scientific reports

Check for updates

OPEN

Intrusion detection in metaverse environment internet of things systems by metaheuristics tuned two level framework

Milos Antonijevic[1], Miodrag Zivkovic[1], Milica Djuric Jovicic[2], Bosko Nikolic[3], Jasmina Perisic[1], Marina Milovanovic[1], Luka Jovanovic[1], Mahmoud Abdel-Salam[4] & Nebojsa Bacanin[1,5,6]✉

Internet of Things (IoT) is one of the most important emerging technologies that supports Metaverse integrating process, by enabling smooth data transfer among physical and virtual domains. Integrating sensor devices, wearables, and smart gadgets into Metaverse environment enables IoT to deepen interactions and enhance immersion, both crucial for a completely integrated, data-driven Metaverse. Nevertheless, because IoT devices are often built with minimal hardware and are connected to the Internet, they are highly susceptible to different types of cyberattacks, presenting a significant security problem for maintaining a secure infrastructure. Conventional security techniques have difficulty countering these evolving threats, highlighting the need for adaptive solutions powered by artificial intelligence (AI). This work seeks to improve trust and security in IoT edge devices integrated in to the Metaverse. This study revolves around hybrid framework that combines convolutional neural networks (CNN) and machine learning (ML) classifying models, like categorical boosting (CatBoost) and light gradient-boosting machine (LightGBM), further optimized through metaheuristics optimizers for leveraged performance. A two-leveled architecture was designed to manage intricate data, enabling the detection and classification of attacks within IoT networks. A thorough analysis utilizing a real-world IoT network attacks dataset validates the proposed architecture's efficacy in identification of the specific variants of malevolent assaults, that is a classic multi-class classification challenge. Three experiments were executed utilizing data open to public, where the top models attained a supreme accuracy of 99.83% for multi-class classification. Additionally, explainable AI methods offered valuable supplementary insights into the model's decision-making process, supporting future data collection efforts and enhancing security of these systems.

Keywords Metaverse, CatBoost, LightGBM, Optimization, Metaheuristics algorithms, Chimp optimization algorithm

The Internet of Things (IoT) integrates together physical objects into the digital world, transforming how the users engage with both realities within the emerging and evolving landscape of the Metaverse[1]. IoT networks drive the development of novel virtual ecosystems across sectors like smart cities, healthcare and entertainment. Thanks to IoT, data is autonomously collected, processed, and shared without interruption[2]. The Metaverse enhances this connection by supporting immersive experiences, personalized interactions, and real-time decision-making[3]. Being a crucial part of the Metaverse, IoT leverages traditional networks into highly interconnected environments, promoting innovation and redefining user experiences by blending together real-world interactions with virtual opportunities. Thus, one of the requirements is to provide reliable operation of these networks, along with high level of availability[4].

Personal IoT networks, consisting of wearables, smart home systems, and AR/VR gadgets, provide users with unprecedented levels of convenience and control over their Metaverse experiences. These devices allow establishment of a tangible connection between an individual's virtual environment or avatar and their physical

[1]Singidunum University, 11000 Belgrade, Serbia. [2]Innovation Centre, School of Electrical Engineering, University of Belgrade, 11000 Belgrade, Serbia. [3]School of Electrical Engineering, University of Belgrade, 11000 Belgrade, Serbia. [4]Faculty of Computer and Information Science, Mansoura University, Mansoura 35516, Egypt. [5]Saveetha School of Engineering, SIMATS, Thandalam, Chennai, Tamilnadu 602105, India. [6]Sinergija University, Bijeljina 76300, Bosnia and Herzegovina. ✉email: nbacanin@singidunum.ac.rs

surroundings, supporting more inherent and advanced management of virtual spaces. The swift expansion of IoT is propelling the evolution of the Metaverse, breaking the limits of connectivity and merging the physical and digital worlds into a smooth and immersive experience[5,6].

However, this rapid expansion of IoT within the Metaverse faces significant challenges including protecting interconnected devices that handle sensitive user data, and mitigating real-time cyber threats that could disrupt immersive experiences[7]. Primarily, IoT devices in general are highly vulnerable to cyberattacks because of their limited processing capabilities and reliance on basic systems[8,9]. This vulnerability is even more critical within the Metaverse, where essential virtual and physical infrastructures are managed by interconnected systems. Potential malicious users may exploit these weaknesses and disrupt online healthcare services, cause financial losses in Metaverse commerce, or gain unauthorized access to personal data streams, smudging the thin line between virtual and real-world consequences. Therefore, innovative security frameworks are essential for ensuring a secure and immersive Metaverse experience for all users. These solutions must hit the balance between the lightweight architecture of IoT gadgets and robust security measures, like advanced encryption methods and real-time updates[10–12].

The principal constraints of traditional security solutions may be summed up as the difficulty to keep up to date with dynamic and swiftly changing Metaverse environment. They are not adaptable enough to cancel out novel emerging threats like botnets attacks[13], attempting to exploit the vulnerabilities of the complex correlations among the real and digital worlds within the Metaverse, as they are mostly designed to be reactive. On the other hand, cybersecurity solutions combined with artificial intelligence (AI) provide considerably more adaptable and data-driven defensive options[14,15]. AI-fueled solutions are capable of analyzing immense datasets in real-time, allowing identification of trends and drifts in risk to prevent damage before it happens. This is vital for maintaining the robustness of the expanding Metaverse, providing users safe and continuous experience while they are exploring and producing new virtual contents.

Despite numerous advantages, AI faces some weaknesses as well. Primarily, inadequate quality of data, ill-judged algorithms and incompetently chosen hyperparameter configurations. Consequently, models trained by inadequate quality datasets may result in unreliable outcomes, highlighting the necessity of high quality data for proper training. Alternatively, selection of the appropriate machine learning (ML) models is crucial, as various methods have tendency to perform unalike regarding of the challenge being solved and utilized dataset. Hyperparameters' configuration, like number of layers, learning rate or dropout can additionally heavily impact the model's performance, and must be carefully optimized to achieve optimal outcomes. Wolpert's no free lunch (NFL) assumption[16] discloses non-existance of all-round solution that works well for all classification challenges. As a result, models have to be selected and adapted to each specific task. Nevertheless, optimizing hyperparameters is broadly recognized as an NP-hard optimization challenge due to its inherent complexity. A key challenge for AI scientists is determining the appropriate hyperparameter configuration in such situations, as it is computationally infeasible. Conventional optimization algorithms regularly fall short in these scenarios, as they struggle to deliver the desired outcomes within a tolerable time frame. One potential answer is to utilize metaheuristics algorithms, capable to scan immense solution spaces to deliver approximate solutions. These methods are well-suited for addressing complex real-world challenges where finding exact answers is impractical.

This paper proposes a framework consisting of two levels, galvanized by the architecture explored in the previous research[17]. Convolutional neural network (CNN) is utilized in the primary layer of the architecture, and assigned the role to extract the features. As outlined by other relevant previous publications[18–20], significant improvements in the performance of CNN can be introduced with replacement of the ultimate dense CNN's layer by other classifiers like AdaBoost or XGBoost. Consequently, this study takes similar approach, by feeding the intercepted output of the final CNN's data processing layer to the inputs of the second level of framework, where CatBoost and LightGBM classifiers are used for further improvement of the classification capability of the architecture, especially for high volume massive streams of data generated by Metaverse IoT networks, necessitating real time processing. Moreover, configuration of both levels of framework is optimized by metaheuristics algorithms, assigned to tune the hyperparameters of regarded models. This approach ensures achieving the finest possible outcomes of the proposed combined framework. Generally speaking, the proposed methodology maximizes the benefits provided by both deep learning and ensemble approaches, where metaheuristics algorithms warrant the proper configuration of models' hyperparameters for achieving superior performance.

An altered version of chimp optimization algorithm (ChOA)[21] was used in this research to tune the hyperparameters of both layers of the framework to ensure good performance. ChOA metaheuristics was selected after careful experimentation with different optimizers, since NFL[16] elaborates that a ubiquitous optimizer that could deliver the best performance for all optimization problems does not exist. Despite the existence of other powerful optimizers like crayfish optimization algorithm (COA)[22], red fox optimizer (RFO)[23] and reptile search algorithm (RSA)[24], elementary version of ChOA rallied astounding results over the smaller scale simulations, and it was consequently selected for auxiliary modifications that would allow reaching even more desirable outcomes for intrusion classification problem.

With respect to all presented facts, primary contributions of this research may be delineated along the following lines:

- A proposition of the novel two-level AI framework for enhancing Metaverse IoT network security.
- Framework comprised of combination of CNN and boosting ensemble classifiers to perform threat classification in IoT networks.
- A proposition of a modified optimization metaheuristics tailored for the problem in hand, building upon the baseline ChOA, that was employed to tune the framework's models.

- The top-performance models were subjected to Explainable AI to establish the relative importance of the features and their effect on forecasts made by the system.This study is arranged in the following units. Section "Related works" puts forth the related works on this matter along with the utilized techniques' foundations. Next section "Methods" delineates the baseline ChOA metaheuristics, and showcases the altered version of algorithm that was later employed in the experiments. The settings of the experimental environment required for reproducibility of the simulations are set forth in Section "Experimental setup", while simulation outcomes of all simulations that were carried out are delineated and discussed in Section "Results". Ultimately, Section "Conclusion and future work" delivers the concluding remarks and suggests possible ways forward in the future research.

Related works

Conventional systems used for network protection, that revolve around firewall and blacklist solutions, have very constrained capabilities. They are not flexible enough, depending of the collection of rules and human interventions to adjust to the novel attacks. Moreover, they can only be upgraded with novel attacking patterns after the system was already breached. In other words, they are capable of responding to the events that have already happened in the past. This drawback makes conventional systems ineffective when encountering zero-day attack and emerging menaces, leaving the networks vulnerable and open to sophisticated cyber-threats. Many approaches were used from early 2000s[25], typically divided into intrusion detection systems (IDS) and intrusion prevention systems (IPS). Nowadays, a wide spectrum of tools is openly available for security of the systems, including firewall and antivirus applications, however, their restricted functionality leaves them open to novel types of threats.

One way to handle these novel types of threats that emerge each day revolves around integration of AI and IoT security applications. Generally speaking, AI couples seamlessly with IoT networks for different purposes as evidenced by numerous practical implementations[26,27]. The role of AI in this scenario is to enhance the security of IoT networks through identification of anomalous behavior in real time, where ML models are utilized to detect and classify possible threats from normal traffic. Hybrid ML solutions tailored specifically for IoT security challenge have been introduced by papers such as[28], highlighting their superiority in threat detection across various IoT devices and architectures. More focused research, such as[29], explored intrusion detection specifically within healthcare-related IoT networks, employing ML classifiers adjusted by hybrid metaheuristics techniques. While these studies showcased the significant potential of ML models, they also emphasized the challenges associated with selecting the appropriate hyperparameters, which is crucial to achieve optimal performance.

Optimizing the hyperparameters of ML models is essential for achieving optimal results and maximizing effectiveness, not only within cybersecurity but across various other fields. Poor tuning often leads to model failure and underperformance. A significant portion of recent research focuses on hyperparameters tuning for various ML structures utilizing metaheuristics algorithms[30,31]. This applies to the IoT intrusion detection problems as well, where hybrid approaches where ML models were tuned by metaheuristics delivered promising outcomes[32,33].

Despite recent progress in this field, a significant research gap remains. While metaheuristics-tuned ML models have been explored to some extent for IoT networks and intrusion detection, the focus has primarily been on models like XGBoost and AdaBoost, with limited investigation into LightGBM tuning. Additionally, the two-level framework suggested within this research, which combines a CNN with CatBoost and LightGBM classifiers and uses metaheuristics techniques to tune both levels, has not been previously studied for the observed challenge. Furthermore, the dataset[34] employed within the experiments, published in 2023, has yet to be thoroughly explored.

The remainder of this section yields brief background of the techniques utilized in this research, by providing basics of CNNs, CatBoost and LightGBM classifying models, followed by a short overview of metaheuristics approaches along with their prosperous applications.

Convolutional neural networks

Convolutional neural networks[35] are famous of their image classification and object detection capabilities, but they also excel in other tasks. Inspirited by the mammal visual cortex structure, they follow similar layered architecture. Input data passes through all layers in a particular order, making use of transfer activation functions like ReLu, tanh and sigmoid for mapping of the non-linear output.

To construct a deep CNN, it is essential to include a convolutional layer along with nonlinear, pooling, and fully connected layers[36]. For the provided input data, multiple filters skid over the convolutional layer, producing an output as the sum of element-wise multiplication of each filter and the receptive field of the input data. This weighted sum is then placed as an element in the subsequent layer. Nonlinear layers primarily function to alter or constrain the output which is produced. Various nonlinear functions are available for use in CNNs, but ReLU remains one of the most widely employed options[37]. The pooling layer effectively shrinks the dimensionality of the input data. The most commonly used method, max pooling, selects the highest value within each pooling filter. Max pooling is highly regarded in the relevant literature for its efficacy, as it downsamples the input by approximately 75%, delivering significant outcomes. Fully connected layers execute the classification task.

Convolution operation, expressed by Eq. (1), manages processing of the inputs:

$$z_{i,j,k}^{[l]} = w_k^{[l]} x_{i,j}^{[l]} + b_k^{[l]}, \tag{1}$$

here, $z_{i,j,k}^{[l]}$ corresponds to the output feature outcomes produced by k-th feature map on position i, j within l-th layer. The input located on i, j is marked as x, w denotes the filter set, while b describes the bias scores.

Following the convolution operation, activation is executed according to the Eq. (2):

$$g_{i,j,k}^{[l]} = g(z_{i,j,k}^{[l]}) \tag{2}$$

here, $g(\cdot)$ describes non-linear operation administered to the outputs.

The outputs resolution is reduced by the pooling layers, that apply either average or max pooling in the majority of the practical applications. This procedure is expressed by Eq. (3).

$$y_{i,j,k}^{[l]} = pooling(g_{i,j,k}^{[l]}). \tag{3}$$

here, y represents the pooling layer's result.

Ultimately, dense layers perform the classifying task. For multi-labeled data, softmax layer executes classifying task, while for binary classification problems, the logistic (sigmoidal) layer is employed. As the epochs pass by, the network updates the weights and bias scores reducing the cross-entropy loss function in a gradient-descent manner[38]. This is mathematically expressed by Eq. (4).

$$H(p, q) = -\sum_{x} p(x) ln(q(x)) \tag{4}$$

where p and q each denote distribution defined over discrete parameter x.

Optimizing CNN's hyperparameters is essential, as they greatly influence the network's accuracy. Key hyperparameters encompass the count and size of kernels within every convolutional layer, learning rate, batch size, the count of convolutional and fully connected (dense) coats, weight regularization within dense coat, activation functions, dropout rate, and others. Since there is no universal solution for hyperparameter tuning procedure, a "trial and error" approach is often necessary.

