Intrusion Detection In-Vehicle Networks using Bio-Inspired Approaches

Thales Guimaraes Marques, Ana M. B. C. Quevedo,
Antonio H. G. Leoncio Junior
State University of Ceara (UECE), Fortaleza, CE, Brazil
{thales.guimaraes,ana.quevedo,helder.leoncio}@aluno.uece.br

Carlos H. O. O. Quevedo and Joaquim Celestino Jr.
State University of Ceara (UECE), Fortaleza, CE, Brazil
{carlos.oria,joaquim.celestino}@uece.br

***Abstract*—Vehicular Ad Hoc Networks (VANETs) are significant for providing services, applications, and communication to vehicles, connecting them to the outside world and supporting Intelligent Transportation Systems (ITS). Inside the car, the in-vehicle network (IVN), also known as CAN (Controller Area Network), handles communication between Electronic Control Units (ECUs) and sensors, ensuring the vehicle's functionality and safety. However, external connections make this internal network vulnerable to unauthorized and malicious access. This work presents a comparative study of two bio-inspired meta-heuristics (Bat and Ant Colony Optimization) for use in Intrusion Detection System (IDS) models. It focuses on machine learning-based classifiers to detect and classify anomalous and malicious traffic on the CAN bus.**

***Index Terms*—CAN, Controller Area Network, Machine Learning, Bat Bio-inspired Algorithm, IDS, IoT, VANET, feature selection.**

I. Introduction

Modern vehicles have evolved from simple mechanical machines into complex systems with advanced technologies. They now feature telematics units with wireless and Global Positioning System (GPS) connectivity, enabling intelligent systems to detect issues autonomously. In-Vehicle Networks (IVNs) connect various subsystems, allowing interaction with external devices through wireless communications protocols and through the On-Board Diagnostics Port (OBD-II). While this connectivity improves functionality, it also introduces security risks.

The Controller Area Network (CAN) is a communication standard used in automotive, industrial, and Internet of Things (IoT) applications. It facilitates efficient data exchange between controllers and devices through specific rules. Key features include priority-based identifiers, dominant and recessive bit states, non-destructive arbitration, and fault tolerance. These characteristics ensure reliable real-time communication, even in congested networks.

Despite the benefits of using OBD-II, sensors and actuators in the electronic components (Electronic Control Units - ECUs), and wireless communications that enhance vehicle efficiency for Intelligent Transportation Systems (ITS), security remains highly vulnerable, making these components targets for malicious actions.

Threats and vulnerabilities in Vehicular Ad Hoc Networks (VANETs) are greater than in conventional wired networks. In addition to traditional attacks like Man in the Middle, Packet Sniffing, and Denial of Service (DoS), VANETs face specific threats aimed at disrupting safety and spreading false information. [1]

IoT represents a global network infrastructure of interconnected devices and systems, enabling seamless communication and interaction across various domains. As described in [2], IoT integrates advanced technologies such as sensors, actuators, and network protocols to monitor, manage, and optimize systems in real-time. A key application of IoT is in Smart Grids, where it enhances data collection, analysis, and predictive management for energy systems. Additionally, [3] highlights the transformative potential of IoT in connecting devices to create smarter, more efficient environments, addressing both operational efficiency and security challenges.

According to [4], IoT is essential in the vehicular industry through VANETs, a subset of IoT. However, VANETs face significant security issues due to the rapid adoption of IoT in vehicles, leading to increased cyber threats and delays in detection. AI techniques, particularly Machine Learning (ML) and bio-inspired algorithms like Ant Colony Optimization (ACO) and Bat algorithms (BA), are key for threat detection and mitigation. Feature Selection (FS) techniques enhance network data processing, improving communication system efficiency, accuracy, and reliability. This paper utilizes IoT-based methods to enhance intrusion detection in vehicular networks.

Additionally, ML bio-inspired algorithms offer effective solutions to complex problems by finding optimal outcomes. These algorithms are gaining prominence in the meta-heuristics field due to their ability to learn and adapt like biological organisms, attracting scientific attention for solving increasingly complex problems in dynamic, constrained environments.

This paper compares the Bat and ACO algorithms for detecting anomalies and malicious attacks in IVNs using bio-inspired ML techniques. By selecting key features, these algorithms identify sensor behavior patterns in CAN systems, effectively distinguishing normal message exchanges from anomalous ones.

