Learning," in *2019 42nd International Conference on Telecommunications and Signal Processing (TSP)*, (Budapest, Hungary), pp. 220–223, IEEE, July 2019.

[15] H. Boubenna and D. Lee, "Feature selection for facial emotion recognition based on genetic algorithm," in *2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*, (Changsha, China), pp. 511–517, IEEE, Aug. 2016.

[16] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher, and Y. Elovici, "N-BaIoT—Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders," *IEEE Pervasive Computing*, vol. 17, pp. 12–22, July 2018.

[17] M. MontazeriShatoori, L. Davidson, G. Kaur, and A. Habibi Lashkari, "Detection of DoH Tunnels using Time-series Classification of Encrypted Traffic," in *2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, (Calgary, AB, Canada), pp. 63–70, IEEE, Aug. 2020.

[18] Z. Liu, N. Thapa, A. Shaver, K. Roy, M. Siddula, X. Yuan, and A. Yu, "Using Embedded Feature Selection and CNN for Classification on CCD-INID-V1—A New IoT Dataset," *Sensors*, vol. 21, p. 4834, July 2021.

[19] S. Harde and V. Sahare, "Design and implementation of ACO feature selection algorithm for data stream mining," in *2016 International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT)*, (Pune, India), pp. 1047–1051, IEEE, Sept. 2016.

[20] S. Fong, R. Wong, and A. V. Vasilakos, "Accelerated PSO Swarm Search Feature Selection for Data Stream Mining Big Data," *IEEE Transactions on Services Computing*, vol. 9, pp. 33–45, Jan. 2016.

[21] T. Mehmood and H. B. M. Rais, "SVM for network anomaly detection using ACO feature subset," in *2015 International Symposium on Mathematical Sciences and Computing Research (iSMSC)*, (Ipon, Perak, Malaysia), pp. 121–126, IEEE, May 2015.

[22] H. Peng, C. Ying, S. Tan, B. Hu, and Z. Sun, "An Improved Feature Selection Algorithm Based on Ant Colony Optimization," *IEEE Access*, vol. 6, pp. 69203–69209, 2018.

[23] H. Li, W. Guo, G. Wu, and Y. Li, "A RF-PSO Based Hybrid Feature Selection Model in Intrusion Detection System," in *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*, (Guangzhou), pp. 795–802, IEEE, June 2018.

[24] S. S. Sivatha Sindhu, S. Geetha, and A. Kannan, "Decision tree based light weight intrusion detection using a wrapper approach," *Expert Systems with Applications*, vol. 39, pp. 129–141, Jan. 2012.

[25] S. Guha, S. S. Yau, and A. B. Buduru, "Attack Detection in Cloud Infrastructures Using Artificial Neural Network with Genetic Feature Selection," in *2016 IEEE 14th Intl Conf on Dependable, Autonomic and Secure Computing, 14th Intl Conf on Pervasive Intelligence and Computing, 2nd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*, (Auckland), pp. 414–419, IEEE, Aug. 2016.

[26] P. Tao, Z. Sun, and Z. Sun, "An Improved Intrusion Detection Algorithm Based on GA and SVM," *IEEE Access*, vol. 6, pp. 13624–13631, 2018.

[27] C. Khammassi and S. Krichen, "A GA-LR wrapper approach for feature selection in network intrusion detection," *Computers & Security*, vol. 70, pp. 255–277, Sept. 2017.

[28] C. Khammassi and S. Krichen, "A NSGA2-LR wrapper approach for feature selection in network intrusion detection," *Computer Networks*, vol. 172, p. 107183, May 2020.

[29] E. De La Hoz, E. De La Hoz, A. Ortiz, J. Ortega, and A. Martínez-Álvarez, "Feature selection by multi-objective optimisation: Application to network anomaly detection by hierarchical self-organising maps," *Knowledge-Based Systems*, vol. 71, pp. 322–338, Nov. 2014.

[30] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 182–197, Apr. 2002.

[31] A. Kleiner, A. Talwalkar, P. Sarkar, and M. Jordan, "The Big Data Bootstrap," 2012. Publisher: arXiv Version Number: 1.

[32] G. Lemaître, F. Nogueira, and C. K. Aridas, "Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning," *Journal of Machine Learning Research*, vol. 18, no. 17, pp. 1–5, 2017.

[33] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.