CNNs are widely adopted in computer vision[35], with recent advancements across areas such as facial recognition[39], document analysis[40], medical images classifying task and diagnostic support in general[41]. Additionally, CNNs also play an essential role in climate change analysis and extreme weather prediction[42], among numerous other applications[43,44].

CatBoost classification model

Handling categorical datasets poses a considerable challenge within machine learning. Often, substantial preprocessing or conversion is required prior to effectively use data in models. Categorical features are characterized by a set of distinct values known as categories that cannot be compared. One common approach for working with categorical features in boosted tree models is one-hot encoding[45], where each category is represented by a novel binary feature. However, for features with large cardinality, this approach can synthesize an impractically large count of new features. A solution to this issue is to group categories into a limited count of clusters prior to applying one-hot encoding. One popular method for this is employing target statistics (TS)[45], where each category is represented by its projected target value. Yandex scientists devised the CatBoost algorithm[46] specifically to enhance the handling of categorical data compared to traditional approaches.

CatBoost adopts a more advantageous outlook inspirited by online learning frameworks, which process training samples sequentially over time, relying on a concept of ordering. In this method, TS for each instance are computed based solely on prior observations. To adapt this concept for traditional offline environments, CatBoost introduces a pseudo-time by creating a random permutation of the training samples. This allows the TS for each instance to be calculated with respect to all available historical data up to that point. Additionally, CatBoost employs a technique called ordered boosting, which prevents prediction shift, further leveraging the model's reliability[46]. Catboost produces $s + 1$ discrete random permutations of the training dataset at the beginning. Here, σ_0 is utilized to select the leaf scores b_j of the generated trees $h(x) = \sum_{j=1}^{J} b_j \mathbb{I}_{\{x \in R_j\}}$, and the permutations $\sigma_1, ..., \sigma_s$ are used to establish tree structure (like internal nodes). Let the model training is performed employing I trees. There have to exist F^{I-1} of them exercised without the sample x_k if there exist unshifted residual $r^{I-1}(x_k, y_k)$. Instances cannot be used in training F^{I-1} since unbiased residuals are necessary for all training samples. Nevertheless, it is possible to maintain a set of models that differ with respect to the samples included in their training process. To compute the residual for a particular example, a model trained without that example is utilized. This set of models can be constructed through application of the same ordering principle utilized for TS. The algorithm for this approach is showcased as follows:

input : $\{(x_k, y_k)\}_{k=1}^{n}$, I;
$\sigma \leftarrow$ arbitrary permutation of $[1, n]$;
$M_i \leftarrow 0$ for $i = 1..n$;
for $t \leftarrow 1$ **to** I **do**
  for $i \leftarrow 1$ **to** n **do**
    $r_i \leftarrow y_i - M_{\sigma(i)-1}(x_i)$;
  for $i \leftarrow 1$ **to** n **do**
    $\Delta M \leftarrow LearnModel((x_j, r_j) : \sigma(j) \leq i)$;
    $M_i \leftarrow M_i + \Delta M$;
return M_n

Algorithm 1. CatBoost ordered boosting procedure.

Within CatBoost, base estimators behave like oblivious decision trees, meaning that the same splitting criteria are applied across all tree levels. This structure considerably enhances execution speed for testing, creates balanced trees, reduces susceptibility to overfitting issue, and enables significant performance acceleration. The scores within the leaves in the ultimate model are established through the standard gradient boosting procedure, applied consistently across both modes, incorporating all constructed trees. Each training sample is mapped to specific leaves, such as $leaf_0(i)$, with the permutation σ_0 utilized to calculate TS within this context. In testing phase, when the final model is applied to a novel example, TS values are calculated utilizing the entire training dataset.

As the count of categorical features within a dataset grows, the possible combinations increase exponentially, making it impractical to process them all. To address this, CatBoost uses a greedy approach to produce feature combinations. For each split in a tree, CatBoost combines all categorical features and their combinations previously employed in earlier splits of the current tree with all categorical features in the dataset.

LightGBM classification model

LightGBM (light gradient boosting machine)[47] was introduced by Microsoft and made open-source. It is a gradient boosting framework designated for high performance and efficiency when dealing with immense datasets. It manages to achieve excellent performance thanks to a novel method labeled gradient-based one-side sampling (GOSS), that decreases the count of data samples while keeping the accuracy. Moreover, LightGBM also employs exclusive feature bundling (EFB) for combining the mutually exclusive attributes, effectively decreasing the data dimensionality and leveraging the computing efficacy. This pair of innovative procedures helps LightGBM to perform training faster in comparison to conventional boosting models, and efficiently handle immense datasets comprising of thousands and millions of samples and features.

LightGBM exhibited excellent performance in classification, regression and ranking challenges, and consequently has become popular choice for various ML-based applications that span from medicine[48] and climate factors[49], all the way to civil engineering[50] and fault detection[51]. Moreover, innovative design provides support for parallel and distributed processing, allowing it to be scaled with great efficiency over several computing machines. The most commonly optimized LightGBM hyperparameters encompass the count of leaves in a tree (principal parameter to control the tree complexity), maximum depth and learning rate, among others. The model's level of performance is significantly affected by proper choice of these values.

Stochastic optimizers

Metaheuristics optimization encompasses a set of algorithms aimed at discovering approximate resolutions for complex optimization challenges (NP-hard), which are impractical to solve exactly with administration of deterministic conventional mechanisms. Many of these methods take inspiration from natural events, such as evolution or collective behavioral patterns[52]. These are especially valuable to resolve large-scale, nonlinear, or unstructured problems where deterministic techniques fall short because of excessive resource requirements and/or infeasible time-frames[53]. Metaheuristics provide versatility and scale well, allowing them to explore a wide search domain while keeping the risk of becoming trapped in local optima at minimum. Despite they are not able to guarantee the establishment of the global optimal solution, they can discover near-optimal results in acceptable time. Swarm intelligence algorithms represent a subset of these optimization techniques, drawing inspiration from nature, where plain individuals can express complex and smart collective behavior. Due to their distributed nature, algorithms belonging to this group are particularly effective for tackling large, high-dimensional optimization problems[54,55].

Notable exemplars of metaheuristics approaches encompass conventional and broadly-respected algorithms such as particle swarm optimization (PSO)[56], genetic algorithm (GA)[57], variable neighborhood search (VNS)[58], artificial bee colony (ABC)[59], firefly algorithm (FA)[60] and bat algorithm (BA)[61]. A considerable portion of more recent techniques were introduced in the last few years, such as COLSHADE[62], crayfish optimization algorithm (COA)[22], reptile search algorithm (RSA)[24], red fox optimizer (RFO)[23] and recently developed sinh cosh optimizer (SCHO)[63]. Methods belonging to this particular family of algorithms are well known as powerful optimizers, and as such were applied in practice in a broad range of application domains, like time series

forecasting[64], software development[17,65], healthcare[31], cloud and edge computing systems[66,67] and power grids tuning[68]. Moreover, the application of metaheuristics algorithms in the domain of AI models hyperparameters optimization can remarkably enhance their performance[69], as evidenced by numerous preceding studies[70–72]. IoT networks were also leveraged with the application of metaheuristics optimization algorithms[73], addressing challenges like data aggregation[74], blockchain performance optimization[75] and security[76,77].

Methods

This unit commences by briefly introducing the concepts of the baseline chimp optimization algorithm, followed by its known constraints. After the limitations are discussed, this chapter offers a modified variant of the algorithm that improves the performance of the elementary version.

Baseline Chimp optimization algorithm

The chimp optimization algorithm (ChOA) belongs to the group of swarm intelligence metaheuristics techniques, and it was developed to emulate the hunt technique and collective behavioral patterns of a troop of chimpanzees[21]. In this approach, chimpanzees are divided into four key subgroups: attackers, chasers, holders, and callers, each contributing uniquely to enhance the optimization procedure. This collaborative approach aids the algorithm to maintain a balance betwixt exploration (search for novel areas) and exploitation (improving existing solutions).

In the baseline ChOA, the attacking group moves in line with their position update pattern, governed by Eq. (5):

$$X_{\text{attack}} = X_{\text{best}} - A \cdot |C \cdot X_{\text{best}} - X| \tag{5}$$

here, X_{best} denotes the top-performing chimp location, while A and C correspond to the coefficient vectors dynamically adjusted within each round, empowering the exploration procedure.

Individuals belonging to the chasing pack X_{chase} update their positions as governed by Eq. (6):

$$X_{\text{chase}} = X_{\text{attack}} - B \cdot |D \cdot X_{\text{attack}} - X| \tag{6}$$

where B and D serve as control variables for maintaining the balance among exploration and exploitation phases.

The individuals belonging to the holder troop X_{hold} refresh their positions in line with Eq. (7):

$$X_{\text{hold}} = X_{\text{chase}} - E \cdot |F \cdot X_{\text{chase}} - X| \tag{7}$$

here, E and F serve as supplementary parameters that govern this update.

Finally, the individuals from caller troop X_{call} preform position update according to the Eq. (8):

$$X_{\text{call}} = X_{\text{hold}} - G \cdot |H \cdot X_{\text{hold}} - X| \tag{8}$$

here G and H have similar roles to A and C, ajdusted to the callers' function of the optimization process.

By iteratively refining positions, ChOA leverages the collective intelligence of the different chimpanzee roles to solve complex optimization tasks, proving to be an efficient method for addressing high-dimensional search domains within a broad spectrum of real-world applications.

Altered ChOA

Notwithstanding excellent optimization characteristics of the relatively novel ChOA algorithm, thorough experiments on the CEC benchmark function collection[78] exposed some areas of the algorithm that may be targeted for enhancements. These empirical experiments have showcased that the baseline ChOA could profit from the early bolster of the population diversity. Moreover, baseline algorithm's converging speed and balance betwixt diversification and intensification stages could also be leveraged. With these opportunities for improvements in mind, several alterations are proposed in this study.

First added alteration targets boosting of the population diversity over the initialization stage, by incorporating the quasi-adaptive learning (QRL)[79] procedure to the elementary ChOA. In the modified initialization stage, only a half of the solutions are synthesized by applying the conventional ChOA initialization process. Other half of solutions are synthesized with QRL mechanism to boost diversification in the early phase of the algorithm's run. Novel solutions are synthesized as quasi-reflexive opposite individuals with respect to the Eq. 9.

$$X_j^{qr} = rnd\left(\frac{lb_j + ub_j}{2}, x_j\right) \tag{9}$$

here, $\frac{lb_j+ub_j}{2}$ is the arithmetic mean for each parameter's search limits, while *rnd*() represents an arbitrary selection procedure within the given boundaries.

Another modification that was implemented into the ChOA algorithm is the soft rollback mechanism, introduced by this study. If the algorithm stagnates in $T/3$ iterations (empirically established), where T is the maximum number of iterations, rollback of the entire population to the previous state is performed. Two novel control parameters were introduced to support this alteration, stagnation counter *sc* and stagnation threshold *st*. The initial values of these parameters are $sc = 0$ and $st = \frac{T}{3}$. If there is no improvement in the current iteration, *sc* is incremented. If the value of *sc* reaches *st*, soft rollback is performed. Ultimately, elitism is also applied in the following way. When the rollback is performed, the best individual (having the best fitness value) is kept in the population, while the rest are produced by applying the proposed initialization described above.

Considering all included modifications, the novel algorithm was named iteration stagnation aware ChOA (ISA-ChOA), with the pseudo code given in Algorithm 2. It is also necessary to note that the ISA-ChOA utilizes identical control parameters' values and their update procedures as suggested by the authors of the baseline algorithm[21]. Regarding the complexity of the introduced algorithm, it is common to express it in terms of fitness function evaluations (FFEs), since it is the most expensive calculation during the metaheuristics algorithm's execution. Since soft rollback is executed after every three iterations (if the stagnation is confirmed), the complexity of ISA-ChOA algorithm is in the worst case scenario $O(n) = N + N \times T + (N-1) \times T/3$, where N is the count of solutions, while T is the count of iterations. However, in practice, soft rollback is on average triggered only once per run, which is significantly less than the worst case scenario.

```
Set maximum number of agents N
Set stagnation criteria st
Produce N/2 of the population P
Produce remaining agents by applying QRL
Separate agents in to simulated chimp colonies
while t < T do
  Evaluate agent fitness
  Use colony appropriate strategy to update c, f and m
  for Each search agent a in P do
    for each search agent s do
      Determine appropriate search strategy
      Update agent position
    end for
  end for
  Check for stagnation
  if Stagnation confirmed then
    Apply soft rollback
  else
    Store solutions for soft rollback
  end if
end while
```

Algorithm 2. ISA-ChOA optimizer pseudo code.

Experimental setup

The experiments in this study were conducted with a recent CICIoT2023 intrusion detection dataset[34], publicly accessible at https://www.unb.ca/cic/datasets/iotdataset-2023.html. This dataset was developed to evaluate security analytics programs for practical IoT environments and includes 33 distinct attack variants executed across an IoT topology of 105 devices. These attacks are categorized into seven types: DDoS, DoS, Recon, Web-based, Brute Force, Spoofing, and Mirai. This allows for both binary classification (attack versus benign traffic) and multiclass classification with either 8 classes (normal and each attack type) or 34 classes (normal and each of the 33 individual attacks). Class dispersal of both binary and multiclass problems is showcased within Fig. 1. This research addressed 8-class multiclass prediction task. The original dataset contains 1048575 samples. Due to the immense size of the dataset and overwhelming computing requirements, it has been reduced to 20% of the initial size, by performing stratifying of the target 8 classes to keep the imbalance among classes identical to the original dataset. The resulting dataset contains 209715 samples, and is subsequently separated into 70% training and 30% testing data. Models were validated by applying conventional ML metrics: accuracy, precision, sensitivity, and F1-score.

Matthews correlation coefficient (MCC)[80] was opted as the objective function that requires maximization. MCC represents an important indicator particularly when facing imbalanced datasets like CICIoT2023. The

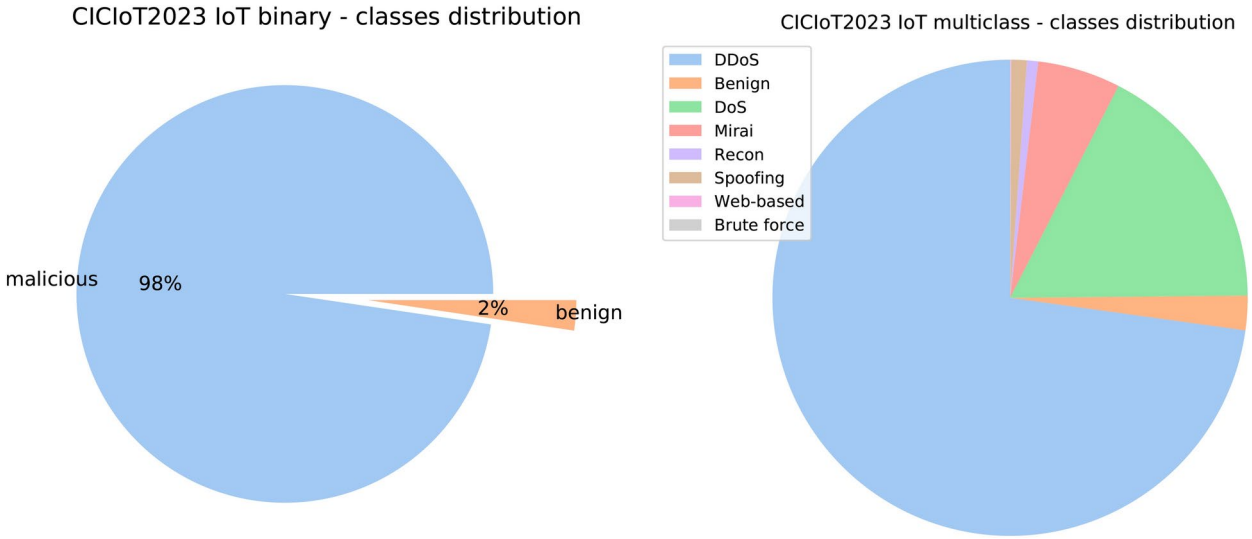

Fig. 1. CICIoT2023 class dispersal for binary and multiclass classification.