This study focuses on improving detection accuracy, scalability, and the reliability of CAN networks to enhance service protection and security measures.

979-8-3315-0694-0/25/$31.00 ©2025 IEEE

The main contributions of this paper are as follows:

1) A comparative study of the Bat and ACO algorithms for anomaly detection in CAN systems.
2) The implementation of bio-inspired feature selection techniques to improve the efficiency and accuracy of ML classifiers.
3) An evaluation of the performance of two supervised ML classifiers: K-Nearest Neighbors (KNN) and Extremely Randomized Trees (Extra Trees), for intrusion detection in CAN networks.
4) A demonstration of the scalability and robustness of bio-inspired algorithms in identifying malicious messages in automotive environments.

The remainder of this paper is organized as follows: Section II presents the existing related work. Section III introduces the proposed data IDS mechanisms, while Section IV describes the results of the experiments performed. Finally, Section V concludes the paper and presents future work.

II. Related Works

In recent years, the security of IVNs, particularly in CAN, has become a critical area of focus due to the heightened vulnerability of modern connected vehicles to cyber-attacks. Various studies have introduced techniques for detecting anomalies, failures, and cyber-attacks within the CAN environment. This summary highlights the most pertinent approaches in the literature, emphasizing their primary detection methodologies.

The study [5] proposes a novel intrusion detection method for vehicular networks utilizing survival analysis. It addresses the rising cybersecurity challenges stemming from the integration of IT and wireless communication in modern vehicles. Key contributions include a generalized intrusion detection approach and the evaluation of multiple attack scenarios on the CAN system.

The authors in [4] propose an anomaly detection method for IoT networks using ACO and the Genetic Algorithm (GA) for feature selection. Their D-ACO/GA system successfully reduces the feature space while maintaining high detection accuracy, with ACO achieving 99.37% accuracy and GA achieving 98.86%. The main contribution is the method's ability to lower computational complexity, enhancing the performance of intrusion detection systems in IoT environments.

In [6], the authors improve security in CAN environments by introducing a Variable Length Message Authentication Code (MAC). Using the Improved CAN Data Reduction (ICANDR) algorithm, they compress data to accommodate the MAC without increasing network load or altering the CAN frame format. The key contribution is dynamically adjusting the MAC length based on data size, ensuring confidentiality and integrity while reducing busload, supported by the Advanced Encryption Standard 128 (AES-128) and Hash-based Message Authentication Code (HMAC) for authentication.

The study [7] introduces CANet, a framework using unsupervised deep learning techniques, including Long Short-Term Memory (LSTM) networks and autoencoders, to capture the temporal dynamics of CAN signals. Intrusion detection is achieved through signal reconstruction and anomaly identification based on deviation scores. CANet is the first deep learning model capable of handling the high-dimensional structure of CAN bus data with diverse message types and intervals. It surpasses traditional ML methods in detecting both known and unknown intrusions, such as signal replay, flooding, and signal suppression attacks.

This review of related works demonstrates the evolution of intrusion detection techniques in CAN networks, transitioning from traditional rule-based and statistical methods to modern deep learning architectures that can handle high-dimensional, complex data with varying signal structures. These contributions have paved the way for more effective anomaly detection like bio-inspired meta-heuristics.

III. Intelligent bio-inspired meta-heuristics

A. BAT Mechanism

The Bat Algorithm (BA), inspired by bats' echolocation behavior, uses the principles of sound pulse emission and echo analysis to determine object distances and positions, mimicking their ability to navigate and hunt in darkness [8]. In BA, artificial bats are characterized by position, velocity, and frequency vectors, which are iteratively updated to explore the search space and refine solutions within the continuous domain.

According to [8], some assumptions about bat echolocation are made:

1) Echolocation is a universal ability in bats, allowing them to determine the distance to food sources, prey, and obstacles, and differentiate between elements in their environment.
2) When searching for prey, bats perform exploratory flights with constant speed (v_i) and emission frequency, while gradually adjusting the wavelength (or frequency), pulse amplitude, and pulse emission rate (r) based on the target proximity.
3) The amplitude of sound can vary but generally ranges from high (A_0) to low values (A_{min}), highlighting the bats' ability to adapt their sound emissions to their environment.