Hyperparameter	Lower bound	Upper bound
Learning rate	0.0001	0.003
Dropout	0.001	0.5
Epochs	10	30
CNN layers	1	2
Dense layers	1	2
Neurons per layer	32	128

Table 1. Collection of CNN hyperparameters tuned in this study with their search restrictions.

Parameter	Lower constraint	Upper constraint
Learning rate	0.001	0.100
Col sample by level (cbl)	0.05	1
Subsample	0.05	1
Iterations	50	100
Depth	1	5
Minimum data in leaf (mdl)	1	5

Table 2. CatBoost set of optimized hyperparameters with search boundaries.

imbalance of the utilized dataset is indeed a challenge, however, it reflects a real-world situation, since most of the real-life network traffic is not balanced. Thus, it is crucial that the proposed model should work properly with highly imbalanced data. MCC value is established by utilizing the Equation (10). Moreover, the classification error (which is defined by $1 - accuracy$) was monitored across all simulations and acted as the indicator function.

$$\mathrm{MCC} = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{10}$$

here, TP corresponds to the true positive forecasts, TN represents the count of true negative predictions, FP is the amount of false positive predictions, and finally, FN denotes the count of false negative classifications.

CNN set of hyperparameters was tuned within the premier tier of the introduced ML framework. The collection of opted hyperparameters, along with search region limits for each parameter is presented in Table 1. Batch size of 512 was used, with early stopping enabled.

Parameter	Lower constraint	Upper constraint
Learning rate	0.001	0.100
Number of leaves	24	80
Feature fraction	0.1	0.9
Bagging fraction	0.8	1
Max depth	5	30
Max bin	20	90
Minimum data in leaf	20	80
Minimum sum hessian in leaf	0	100
Subsample	0.01	1

Table 3. LightGBM set of optimized hyperparameters with search boundaries.

CNN's intermediate outputs were wired to the framework's second tier. This collection of outputs were stored within CNN's classifying activity for every sample in the dataset, and subsequently separated into another 70% training and 30% testing split, which was fed to CatBoost and LightGBM structures throughout their respective tuning processes. The hyperparameters' collection of CatBoost opted for optimization procedure in this study is showcased in Table 2. Likewise, LightGBM hyperparameters that were tuned are presented in Table 3. These opted parameters are known to have the most influence on the model's behavior.

The suggested ISA-ChOA metaheuristics was utilized for optimization, and comparative analysis with multiple cutting-edge optimizers was performed. The set of contending algorithms comprised of elementary ChOA[21], VNS[58], PSO[56], BA[61], ABC[59], WOA[81] and RSA[24]. The contending metaheuristics were separately implemented for the sake of this research, with default configurations of control parameters that were recommended by their respective creators. In case of CNN simulations, every metaheuristics used 8 individuals in the populace, with 5 iterations in each run and 5 separate executions, to account the randomness linked to the stochastic algorithms. Similarly, for CatBoost and LightGBM tuning, metaheuristics used 10 individuals per populace, 10 iterations per run and a total of 30 independent executions. For CNN tuning process the authors opted for smaller population size and number of rounds, since CNN optimization requires considerably more computing resources. A simulation framework flowchart is provided in Fig. 2

Results

This unit showcases the experimental findings from the conducted simulations. In multiclass simulations, the superior outcomes for each category for all tables showing the simulation findings are emphasized in bold font.

Layer 1 CNN multiclass experiments

The simulation findings of the premier tier of the framework, where CNN was optimized with respect to the fitness function (MCC) for multiclass classifying venture, are delivered within Table 4. The proposed ISA-ChOA algorithm dispatched highest ranking results, by attaining the MCC of 0.691852 for the best run and 0.639513 in the worst execution, with mean and median scores of 0.639513 and 0.671254, respectively. On the opposite, the superior stability indicated by the lowest deviation and variance scores was attained by WOA metaheuristics. Despite respectful stability, however, WOA was considerably behind more advanced algorithms regarding other metrics.

Indicator function (set as classification error outlay) results are outlined within Table 5. Once more, the supremacy of the introduced ISA-ChOA metaheuristics may be observed, reflected in the best outcome of 0.137344. ISA-ChOA also outclassed other contending algorithms for the worst, mean and median scores, while again the superior stability of the outcomes was exhibited by WOA metaheuristics.

Violin and swarm plots of the fitness function (MCC) for the multiclass classifying problem are presented within Fig. 3. ISA-ChOA was not able to establish the highest stability of the results, nevertheless, other contenders which obtained better stability of MCC across independent runs did not match the overall superior performance of ISA-ChOA. This is also visible from the swarm plot graph, showing the diversity of the population within the final round of the best execution. Supplementary visualizations of the outcomes are outlined within Fig. 4 through box and swarm plots of the indicator function.

Converging diagrams of both MCC and error rate, for every considered algorithm are outlined within Fig. 5, where it is clear that the proposed ISA-ChOA demonstrated superior convergence, and outclassed all contenders by establishing the best outcome of the fitness function. The same applies for the converging of the error rate (indicator), although it was not specified as the goal for tuning.

Table 6 sets forth a comprehensive evaluation of the top-performing CNN architectures for multiclass classifying challenge, tuned with all optimizers encompassed in comparative evaluation. Even optimized CNNs are frequently struggling to properly detect mirai and recon attacking patterns, which is evident from the provided results. Additionally, PSO-based CNN fails entirely to converge, with abysmal final accuracy. A couple of measurements should be considered for determining the optimal method, including precision, recall and f1-score per each class. Nevertheless, the greatest accuracy among all observed methods was achieved by the suggested ISA-ChOA CNN model, that outclassed other approaches with the final overall accuracy of 0.862656.

The best sets of CNN's hyperparameter values determined with each regarded optimizer are put forth within Table 7, to provide support for subsequent replication studies. These values may help other scientists that seek

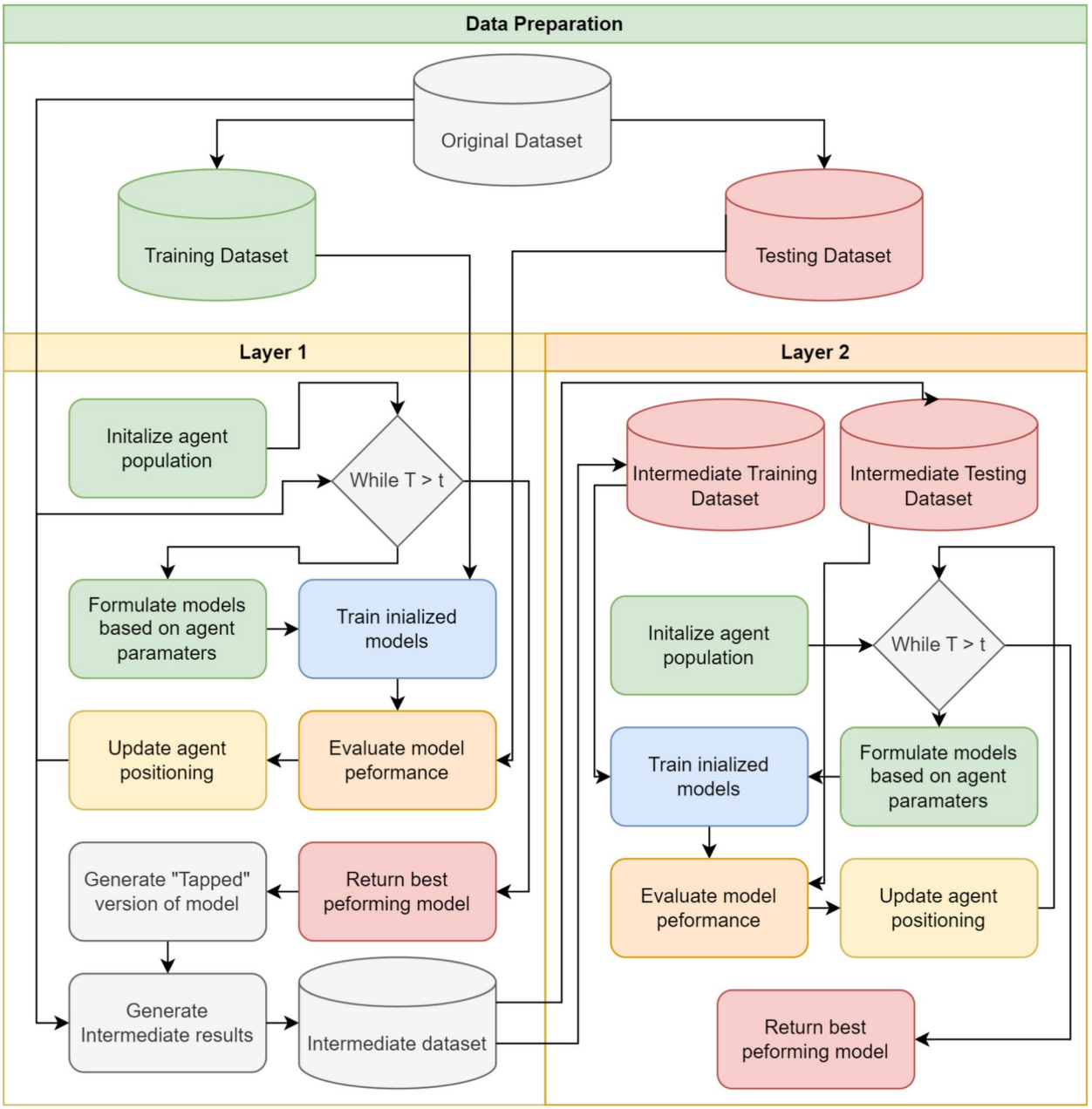

Fig. 2. Proposed simulation framework flowchart.

Algorithm	Best	Worst	Mean	Median	Std	Var
CNN-ISA-ChOA	**0.691852**	**0.639513**	**0.671254**	**0.691852**	0.028211	7.95E-04
CNN-ChOA	0.677421	0.634565	0.646993	0.641397	0.015474	2.39E-04
CNN-VNS	0.639245	0.633145	0.636067	0.636056	0.002561	6.56E-06
CNN-PSO	0.639819	0.634424	0.635993	0.634763	0.002054	4.22E-06
CNN-BA	0.635105	0.621941	0.629489	0.631148	0.004595	2.11E-05
CNN-ABC	0.635973	0.627362	0.633416	0.635611	0.003277	1.07E-05
CNN-WOA	0.636434	0.632404	0.633983	0.633352	**0.001490**	**2.22E-06**
CNN-RSA	0.638300	0.631014	0.634379	0.635692	0.002857	8.16E-06

Table 4. Layer 1 CNN multiclass objective function scores over 30 simulations. Best obtained metrics are shown in bold style.

Algorithm	Best	Worst	Mean	Median	Std	Var
CNN-ISA-ChOA	**0.137344**	**0.147135**	**0.141140**	**0.137344**	0.005202	2.71E-05
CNN-ChOA	0.145736	0.149249	0.146973	0.146467	0.001286	1.65E-06
CNN-VNS	0.147675	0.149519	0.148610	0.148518	0.000935	8.74E-07
CNN-PSO	0.147119	0.149058	0.148527	0.149058	0.000763	5.82E-07
CNN-BA	0.148931	0.153668	0.151090	0.150362	0.001775	3.15E-06
CNN-ABC	0.148709	0.151776	0.149503	0.148709	0.001226	1.50E-06
CNN-WOA	0.148311	0.149901	0.149287	0.149424	**0.000597**	**3.57E-07**
CNN-RSA	0.147564	0.150314	0.149084	0.148661	0.001060	1.12E-06

Table 5. Layer 1 CNN multiclass indicator function scores over 30 simulations. Best obtained metrics are shown in bold style.

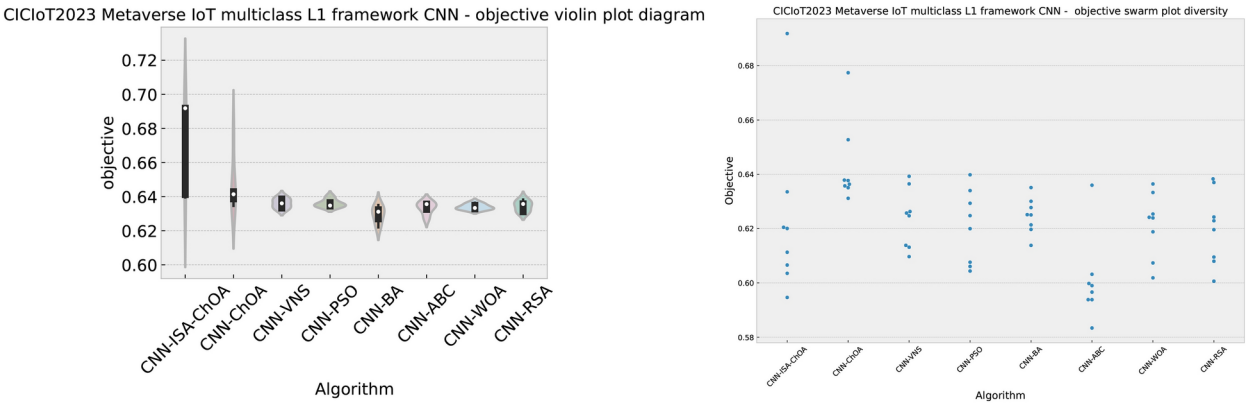

Fig. 3. Layer 1 CNN multiclass objective function distribution and swarm diagrams.

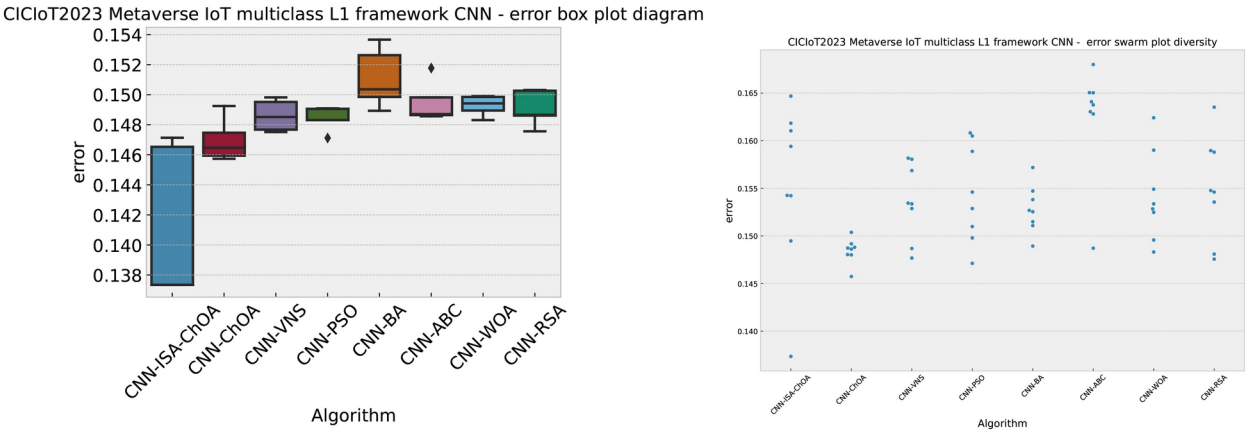

Fig. 4. Layer 1 CNN multiclass indicator function distribution and swarm diagrams.

to recreate the experimental aftermaths on their own, as these CNN architectures achieved the outcomes shown and discussed within Table 6. Ultimately, additional visualizations in shape of PR curves and confusion matrix for the most suitable model (CNN-ISA-ChOA in this scenario) are outlined within Fig. 6.

The best performing CNN model architectures, and the "tapped" intermediate version are provided visually in Fig. 7, where it can be noted that 32 features were extracted by the CNN.

Layer 2 CatBoost multiclass experiments

The findings of the conducted simulations of the framework's second tier, in terms of CatBoost tuning process with the MCC set as the fitness function for multiclass classifying venture, are showcased within Table 8. The suggested ISA-ChOA algorithm dispatched highest ranking results, by attaining the best MCC of 0.806747 for the best run with mean score of 0.805208. Moreover, ISA-ChOA shared the best results of worst and median

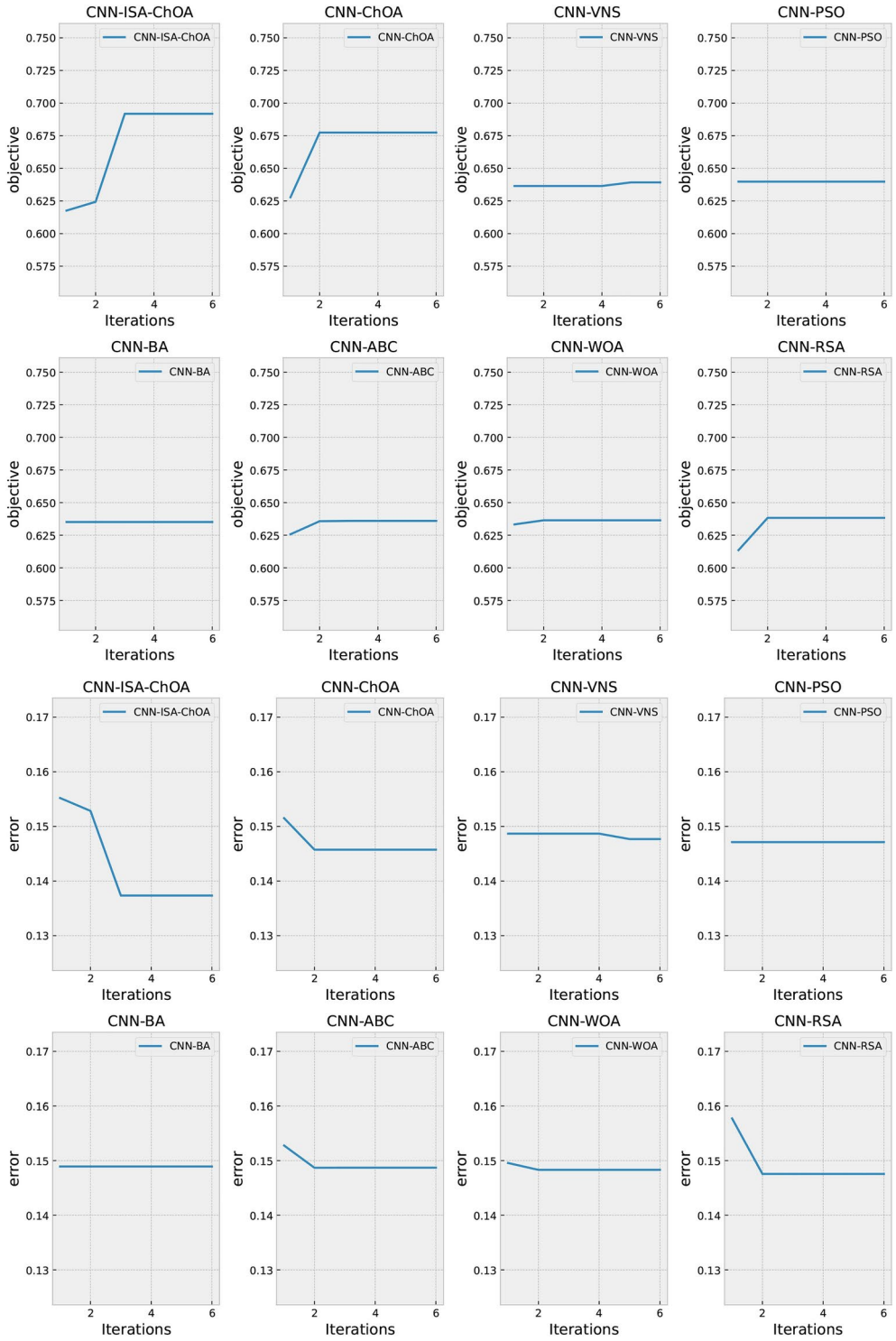

Fig. 5. Layer 1 CNN multiclass objective and indicator function convergence diagrams.

metrics with a couple of other oprimizers. On the opposite, the superior stability indicated by the lowest deviation and variance scores was attained by BA and RSA metaheuristics. Despite respectful stability, however, these algorithms were behind other optimizers regarding the remaining metrics.

Indicator function (set as classification error outlay) findings are outlined within Table 9. Once more, the supremacy of the introduced ISA-ChOA metaheuristics may be observed, reflected in the best outcome for classification error of 0.082222. ISA-ChOA also outclassed other contending algorithms for the mean score, and shared the best outcomes of worst and median metrics with several other optimizers, while the supreme stability of the outcomes was exhibited by baseline ChOA, BA and RSA metaheuristics.

Approach	Metric	Benign	Brute force	DDoS	DoS	Mirai	Recon	Spoofing	Web	Accuracy	Macro avg	Weighted avg
CNN-ISA-ChOA	Precision	0.721480	0.922479	0.623072	0.792727	0.000000	1.000000	0.840000	0.999433	0.862656	0.737399	0.868015
	Recall	0.955752	0.901860	0.682524	0.457023	0.000000	0.052632	0.570136	0.991840	0.862656	0.576471	0.862656
	f1-score	0.822255	0.912053	0.651444	0.579787	0.000000	0.100000	0.679245	0.995622	0.862656	0.592551	0.863891
CNN-ChOA	Precision	0.777710	0.922179	0.597816	0.668235	0.000000	1.000000	0.692053	0.992705	0.854264	0.706337	0.861855
	Recall	0.874064	0.890072	0.683626	0.595388	0.000000	0.105263	0.630468	0.995498	0.854264	0.596797	0.854264
	f1-score	0.823077	0.905841	0.637848	0.629712	0.000000	0.190476	0.659826	0.994099	0.854264	0.605110	0.857149
CNN-VNS	Precision	0.773058	0.853032	0.790274	0.619617	0.000000	0.000000	0.717391	0.999717	0.852325	0.594136	0.844698
	Recall	0.867257	0.981511	0.291028	0.542977	0.000000	0.000000	0.647059	0.992684	0.852325	0.540314	0.852325
	f1-score	0.817453	0.912773	0.425398	0.578771	0.000000	0.000000	0.680412	0.996188	0.852325	0.551374	0.825186
CNN-PSO	Precision	0.023349	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.023349	0.002919	0.000545
	Recall	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.023349	0.125000	0.023349
	f1-score	0.045632	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.023349	0.005704	0.001065
CNN-BA	Precision	0.696793	0.863293	0.699412	0.886076	0.000000	0.000000	0.909605	0.995781	0.851069	0.631370	0.838487
	Recall	0.976174	0.962979	0.360272	0.440252	0.000000	0.000000	0.485671	0.996061	0.851069	0.527676	0.851069
	f1-score	0.813156	0.910415	0.475573	0.588235	0.000000	0.000000	0.633235	0.995921	0.851069	0.552067	0.831613
CNN-ABC	Precision	0.745338	0.864658	0.685934	0.762658	0.000000	1.000000	0.844828	0.998024	0.851291	0.737680	0.837092
	Recall	0.952349	0.959727	0.369455	0.505241	0.000000	0.105263	0.591252	0.994654	0.851291	0.559743	0.851291
	f1-score	0.836222	0.909715	0.480244	0.607818	0.000000	0.190476	0.695652	0.996336	0.851291	0.589558	0.833337
CNN-WOA	Precision	0.721278	0.861107	0.715427	0.682065	0.000000	0.000000	0.921788	1.000000	0.851689	0.612708	0.839058
	Recall	0.953029	0.967345	0.345394	0.526205	0.000000	0.000000	0.497738	0.992122	0.851689	0.535229	0.851689
	f1-score	0.821114	0.911139	0.465874	0.594083	0.000000	0.000000	0.646425	0.996045	0.851689	0.554335	0.830837
CNN-RSA	Precision	0.728610	0.860137	0.728834	0.711429	0.000000	0.000000	0.821739	0.999434	0.852436	0.606273	0.839980
	Recall	0.910143	0.970161	0.339150	0.522013	0.000000	0.000000	0.570136	0.992966	0.852436	0.538071	0.852436
	f1-score	0.809322	0.911842	0.462898	0.602177	0.000000	0.000000	0.673197	0.996189	0.852436	0.556953	0.830910
	Support	1469	45812	10889	477	32	19	663	3554			

Table 6. Layer 1 CNN multiclass comprehensive metrics for best tuned models.

Method	Learning Rate	Dropout	Epochs	Layers CNN	Layers Dense	Neurons CNN L1	Neurons CNN L2	Neurons Dense L1	Neurons Dense L2
CNN-ISA-ChOA	2.16e-03	1.07e-03	30	1	1	128	N/A	50	N/A
CNN-ChOA	2.69e-03	8.03e-03	25	2	2	120	92	82	40
CNN-VNS	3.00e-03	2.47e-01	30	1	2	111	N/A	128	106
CNN-PSO	2.03e-03	3.71e-02	27	2	2	59	125	40	91
CNN-BA	3.00e-03	5.00e-01	30	2	2	92	32	128	122
CNN-ABC	2.55e-03	3.03e-01	29	1	1	128	N/A	70	N/A
CNN-WOA	3.00e-03	1.79e-01	27	1	1	128	N/A	79	N/A
CNN-RSA	3.00e-03	7.33e-02	30	1	1	128	N/A	77	N/A

Table 7. Layer 1 CNN multiclass optimized CNN model parameter selections.

Violin and swarm plots of the fitness function (MCC) for the multiclass classifying problem are presented within Fig. 8. ISA-ChOA was not able to establish the highest stability of the results, nevertheless, other contenders which obtained better stability of MCC across independent runs did not match the overall superior performance of ISA-ChOA. This is also visible from the swarm plot graph, showing the diversity of the population within the final round of the best execution. Supplementary visualizations of the outcomes are outlined within Fig. 9 through box and swarm plots of the classification error rate.

Converging diagrams of both MCC and error rate, for every considered algorithm are outlined within Figs. 10 and 11, where it is clear that the proposed ISA-ChOA demonstrated superior convergence, and outclassed all contenders by establishing the best outcome of the fitness function. The same applies for the converging of the error rate (indicator), although it was not targeted as the goal for tuning.

Table 10 sets forth a comprehensive analysis of the top-performing CatBoost architectures for multiclass classifying challenge, tuned with all optimizers encompassed in comparative evaluation. Even optimized CatBoost models are frequently struggling to properly detect mirai and recon attacking patterns, which is evident from the provided results. A couple of measurements should be considered for determining the optimal method, including precision, recall and f1-score per each class. Nevertheless, the greatest accuracy among all

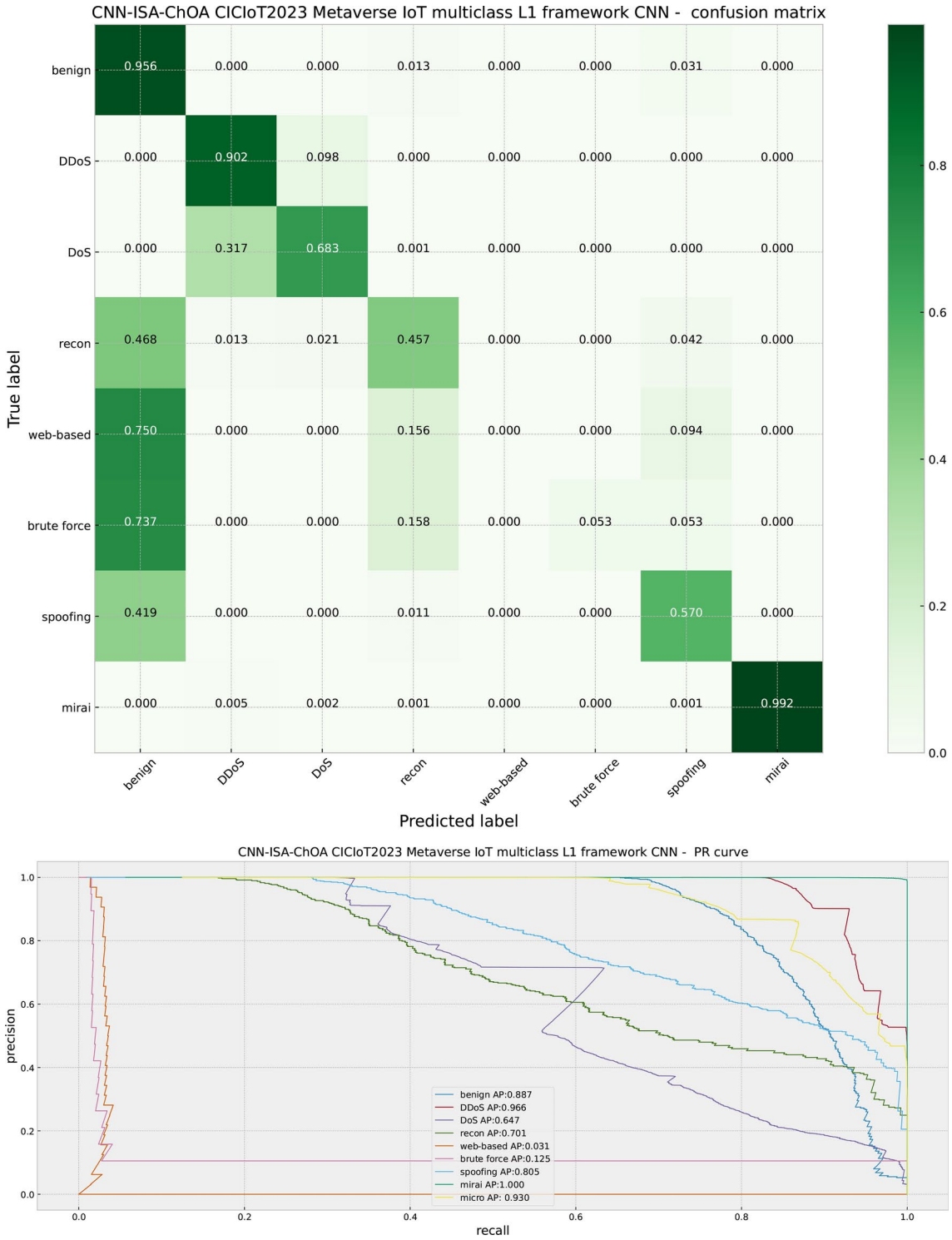

Fig. 6. Layer 1 CNN multiclass Best performing CNN-ISA-ChOA optimized model confusion matrix and PR diagram.

observed methods was achieved by the suggested ISA-ChOA CatBoost model, that outclassed other approaches with the final overall accuracy of 0.917778.