The bat algorithm's pseudo-code begins with randomly creating bats (line 02), where each bat has its own characteristics such as pulse emission rate (r_i), pulse volume (A_i), frequency (f_i), and speed (v_i). These attributes are set randomly within predefined limits. Line 03 sets the initial frequency (f_i) for position (x_i). The pulse rate (r_i) and amplitude (A_i) are initialized (line 04), and from lines 06 to 15, the bats evolve over time. Their frequencies, speeds, and positions are updated, and solutions (the positions of the bats) are evaluated using an objective function, assigning the best-performing bat to position (x_*).

At each iteration t, the bat parameters are updated, therefore, a new frequency f_i, speed v_i and position x_i for each individual i, are modified based on the following equations [8]:

TABLE I
RELATED WORK

Reference	Technique	Advantages	Disadvantages
[5]	Survival analysis applied to CAN messages in vehicular networks	High accuracy,low computational cost, can do real-time detection	May not address all types of attacks, Performance could vary
[4]	D-ACO/GA for Feature Selection in IoT Anomaly Detection	High accuracy, Suitable for IoT	Longer processing time for large datasets
[6]	Variable Length MAC for CAN	Reduces CAN busload, no modifications to CAN protocol	Additional complexity due to compression and dynamic MAC size
[7]	LSTM and autoencoders to model CAN bus data.	handling multiple CAN signals, high accuracy.	High computational resources for training the neural networks.

$$f_i = f_{\min} + (f_{\max} - f_{\min})\beta \tag{1}$$

$$v_i^{t+1} = v_i^t + (x_i^t - x_*)f_i \tag{2}$$

$$x_i^{t+1} = x_i^t + v_i^t \tag{3}$$

The range f_{min} and f_{max} represents the frequency of each bat. $\beta \in [0, 1]$ is a value randomly generated based on a normal distribution. After updating the frequency and speed parameters, the pulse emission rates for each bat are checked. In line 09 of the algorithm, a comparison is made between the pulse rate r_i and a randomly generated value from a normal distribution. If r_i is lower than the random value, it suggests that bat i is likely at a certain distance from its prey x_i. Thus, local exploration is performed by selecting a promising solution and making slight adjustments to generate a new solution for bat i (line 11). In step 13, a random solution is generated, and in the next step, several conditions are examined to determine its feasibility. If the conditions are favorable, the newly generated solutions are validated (line 14), the rate r_i is increased, and the amplitude A_i is reduced (line 15). Finally, the bats are ranked, and based on this ranking, the most successful bat is selected (line 17).

```
Algorithm 1 BAT Algorithm
Objective function $f(x)$, $x = (x_1, ..., x_d)^T$
Initialize bat population $x_i$ ( $i = 1, 2, ..., n$) and velocity $v_i$
Define the pulse frequency $f_i$ at $x_i$
Initialize pulse rates $r_i$ and loudness $A_i$
begin
    for ($t <$ Maximum number of iterations) do
        Generate new solutions by adjusting frequency, updating velocities, and locations/solutions [equations (2.1) to (2.3)]
        if ($rand < r_i$) then
            Select a solution among the best solutions Generate a local solution around the selected best solution
        end
        Generate a new solution by flying randomly
        if ($rand < A_i$ & $f(x_i) < f(x^*)$) then
            Accept the new solutions
            Increase $r_i$ and reduce $A_i$
        end
        Rank the bats and find the best one
    end
end
```

Fig. 1. BAT Algorithm

The primary goal of the BA is to find the optimal solution to a given problem by adjusting its pulse rate and sound intensity to effectively navigate the search space. The following parameters are key to the algorithm's performance and should be carefully tuned:

- Number of Bats (n): This affects the diversity of the search. A low number can result in insufficient exploration, while a high number may increase runtime. Experiment to balance exploration and exploitation.
- Pulse Rate (r): Controls the intensity of the bats' movement. Higher values allow for broader movements, but values too low can cause premature convergence, while values too high may lead to over-searching.
- Intensity (A): Determines how strong a bat emits a 'call', that is to attract others. Higher values encourage exploration, but can also cause large jumps.
- Pulse Emission Rate (α): Controls the likelihood of a bat emitting a pulse. Higher values increase exploration but may make the search less focused.
- Position Update Rate (γ): Defines how often the bats update their positions. Lower values favor exploration, while higher values focus on local exploration.
- Stopping Criterion: Set a stopping condition, such as a maximum number of iterations or minimal improvement in the solution.