The best sets of CatBoost's hyperparameter values determined with each regarded optimizer are put forth within Table 11, to provide support for possible subsequent replication experiments. These values may help other scientists that seek to recreate the experimental aftermaths on their own, as these CatBoost architectures achieved the outcomes shown and discussed within Table 10. Ultimately, additional visualization in shape of confusion matrix for the most suitable model (CNN-CB-ISA-ChOA in this scenario) is outlined within Fig. 12.

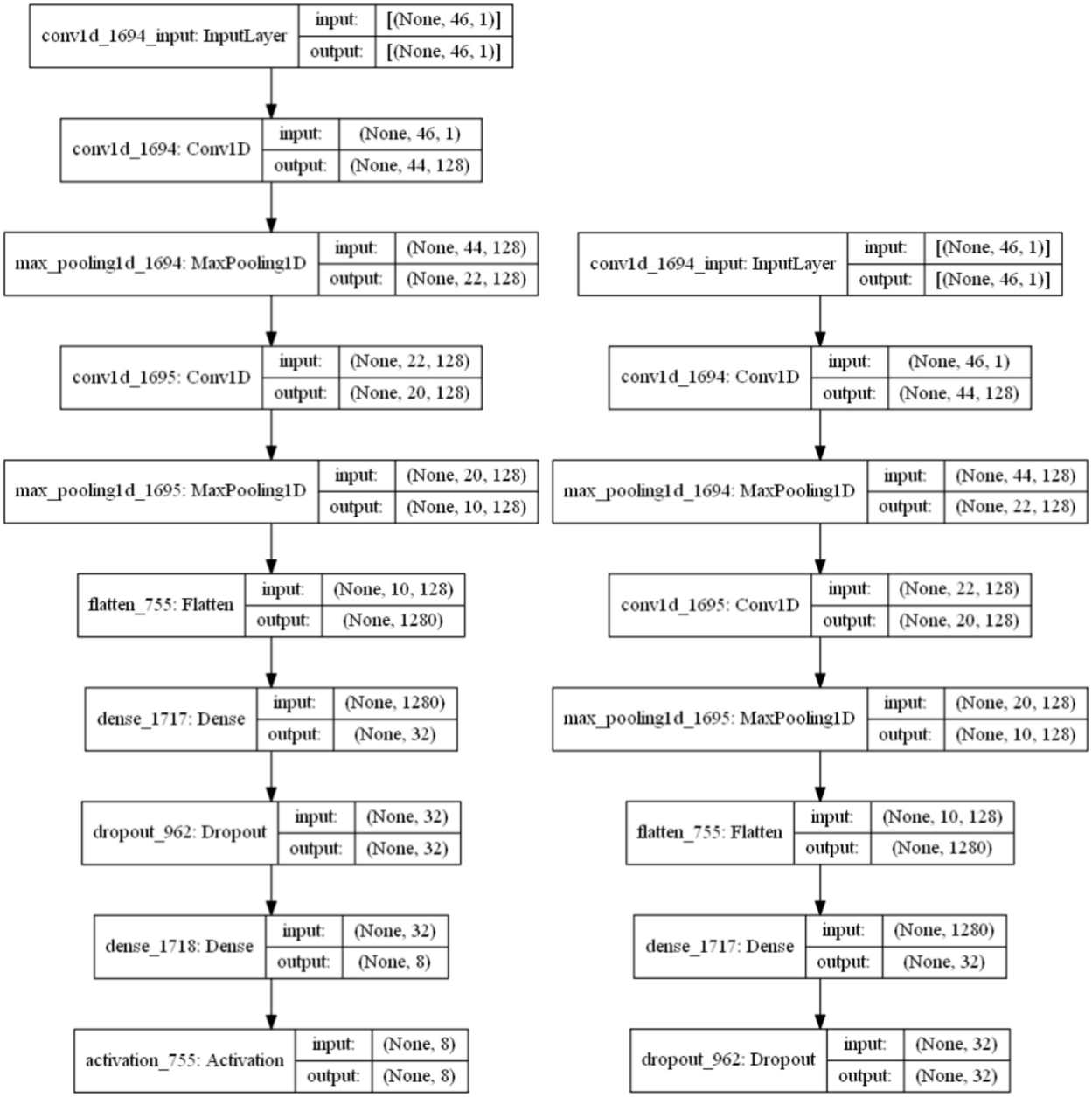

Fig. 7. The best performing CNN model, and the "tapped" version of the network where the output layers are intercepted.

Algorithm	Best	Worst	Mean	Median	Std	Var
CNN-CB-ISA-ChOA	**0.806747**	0.804912	**0.805208**	0.804912	0.000607	3.68E-07
CNN-CB-ChOA	0.804912	0.804912	0.804912	0.804912	0.000000	0.00E-00
CNN-CB-VNS	0.804912	0.804076	0.804808	0.804912	0.000277	7.65E-08
CNN-CB-PSO	0.805425	0.804912	0.804981	0.804912	0.000168	2.83E-08
CNN-CB-BA	0.804912	0.804912	0.804912	0.804912	0.000000	0.00E-00
CNN-CB-ABC	0.805256	0.791713	0.799831	0.802300	0.005020	2.52E-05
CNN-CB-WOA	0.805361	0.804912	0.805064	0.804912	0.000198	3.91E-08
CNN-CB-RSA	0.804912	0.804912	0.804912	0.804912	0.000000	0.00E-00

Table 8. Layer 2 CatBoost multiclass objective function scores over 30 simulations. Best obtained metrics are shown in bold style.

Algorithm	Best	Worst	Mean	Median	Std	Var
CNN-CB-ISA-ChOA	**0.082222**	0.082953	**0.082840**	0.082953	2.40E-04	5.78E-08
CNN-CB-ChOA	0.082953	0.082953	0.082953	0.082953	0.00E-00	0.00E-00
CNN-CB-VNS	0.082953	0.083382	0.083007	0.082953	1.42E-04	2.01E-08
CNN-CB-PSO	0.082890	0.082953	0.082943	0.082953	2.09E-05	4.38E-10
CNN-CB-BA	0.082953	0.082953	0.082953	0.082953	0.00E-00	0.00E-00
CNN-CB-ABC	0.082890	0.088469	0.085093	0.084050	2.06E-03	4.26E-06
CNN-CB-WOA	0.082921	0.082953	0.082921	0.082953	5.27E-05	2.78E-09
CNN-CB-RSA	0.082953	0.082953	0.082953	0.082953	0.00E-00	0.00E-00

Table 9. Layer 2 CatBoost multiclass indicator function scores over 30 simulations. Best obtained metrics are shown in bold style.

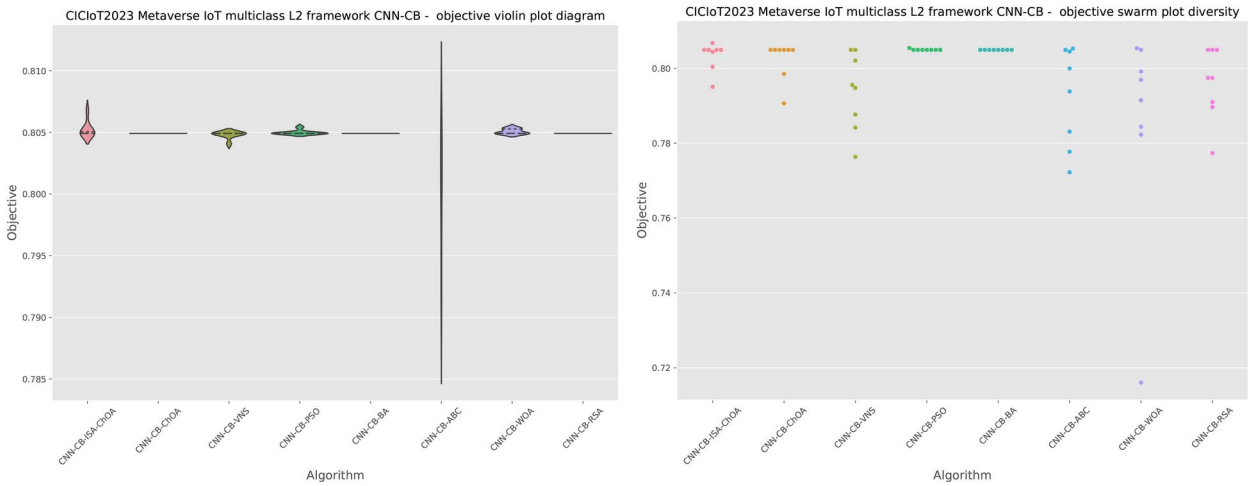

Fig. 8. Layer 2 CatBoost multiclass objective score distribution and swarm diagrams.

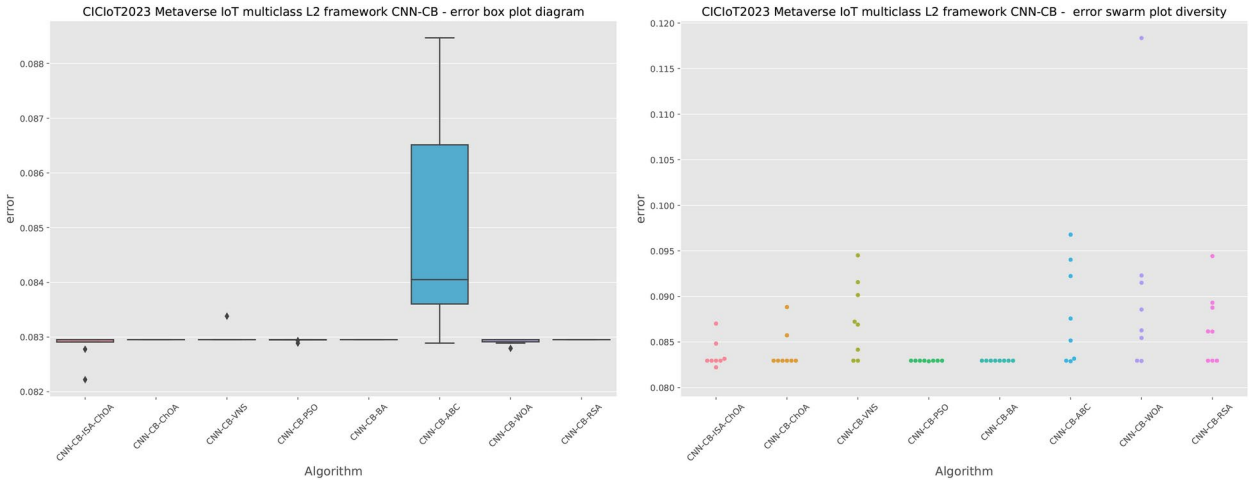

Fig. 9. Layer 2 CatBoost multiclass indicator score distribution and swarm diagrams.

Layer 2 LightGBM multiclass experiments

The findings of the conducted simulations of the framework's second tier, in terms of LightGBM tuning process with the MCC set as the fitness function for multiclass classifying venture, are showcased within Table 12. The suggested ISA-ChOA algorithm dispatched supreme level of outcomes for all observed metrics, by attaining the best MCC of 0.996207 for the best run, 0.985756 for the worst execution, with mean and median outcomes

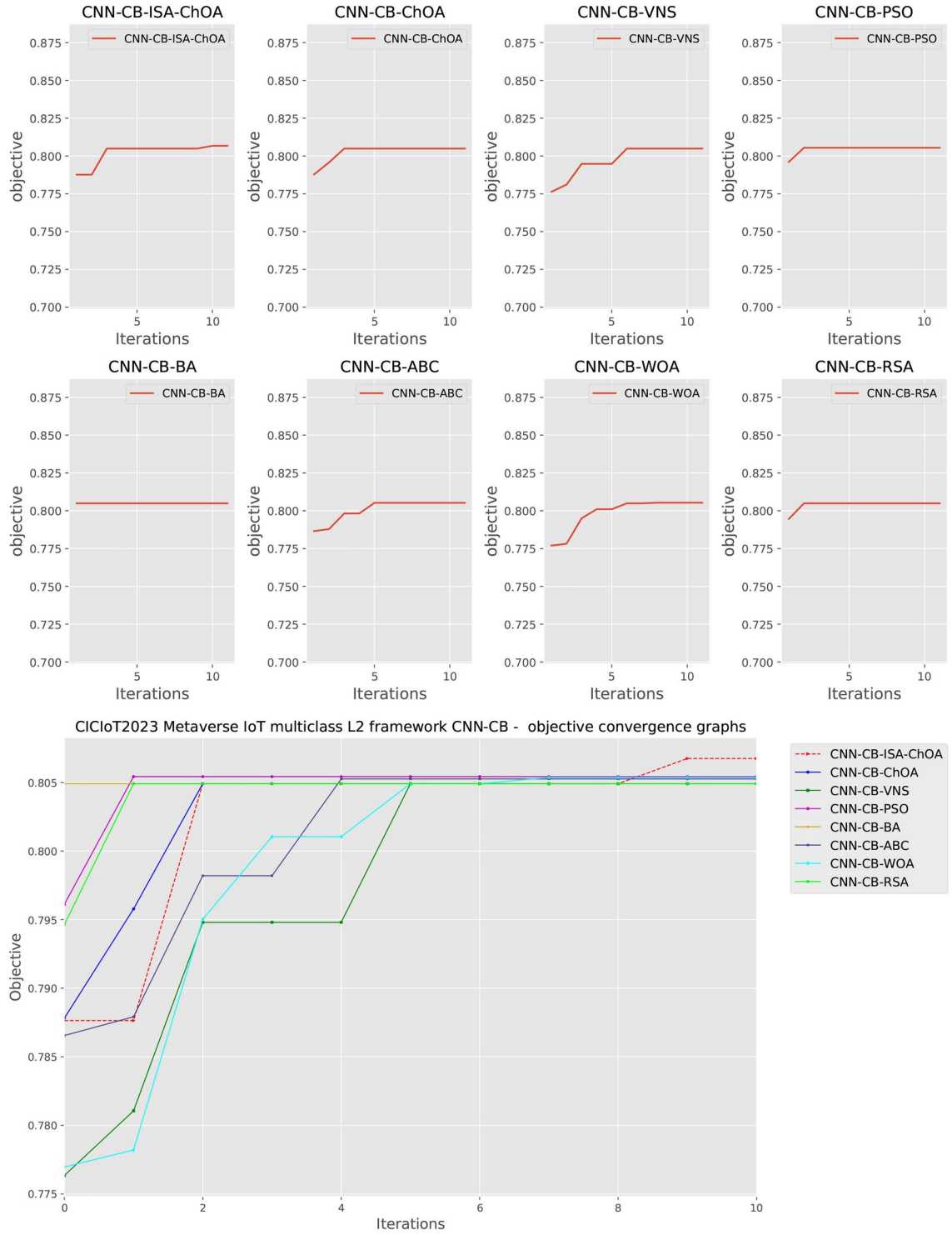

Fig. 10. Layer 2 CatBoost multiclass objective convergence diagrams.

of 0.991700 and 0.993378. Moreover, in this experiment, ISA-ChOA obtained the superior stability as well, indicated by the lowest deviation and variance scores of 0.003788 and 0.000014, respectively.

Indicator function (set as classification error outlay) findings are outlined within Table 13. Once again, the supremacy of the proposed ISA-ChOA metaheuristics may be noted, reflected in the best outcome for classification error of 0.001653. ISA-ChOA also outclassed other contending algorithms for the worst, mean and median metrics. The supreme stability of the outcomes was exhibited by ISA-ChOA metaheuristics as well.

Violin and swarm plots of the fitness function (MCC) for the multiclass classifying problem are presented within Fig. 13. ISA-ChOA was able to establish the highest stability of the results, while other contenders which obtained good stability of MCC across independent runs did not match the overall superior performance of ISA-

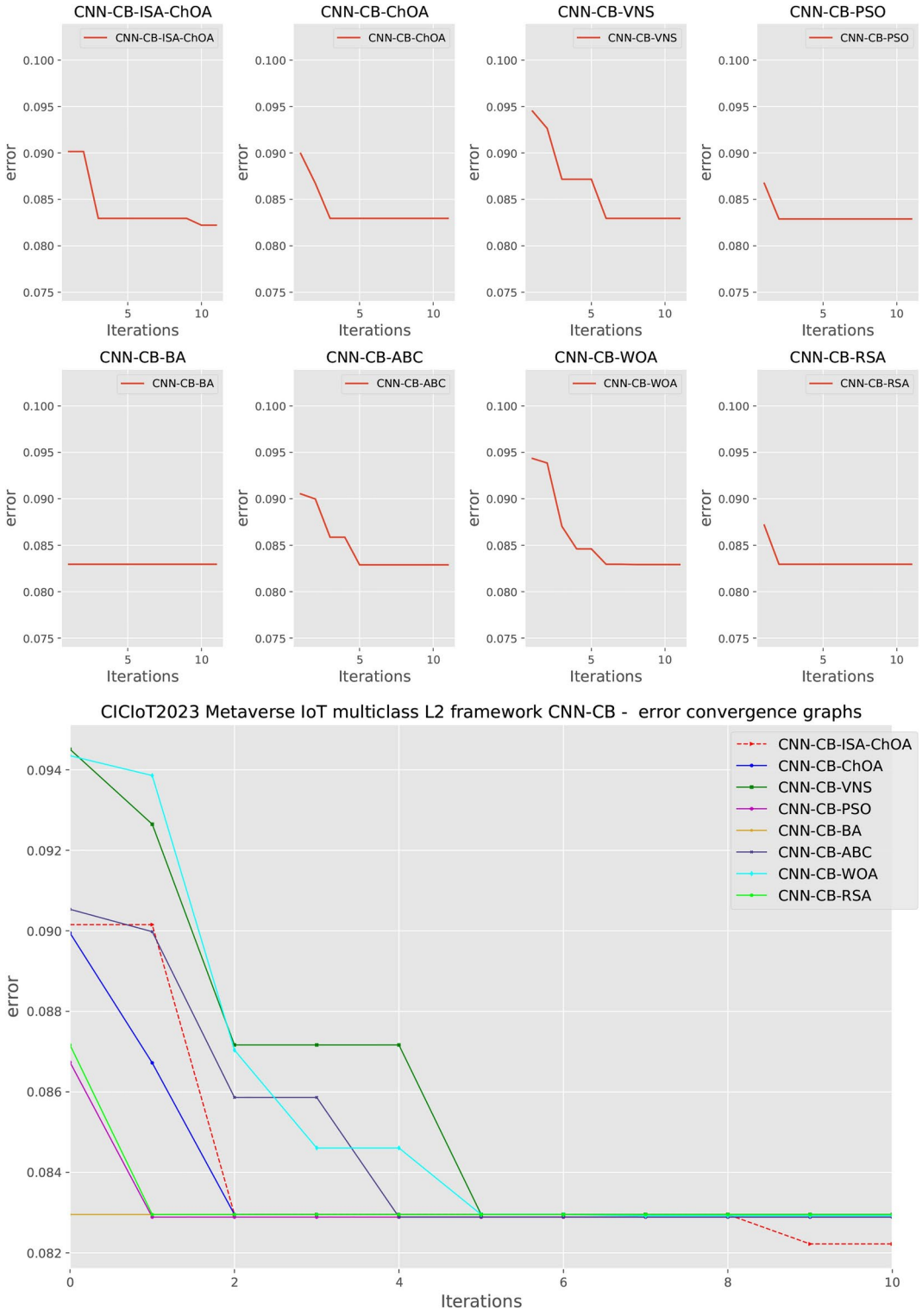

Fig. 11. Layer 2 CatBoost multiclass indicator convergence diagrams.

ChOA. This is also visible from the swarm plot graph, showing the diversity of the population within the final round of the best execution. Supplementary visualizations of the outcomes are outlined within Fig. 14 through box and swarm plots of the classification error rate.

Converging diagrams of both MCC and error rate, for every considered algorithm are outlined within Figs. 15 and 16, where it is clear that the proposed ISA-ChOA demonstrated superior convergence, and outclassed all contenders by establishing the best outcome of the fitness function. The same applies for the converging of the error rate (indicator), although it was not targeted as the goal for tuning.

Approach	Metric	Benign	Brute force	DDoS	DoS	Mirai	Recon	Spoofing	Web	Accuracy	Macro avg	Weighted avg
CNN-CB-ISA-ChOA	Precision	0.717203	0.935205	0.845409	0.785953	0.000000	1.000000	0.890306	0.999433	0.917778	0.771689	0.916141
	Recall	0.956433	0.968480	0.721187	0.492662	0.000000	0.368421	0.526395	0.991840	0.917778	0.628177	0.917778
	f1-score	0.819720	0.951552	0.778372	0.605670	0.000000	0.538462	0.661611	0.995622	0.917778	0.668876	0.914704
CNN-CB-ChOA	Precision	0.718670	0.934240	0.844639	0.788591	0.000000	1.000000	0.885787	0.999433	0.917047	0.771420	0.915312
	Recall	0.956433	0.968480	0.716962	0.492662	0.000000	0.368421	0.526395	0.991840	0.917047	0.627649	0.917047
	f1-score	0.820678	0.951052	0.775581	0.606452	0.000000	0.538462	0.660360	0.995622	0.917047	0.668526	0.913872
CNN-CB-VNS	Precision	0.718670	0.934240	0.844639	0.788591	0.000000	1.000000	0.885787	0.999433	0.917047	0.771420	0.915312
	Recall	0.956433	0.968480	0.716962	0.492662	0.000000	0.368421	0.526395	0.991840	0.917047	0.627649	0.917047
	f1-score	0.820678	0.951052	0.775581	0.606452	0.000000	0.538462	0.660360	0.995622	0.917047	0.668526	0.913872
CNN-CB-PSO	Precision	0.719674	0.936015	0.837123	0.807971	0.000000	1.000000	0.889447	0.999433	0.917110	0.773708	0.915512
	Recall	0.961198	0.966253	0.726880	0.467505	0.000000	0.368421	0.533937	0.991277	0.917110	0.626934	0.917110
	f1-score	0.823084	0.950894	0.778116	0.592297	0.000000	0.538462	0.667295	0.995338	0.917110	0.668186	0.914201
CNN-CB-BA	Precision	0.718670	0.934240	0.844639	0.788591	0.000000	1.000000	0.885787	0.999433	0.917047	0.771420	0.915312
	Recall	0.956433	0.968480	0.716962	0.492662	0.000000	0.368421	0.526395	0.991840	0.917047	0.627649	0.917047
	f1-score	0.820678	0.951052	0.775581	0.606452	0.000000	0.538462	0.660360	0.995622	0.917047	0.668526	0.913872
CNN-CB-ABC	Precision	0.714792	0.935303	0.841486	0.802867	0.000000	1.000000	0.882952	0.998867	0.917110	0.772033	0.915495
	Recall	0.960517	0.967519	0.722013	0.469602	0.000000	0.368421	0.523379	0.991840	0.917110	0.625411	0.917110
	f1-score	0.819634	0.951138	0.777185	0.592593	0.000000	0.538462	0.657197	0.995341	0.917110	0.666444	0.914034
CNN-CB-WOA	Precision	0.719084	0.936053	0.837017	0.807273	0.000000	1.000000	0.889447	0.999149	0.917079	0.773503	0.915486
	Recall	0.961879	0.966232	0.726789	0.465409	0.000000	0.368421	0.533937	0.991277	0.917079	0.626743	0.917079
	f1-score	0.822947	0.950903	0.778018	0.590426	0.000000	0.538462	0.667295	0.995198	0.917079	0.667906	0.914165
CNN-CB-RSA	Precision	0.718670	0.934240	0.844639	0.788591	0.000000	1.000000	0.885787	0.999433	0.917047	0.771420	0.915312
	Recall	0.956433	0.968480	0.716962	0.492662	0.000000	0.368421	0.526395	0.991840	0.917047	0.627649	0.917047
	f1-score	0.820678	0.951052	0.775581	0.606452	0.000000	0.538462	0.660360	0.995622	0.917047	0.668526	0.913872
	Support	1469	45812	10889	477	32	19	663	3554			

Table 10. Layer 2 CatBoost multiclass comprehensive metrics for best tuned models.

Approach	lr	cbl	Subsample	Iterations	Depth	mdl
CNN-CB-ISA-ChOA	9.96E-02	8.42E-01	5.00E-02	100	5	3
CNN-CB-ChOA	1.00E-01	3.09E-01	1.00E-00	100	5	2
CNN-CB-VNS	1.00E-01	5.00E-02	1.00E-00	100	5	2
CNN-CB-PSO	9.89E-02	2.33E-01	6.99E-01	97	5	4
CNN-CB-BA	1.00E-01	7.37E-01	1.00E-00	100	5	2
CNN-CB-ABC	9.91E-02	9.79E-01	5.64E-01	100	5	4
CNN-CB-WOA	9.89E-02	5.33E-01	6.50E-02	100	5	1
CNN-CB-RSA	1.00E-01	1.98E-01	1.00E-00	100	5	3

Table 11. Best CatBoost model parameter selections made by each optimizer.

Table 14 sets forth a comprehensive analysis of the top-performing CatBoost architectures for multiclass classifying challenge, tuned with all optimizers encompassed in comparative evaluation. Even optimized CatBoost models are frequently struggling to properly detect mirai and recon attacking patterns, which is evident from the provided results. A couple of measurements should be considered for determining the optimal method, including precision, recall and f1-score per each class. Nevertheless, the greatest accuracy among all observed methods was achieved by the suggested ISA-ChOA LightGBM model, that outclassed other approaches with the final overall accuracy of 0.998346.

The best sets of LightGBM's hyperparameter values determined with each regarded optimizer are put forth within Table 15, to provide support for possible subsequent replication experiments. These values may help other scientists that seek to recreate the experimental aftermaths on their own, as these CatBoost architectures achieved the outcomes shown and discussed within Table 14. Ultimately, additional visualization in shape of confusion matrix for the most suitable model (CNN-LGBM-ISA-ChOA in this scenario) is outlined within Fig. 17.

Comparison with state of the art classifcaiton models

To demonstrate the improvements attained by utilizing the introduced optimization framework, a comparative analysis between several baseline classifiers is included. Commonly used as well as relatively recent models have all been evaluated including decision trees[82], random forests[83], KNN[84], XGBoost[85], AdaBoost[86], baseline

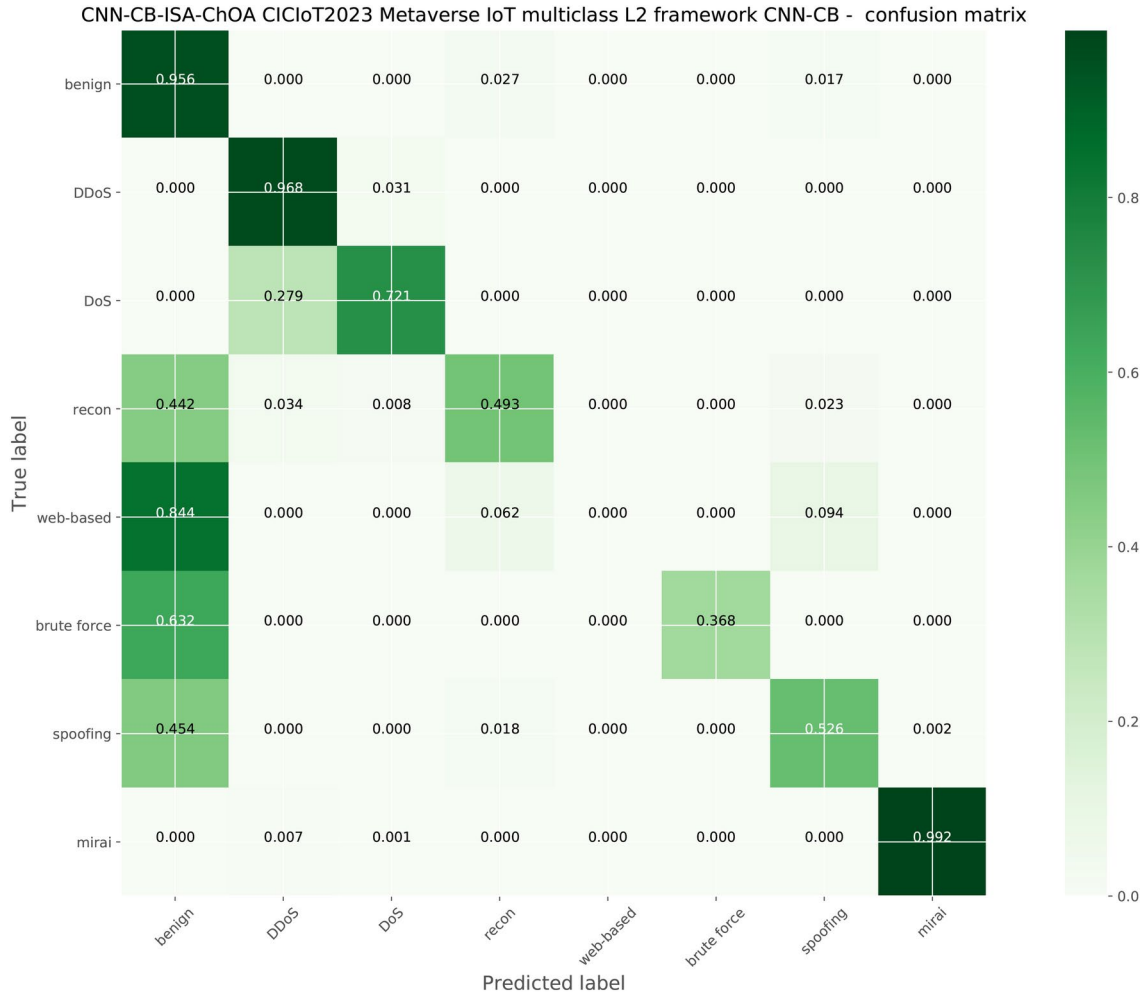

Fig. 12. Layer 2 CatBoost multiclass Best performing CNN-ISA-ChOA optimized model confusion matrix.