Key Considerations:

- Search Space Size: Larger spaces require more bats to enhance exploration and effectively cover the solution space.
- Problem Complexity: Complex problems benefit from higher pulse rates and intensities, enabling better movement and focus on promising solutions.
- Local Minima: A higher position update rate (γ) and local optimization techniques (e.g., 2-opt) help escape local minima.
- Runtime: Increasing parameter values can result in longer runtime.
- Exploration vs. Exploitation: Adjusting pulse rates and intensities balances global exploration with focused local search.

The BA is a bio-inspired meta-heuristic leveraging bat echolocation to address optimization problems. Its adaptable parameters, including pulse emission rate and sound amplitude, enable effective balancing of global and local search. This flexibility ensures robust solutions, making it well-suited for anomaly detection and complex optimization challenges.

B. Ant Colony Optimization (ACO) Mechanism

In [4], to ensure IoT security, it is essential to reprocess and scrutinize data during transmission to remove non-conforming or redundant information that could compromise security. Feature Selection (FS) techniques are used to process the vast amount of data in the network, enhancing the efficiency and accuracy of the communication system and preventing errors and system downtime. In this study, ACO is used as feature selection approach (FS) to reduce the dimensionality of the dataset and identify relevant features without compromising the prediction accuracy.

According to the study of [4], ACO is motivated by the collective behavior of real ants as they search for food, using the same principles of cooperation and collaboration to explore the search space. The agents in this algorithm employ the bio-nspired strategy based on the principle that ants operate individually with simple rules. During their journeys, they deposit pheromones, and collectively, they have the ability to detect variations in the concentration of these pheromones in the vicinity. As a result, they tend to move in the direction where the concentration is higher. The functioning of ACO can be comprehended through [4] work.

```
Algorithm 2 ACO Algorithm [4]
Input: matrix of distances between the features
Output: the best subset of features
Procedure ACO:
    while not termination condition is met do
        foreach interaction in the application do
            init of the parameters: $\alpha$, $\beta$, Q, p, and $\tau_0 \geq m = n$
              – number of ants is equal to the number of towns
              for generated ant population do
                  calculate partial fitness for each ant
            end
            bestsolutions ← partial fitness
        end
        update pheromone trails
    end
```

Fig. 2. ACO Algorithm

C. Proposed Technique

The IDS is designed to detect malicious messages within the CAN bus of in-vehicle environments in autonomous and/or connected cars. It utilizes bio-inspired meta-heuristics and ML techniques to enhance detection efficiency. This mechanism can be implemented on the same bus as other ECUs, and can function as a Firewall ECU, capable of evaluating the data packets travelling on the CAN bus and classify them as attack packets or normal packets. In [9] presents locations for deploying IDS on the CAN bus system. It can be deployed at the CAN, or ECU or in the gateways. The IDS will passively "listen" to the communication in search of malicious content. The aim of the system is to analyse the message exchanges between the sensors that control the car's vital functions. The proposed system selects characteristics of a CAN bus sensoring data set of In-vehicle messages from a chosen car, and applies ML techniques to classify them into attack or normal categories. To do this, KNN and ExtraTrees classifiers are used. The performance of the mechanism is evaluated using these two ML algorithms. As demonstrated in the model outlined in Figure 3.

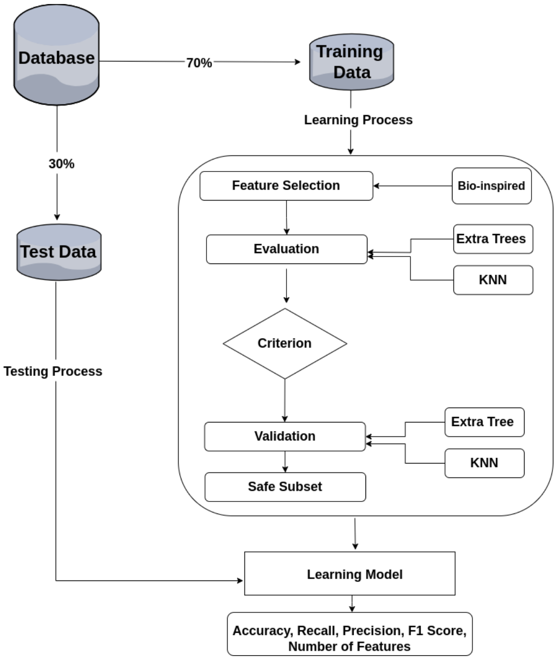

Fig. 3. The Bio-inspired Model

D. Data Processing

In this paper we used the same datasets studied at [5] however in this study we concatenate the 9 files used in the previous work

According to [10] Data science is commonly defined as a methodology by which actionable insights can be inferred from data. The representation of complex environments by rich data opens up the possibility of applying all the scientific knowledge we have regarding how to infer knowledge from data. In this study we want to discover patterns of malicious behavior in IVNs environments.

The problem of finding a function

$$H(x) : \mathbb{R}^d \to \mathbb{K} \tag{4}$$

that maps an input space in R onto a discrete set of k target outputs or classes K = 1,..., k. In this setting, the features are arranged as a vector x of d real-valued numbers.

Still in [4], the effectiveness of the learning algorithm relies heavily on the nature of the input data it receives. To enhance the quality of the data, several data pre-processing techniques are employed, including transformation, normalization, and sampling. From the ECUs devices, a diverse array of data is gathered, containing numerous features. However, not all of these features contribute meaningfully to the classification task; and some are considered redundant and irrelevant. Consequently, these superfluous and non-informative features are removed. Additionally, numerical feature values within the dataset may exhibit fluctuations. To mitigate this, a normalization process is applied to confine these values to a range between 0 and 1. The dataset may exhibit an imbalance, with instances of intrusive traffic significantly outnumbering instances of normal traffic. This imbalance adversely impacts the classifiers' performance, leading to overfitting and distortions in the model.

To address the discrepancy in class frequency within the training dataset, in this work, we used the ADASYN (Adaptive Synthetic Sampling) Oversampling technique [11]. This technique is an enhancement of SMOTE (Synthetic Minority Oversampling Technique). In ADASYN, the density distribution r_i is used as the decision criterion for the number of synthetic data points to be generated for the minority class. This criterion places greater emphasis on observations that are harder to generalize. The calculation of ADASYN can be represented as equations:

$$d = \frac{m_s}{m_l} \tag{5}$$

$$G = (m_l - m_s) \times \beta \tag{6}$$

$$r_i = \frac{\Delta_i}{K}, \quad i = 1, \ldots, m_s \tag{7}$$

$$\hat{r_i} = \frac{r_i}{\sum_{i=1}^{m_s} r_i} \tag{8}$$

$$g_i = \hat{r_i} \times G \tag{9}$$

The d represents the imbalance ratio, used to assess whether applying the technique is necessary. Here, m_s and m_l represent the sample sizes of the minority and majority classes, respectively. G represents the number of synthetic data points generated, where $\beta \in [0, 1]$ regulates the balance level.

The sample weight r_i is calculated where Δ_i represents the number of examples in the K-nearest neighbors of x_i that belong to the majority class. This weight is normalized using the formula $\hat{r_i}$, where $\hat{r_i}$ represents the density distribution.

Finally, g_i represents the number of synthetic data points generated for each minority class observation. If a minority observation has few neighbors, it will be located in a region where the minority class is underrepresented.

Utilizing the feature subset derived from the feature selection phase, we conduct classification and detection tasks using a ML classifier. In this research, our primary focus is on binary classification, specifically centered on assessing the likelihood of an attack occurrence. The classifiers employed in this study encompass K-Nearest Neighbors (KNN) and Extra Trees. We present here an overview of these classifiers.

- KNN is a widely used supervised ML classifier that classifies data based on the proximity of instances using statistical measures. The ”K” refers to the number of nearest neighbors considered in the classification process. Instances with similar features are grouped into the same class, and KNN labels new data points based on patterns learned from previously labeled data.
- The Extremely Randomized Trees (Extra Trees) algorithm is a supervised ensemble learning method similar to random forests but often faster. It builds multiple decision trees using random sampling without replacement, creating unique datasets for each tree. Additionally, a random subset of features is selected for each tree. The key distinction of Extra Trees is its use of randomly chosen split values for features, instead of computing optimal splits. This randomness increases the diversity and reduces the correlation between the trees, enhancing model performance.