Algorithm	Best	Worst	Mean	Median	Std	Var
CNN-LGBM-ISA-ChOA	**0.996207**	**0.985756**	**0.991700**	**0.993378**	**0.003788**	**0.000014**
CNN-LGBM-ChOA	0.984145	0.945072	0.969630	0.969792	0.010915	0.000119
CNN-LGBM-VNS	0.990692	0.956404	0.979813	0.984732	0.012299	0.000151
CNN-LGBM-PSO	0.991862	0.937973	0.961005	0.958011	0.016631	0.000277
CNN-LGBM-BA	0.977663	0.929784	0.953932	0.957025	0.014720	0.000217
CNN-LGBM-ABC	0.973858	0.937252	0.960740	0.965956	0.012290	0.000151
CNN-LGBM-WOA	0.986708	0.945238	0.966248	0.967594	0.015793	0.000249
CNN-LGBM-RSA	0.986708	0.917790	0.967001	0.978231	0.022495	0.000506

Table 12. Layer 2 LightGBM multiclass objective function scores over 30 simulations. Best obtained metrics are shown in bold style.

CatBoost[46] and LGBM models as well as a simple multilayer perception (MLP)[87] models. The results of this comparative analysis are provided in detail in Table 16. The introduced hybrid framework shows clear advantages over other contemporary optimizers.

Statistical analysis and the best models interpretation

When conducting comparative simulations between optimizers several angles need to be considered when discerning a conclusion. Comparisons in terms of objective function scores are often insufficient to draw a definitive conclusion. Therefore statistical evaluations are conducted to establish if an attained improvement is significant. Two approaches can be taken when comparing metaheuristics, parametric and non-parametric testing. To ensure parametric tests can be safely applied a set of criteria needs to be fulfilled[88]. These include the

Algorithm	Best	Worst	Mean	Median	Std	Var
CNN-LGBM-ISA-ChOA	**0.001653**	**0.006199**	**0.003614**	**0.002885**	**0.001648**	**2.71E-06**
CNN-LGBM-ChOA	0.006898	0.023810	0.013188	0.013121	0.004723	2.23E-05
CNN-LGBM-VNS	0.004053	0.018914	0.008774	0.006644	0.005331	2.84E-05
CNN-LGBM-PSO	0.003544	0.026846	0.016910	0.018215	0.007193	5.17E-05
CNN-LGBM-BA	0.009712	0.030406	0.019975	0.018644	0.006359	4.04E-05
CNN-LGBM-ABC	0.011365	0.027180	0.017033	0.014782	0.005308	2.82E-05
CNN-LGBM-WOA	0.005786	0.023730	0.014649	0.014075	0.006835	4.67E-05
CNN-LGBM-RSA	0.005786	0.035461	0.014299	0.009465	0.009690	9.39E-05

Table 13. Layer 2 LightGBM multiclass objective function scores over 30 simulations. Best obtained metrics are shown in bold style.

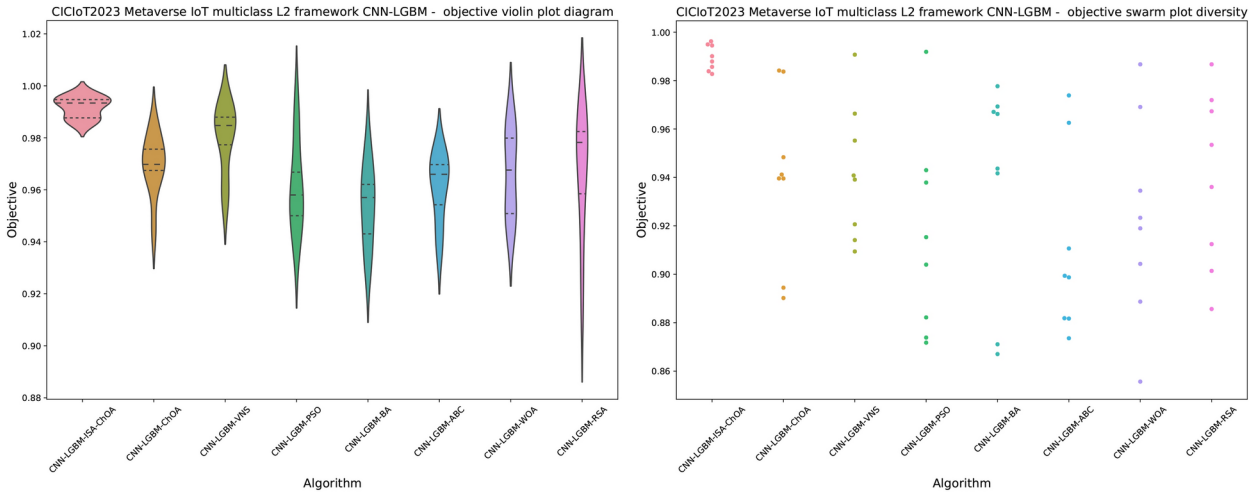

Fig. 13. Layer 2 LightGBM multiclass objective score distribution and swarm diagrams.

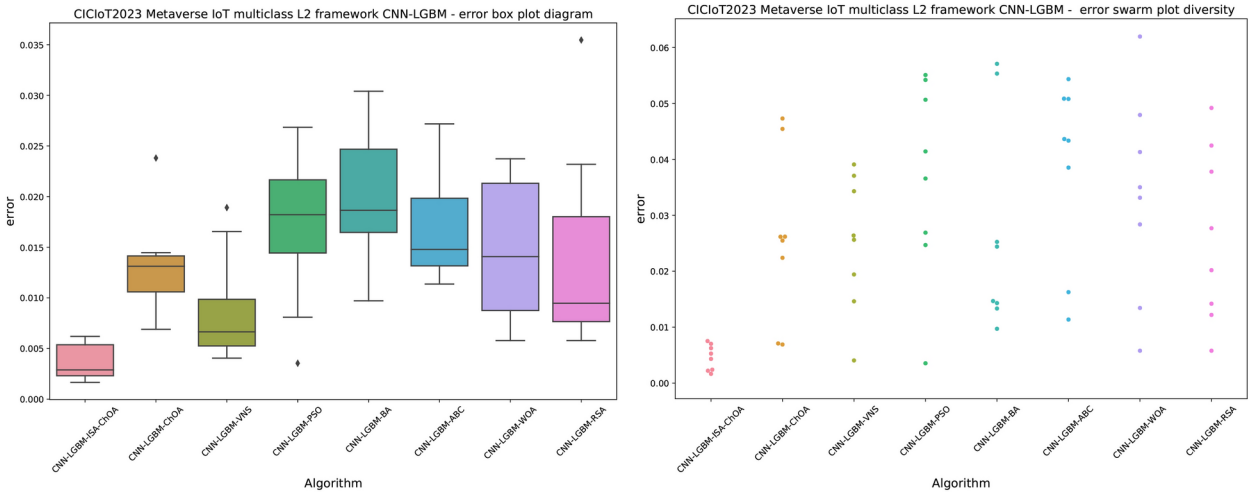

Fig. 14. Layer 2 LightGBM multiclass indicator score distribution and swarm diagrams.

independence of runs, a condition fulfilled by conduction individual optimizations using independent random seeds; Homoscedasticity, that is carrying out the Levene's test[89], and with all conditions for the conducted situations attaining a p-value of 0.62, this condition can be considered fulfilled as well; Normality for the attained scores must also be confirmed using the Shapiro-Wilk[90] test with p-values presented in Table 17. With p-values

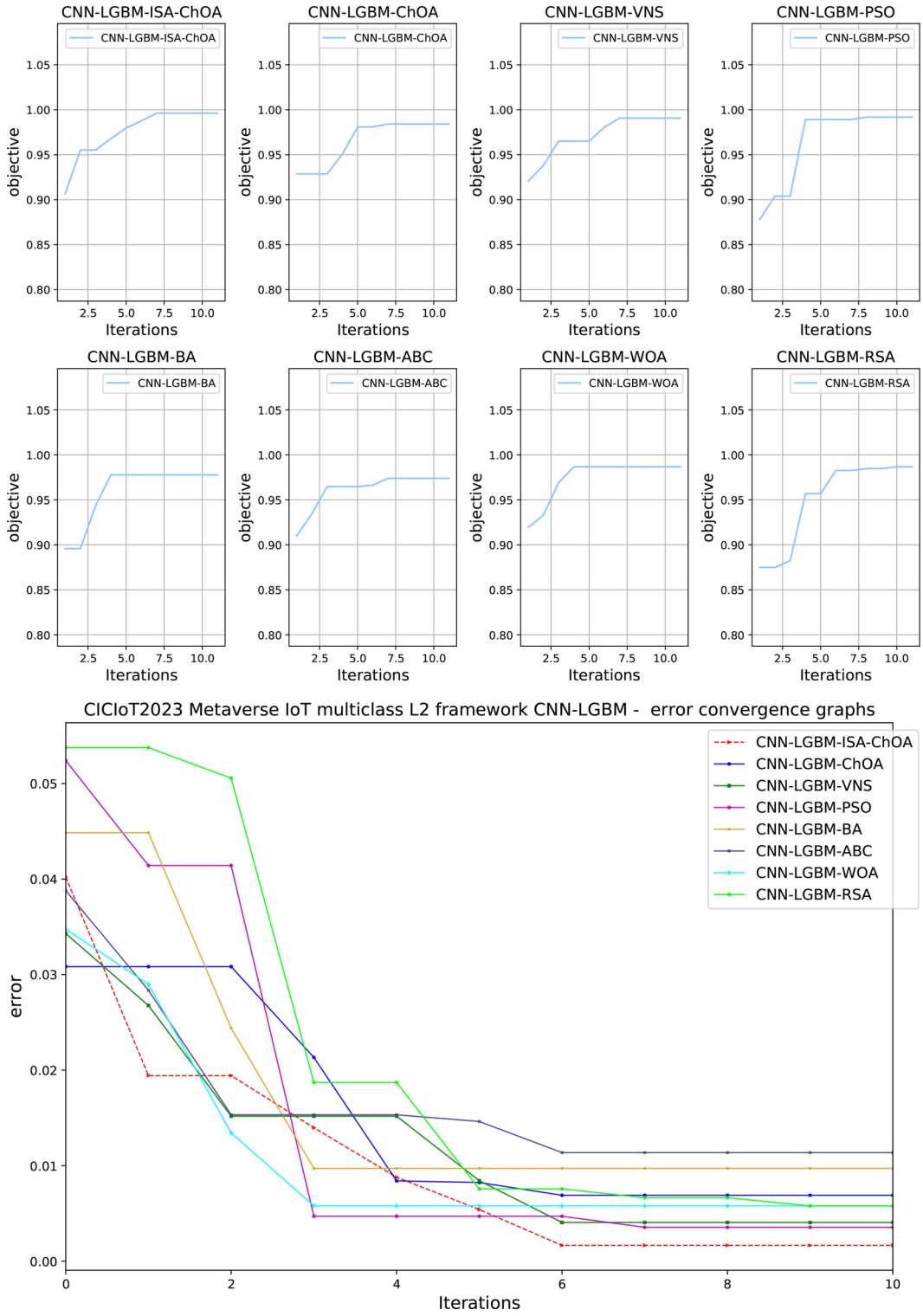

Fig. 15. Layer 2 LightGBM multiclass objective convergence diagrams.

not meeting the established criteria, normality cannot be confirmed, as the use of parametric test cannot be considered justified.

With the required normality condition not fulfilled non-parametric testing is applied to establish a further comparison. The Wilcoxon signed-rank test[91] is applied, and the ISA-ChOA algorithm is compared with other algorithms included in the comparative simulations and the p-value scores are presented in Table 18. As the significance threshold of $\alpha = 0.05$ is not exceeded, the outcomes attained in the comparative analysis can be considered statistically significant.