IV. Results

This section shows the evaluation of both BAT-IDS and ACO. In Figure 4, we compare ACO and BAT-IDS in terms of the number of features selected per iteration, while 5 shows the accuracy achieved by the classifiers. In 4, the number of features selected by ACO and BAT-IDS varies over 25 iterations. ACO, represented by the blue line, shows greater variability in the number of features selected, ranging from 9 to 11 features across different iterations. In contrast, BAT-IDS, represented by the red line, demonstrates more stability, selecting between 8 and 10 features over the iterations.

This behavior reflects the nature of each algorithm: ACO tends to explore a broader solution space, resulting in more fluctuation in feature selection, whereas BAT-IDS offers a more consistent selection process due to its more controlled parameter adjustment. In Figure 5, the classification accuracy of the Extra-Trees and KNN classifiers is displayed for the features selected by both ACO and BAT-IDS.

The findings suggest that both classifiers are highly accurate, achieving values above 0.94 across all instances. When it comes to features selected by ACO, KNN slightly edges out with an accuracy of 0.9993, compared to Extra-Trees at 0.9989. On the other hand, for the BAT-IDS-selected features, Extra-Trees performed slightly better, reaching an accuracy of 0.9481, compared to 0.9460 for KNN.

These small differences suggest that KNN may benefit more from the features selected by ACO, while Extra-Trees is better suited to the feature sets generated by BAT-IDS. Overall, both BAT-IDS and ACO, when combined with the classifiers, exhibit strong performance in detecting malicious messages within the CAN bus of autonomous and connected

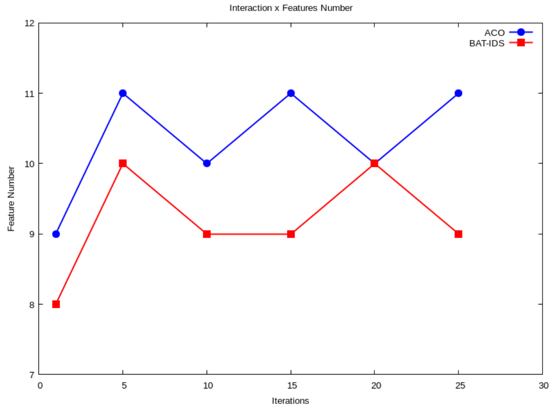

Fig. 4. Features selection x iteractions.

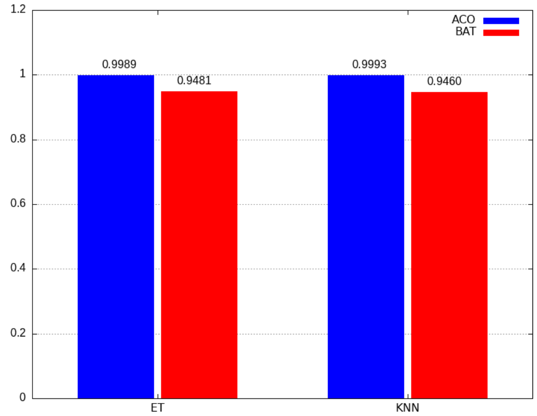

Fig. 5. Attack Classification

vehicles, making them suitable choices for intrusion detection systems. The differences in feature selection patterns and classifier performance highlight the potential for optimizing these methods depending on specific operational requirements.

The Figure 6 shows how the performance of the ACO and BAT algorithms improves as iterations progress. In the initial stages, both algorithms (ACO in blue and BAT in red) show an increase in accuracy. As iterations continue, they reach near-perfect accuracy levels, with both stabilizing by the third iteration.

This performance indicates that both ACO and BAT quickly attain high accuracy levels within just a few iterations. This rapid convergence and subsequent stabilization suggest that the algorithms effectively adapt to the selected features, ensuring robust intrusion detection. Thus, their reliable performance confirms their suitability for applications like monitoring vehicle systems.

V. Conclusion

As modern vehicles become increasingly connected, the CAN bus faces significant vulnerabilities, making intrusion detection essential for ensuring the safety and reliability of in-vehicle networks. In our research, we explored two innovative, nature-inspired algorithms: Bat Algorithm and ACO. The study examined how these algorithms can be applied to IDS in CAN bus networks. By employing ML techniques in conjunction with these bio-inspired algorithms, we were able to enhance feature selection, improve classification accuracy, and demonstrate robust detection of anomalies and malicious messages.

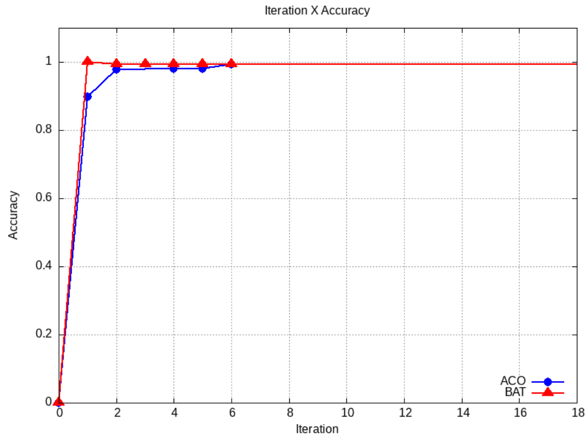

Fig. 6. Accuracy.

Our results show that both algorithms achieve high accuracy rates in detecting CAN bus intrusions, with ACO slightly outperforming Bat in terms of feature selection variability, while Bat exhibits more consistent performance. The empirical investigations demonstrate that these algorithms exhibit a significant degree of adaptability across various operational scenarios, rendering them as highly viable candidates for practical implementations of IDS within autonomous and interconnected vehicular environments.

As future work, we aim to explore additional bio-inspired algorithms to further improve feature selection and classification accuracy in CAN networks. Expanding the analysis to include different types of attack scenarios and datasets is another key direction. Furthermore, implementing the proposed methodology on dedicated hardware platforms, such as embedded systems, will allow for real-time evaluation and validation in practical vehicular environments.

References

[1] C. H. Quevedo, A. M. Quevedo, G. A. Campos, R. L. Gomes, J. Celestino, and A. Serhrouchni, “An intelligent mechanism for sybil attacks detection in vanets,” in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*. IEEE, 2020, pp. 1–6.

[2] A. Ghasempour, “Internet of things in smart grid: Architecture, applications, services, key technologies, and challenges,” *Inventions*, vol. 4, no. 1, pp. 1–12, 2019.

[3] IBM, “What is the iot?” 2024, accessed: 2024-11-20. [Online]. Available: https://www.ibm.com/topics/internet-of-things

[4] A. H. G. Júnior, J. Celestino, and G. Campos, “D-aco/ga-a bio-inspired strategy for feature selection in anomaly traffic detection in smart internet of things environments,” in *2024 International Conference on Information Networking (ICOIN)*. IEEE, 2024, pp. 47–52.

[5] M. L. Han, B. I. Kwak, and H. K. Kim, “Anomaly intrusion detection method for vehicular networks based on survival analysis,” *Vehicular communications*, vol. 14, pp. 52–63, 2018.

[6] Y.-J. Kim and J.-G. Chung, “Variable length mac for can security protocol,” in *2020 International SoC Design Conference (ISOCC)*. IEEE, 2020, pp. 272–273.

[7] M. Hanselmann, T. Strauss, K. Dormann, and H. Ulmer, “Canet: An unsupervised intrusion detection system for high dimensional can bus data,” *Ieee Access*, vol. 8, pp. 58 194–58 205, 2020.

[8] X.-S. Yang, “A new metaheuristic bat-inspired algorithm,” in *Nature inspired cooperative strategies for optimization (NICSO 2010)*. Springer, 2010, pp. 65–74.

[9] I. Studnia, E. Alata, V. Nicomette, M. Kaâniche, and Y. Laarouchi, “A language-based intrusion detection approach for automotive embedded networks,” *International Journal of Embedded Systems*, vol. 10, no. 1, pp. 1–12, 2018.

[10] L. Igual and S. Seguí, “Introduction to data science,” in *Introduction to Data Science: A Python Approach to Concepts, Techniques and Applications*. Springer, 2024, pp. 1–4.

[11] H. He, Y. Bai, E. A. Garcia, and S. Li, “Adasyn: Adaptive synthetic sampling approach for imbalanced learning,” in *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*. Ieee, 2008, pp. 1322–1328.