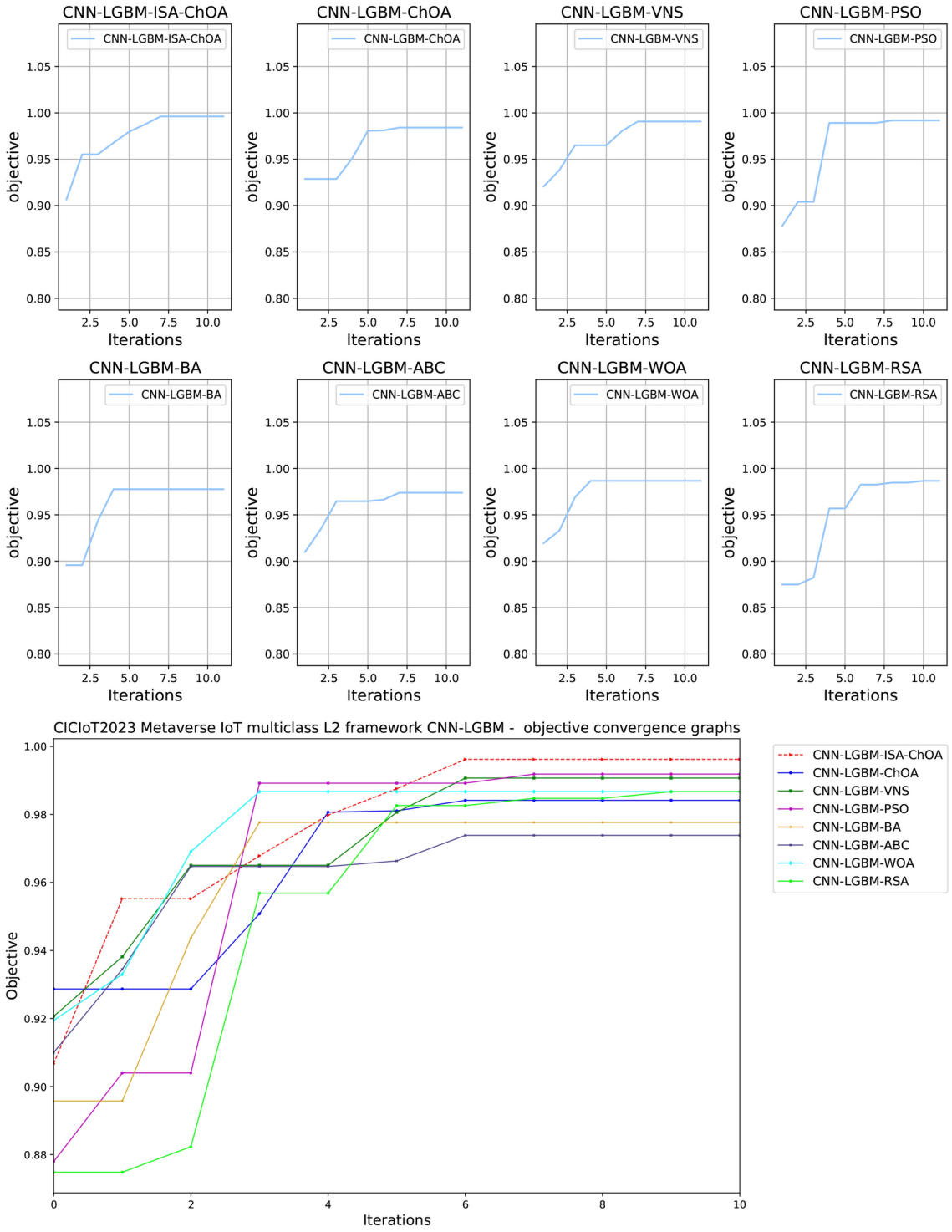

Fig. 16. Layer 2 LightGBM multiclass indicator convergence diagrams.

Interpretation of the best performing models

In modern AI research model classifications are without a doubt important. However, the contributing factors that allow a model to determine the class of a certain sample can also provide valuable feedback on a model decisions. Feature importance can help highlight hidden biases in the data as well as help reduce input and collected features for future research. While models are often treated by researchers as a black box, by leveraging advanced model interpretation tools, information on feature importances as well as their impact on classification can be computed.

Explainable AI (XAI) methods aim to make ML models significantly more transparent, interpretable, and trustworthy. XAI techniques help stakeholders comprehend how models determine decisions, increasing trust

Approach	Metric	Benign	Brute force	DDoS	DoS	Mirai	Recon	Spoofing	Web	Accuracy	Macro avg	Weighted avg
CNN-LGBM-ISA-ChOA	Precision	0.998639	0.998647	0.996685	1.000000	0.968750	0.900000	0.998489	1.000000	0.998346	0.982651	0.998347
	Recall	0.999319	0.999214	0.994306	1.000000	0.968750	0.947368	0.996983	0.999718	0.998346	0.988207	0.998346
	f1-score	0.998979	0.998930	0.995494	1.000000	0.968750	0.923076	0.997735	0.999859	0.998346	0.985353	0.998346
CNN-LGBM-ChOA	Precision	0.992563	0.993864	0.987433	0.997895	0.925926	1.000000	0.995475	1.000000	0.993102	0.986644	0.993082
	Recall	0.999319	0.997053	0.974102	0.993711	0.781250	0.842105	0.995475	1.000000	0.993102	0.947877	0.993102
	f1-score	0.995929	0.995456	0.980722	0.995798	0.847458	0.914286	0.995475	1.000000	0.993102	0.965641	0.993077
CNN-LGBM-VNS	Precision	0.997283	0.996513	0.992415	0.993711	1.000000	1.000000	0.990991	1.000000	0.995947	0.996364	0.995942
	Recall	0.999319	0.998210	0.985306	0.993711	0.875000	0.894737	0.995475	1.000000	0.995947	0.967720	0.995947
	f1-score	0.998300	0.997361	0.988848	0.993711	0.933333	0.944444	0.993228	1.000000	0.995947	0.981153	0.995939
CNN-LGBM-PSO	Precision	0.998638	0.996970	0.992888	1.000000	0.968750	0.947368	0.995482	1.000000	0.996456	0.987512	0.996452
	Recall	0.997958	0.998319	0.987235	1.000000	0.968750	0.947368	0.996983	1.000000	0.996456	0.987077	0.996456
	f1-score	0.998298	0.997644	0.990053	1.000000	0.968750	0.947368	0.996232	1.000000	0.996456	0.987293	0.996452
CNN-LGBM-BA	Precision	0.966822	0.992328	0.985598	0.945720	0.681818	0.000000	0.976923	0.998312	0.990288	0.818440	0.989932
	Recall	0.991831	0.996595	0.967857	0.949686	0.468750	0.000000	0.957768	0.998593	0.990288	0.791385	0.990288
	f1-score	0.979167	0.994457	0.976647	0.947699	0.555556	0.000000	0.967251	0.998453	0.990288	0.802404	0.990078
CNN-LGBM-ABC	Precision	0.977242	0.990826	0.978133	0.984816	0.892857	1.000000	0.977307	1.000000	0.988635	0.975148	0.988595
	Recall	0.993873	0.994892	0.961245	0.951782	0.781250	0.894737	0.974359	0.999719	0.988635	0.943982	0.988635
	f1-score	0.985488	0.992855	0.969616	0.968017	0.833333	0.944444	0.975831	0.999859	0.988635	0.958680	0.988593
CNN-LGBM-WOA	Precision	0.994565	0.995188	0.990070	0.993737	0.920000	0.666666	0.985031	0.997471	0.994214	0.942841	0.994161
	Recall	0.996596	0.997643	0.979704	0.997904	0.718750	0.421053	0.992459	0.998875	0.994214	0.887873	0.994214
	f1-score	0.995580	0.996414	0.984860	0.995816	0.807018	0.516129	0.988730	0.998172	0.994214	0.910340	0.994167
CNN-LGBM-RSA	Precision	0.990553	0.995253	0.990166	0.995798	0.676471	0.000000	0.990950	0.998033	0.994214	0.829653	0.993916
	Recall	0.999319	0.997664	0.980163	0.993711	0.718750	0.000000	0.990950	0.999156	0.994214	0.834964	0.994214
	f1-score	0.994917	0.996457	0.985139	0.994753	0.696970	0.000000	0.990950	0.998594	0.994214	0.832223	0.994059
	Support	1469	45812	10889	477	32	19	663	3554			

Table 14. Layer 2 LightGBM multiclass comprehensive metrics for best tuned models.

Approach	rounds	max_depth	Leaves	mcw	ff	bf	msg	λ L1	λ L2	lr
CNN-LGBM-ISA-ChOA	296	9	42	5	8.62E-01	5.15E-01	1.00E-02	1.14E-01	5.52E-01	5.83E-01
CNN-LGBM-ChOA	300	9	45	8	6.73E-01	8.39E-01	2.17E-02	4.88E-03	0.00E+00	3.77E-01
CNN-LGBM-VNS	300	9	45	5	2.90E-01	8.45E-01	1.51E-02	0.00E-00	1.73E+00	5.78E-01
CNN-LGBM-PSO	287	7	41	5	3.91E-01	9.48E-01	1.06E-03	0.00E-00	1.25E+00	9.00E-01
CNN-LGBM-BA	263	7	45	9	9.00E-01	9.67E-01	1.00E-03	1.31E+00	1.12E+00	5.26E-01
CNN-LGBM-ABC	212	10	33	5	6.59E-01	7.87E-01	1.00E-01	0.00E-00	3.52E-01	7.29E-01
CNN-LGBM-WOA	300	9	28	7	9.00E-01	5.00E-01	2.94E-02	1.12E-02	1.53E+00	6.22E-01
CNN-LGBM-RSA	277	8	45	11	6.13E-01	5.06E-01	1.14E-03	0.00E-00	1.99E-01	6.89E-01

Table 15. Best LightGBM model parameter selections made by each optimizer.

and aiding in regulatory compliance along with ethical AI considerations, which is vital in security applications[92]. XAI can help users understand which features are critical for predictions, enabling domain experts to validate the model's logic and performance. Finally, this process may aid in feature engineering by highlighting useful attributes and de-emphasizing redundant ones.

A notable contribution in terms of feature importance is the development and application of Shapely additive explanations (SHAP)[93] techniques. Based on game theory concepts SHAP analysis can help highlight feature importance on the global as well as local level. SHAP interpretations on a global scale are presented in Fig. 18, while per class interpretations are provided in swarm diagrams for each of the 7 glasses in Fig. 19. Additionally, it is important to note that SHAP analysis did not indicate any significant bias toward specific classes associated with certain features.

Conclusion and future work

Integrating the Metaverse with IoT is crucial, since IoT devices deliver real-time data and enable smooth connection betwixt the physical and virtual realms. Nevertheless, since attacks on IoT systems have become increasingly sophisticated, conventional security systems have struggled to keep up. Consequently, adaptive AI-driven methods were investigated to more appropriate tackle the challenges of today's IoT infrastructure and create safe environment for users. Effective AI models must manage intricate data correlations and remain

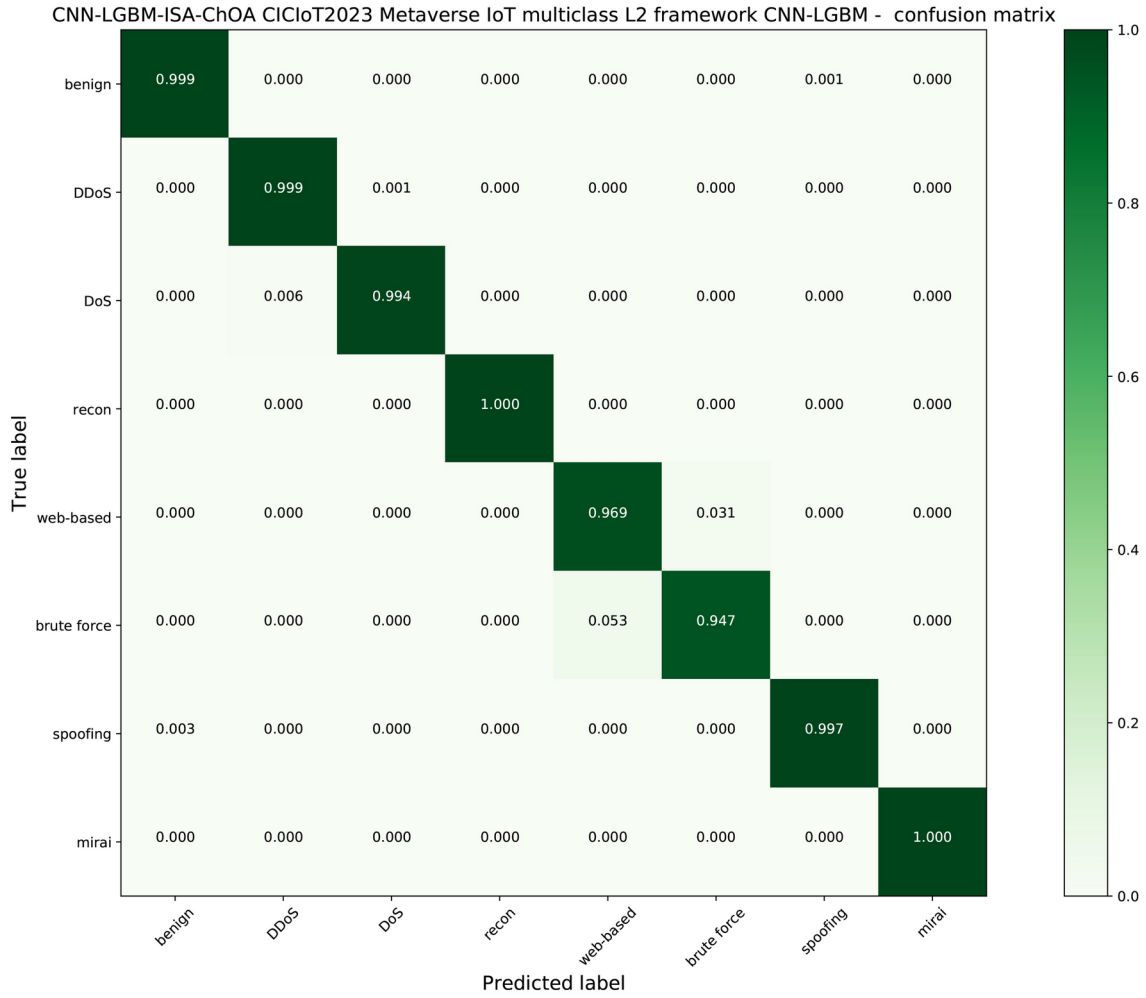

Fig. 17. Layer 2 LightGBM multiclass Best performing CNN-ISA-ChOA optimized model confusion matrix.

adaptable to evolving conditions. Achieving optimal results also required careful choice of algorithms and hyperparameter tuning. This study proposed a two-tier hybrid architecture that combines CNN with sophisticated ML classifiers, CatBoost and LightGBM. Metaheuristics techniques were employed to enhance performance, optimize the models, and refine parameter selection. Utilizing a realistic dataset, the framework was evaluated through comparative analysis, targeting multi-class classification to identify various types of attacks against IoT systems. A custom-altered optimizer was developed particularly for this study, resulting in the best-performing models, which attained a supreme accuracy level of 99.83% for multi-class classification. Afterwards, a rigorous statistical analysis outlined significant enhancements in comparison to the baseline metaheuristics and other contenting optimizers. Lastly, explainable AI method SHAP was employed on the best-performing model for understanding the significance of each feature and model's decision making process.

The methodology introduced in this study yielded a couple of benefits, notably achieving enhanced optimizer's performance over contemporary algorithms. Framework's two-tier architecture outperformed baseline CNNs while keeping computational demands within acceptable levels. For practical implementations, the suggested system might be deployed on IoT nodes for traffic management, requests processing, and mitigation of network-wide assaults. In the context of the Metaverse, this approach could improve general device safety, promoting trust and reinforcing the integration of virtual and physical domains. This advanced system could also support real-time attack detection in IoT by processing high-dimensional, streaming data, identifying anomalies, and mitigating threats promptly.

Although showcased study achieved promising results, some limitations still remain. The comparative evaluations included just a small selection of optimizing algorithms, and optimizations were executed with a relatively small population sizes and number of iterations. Thus, future work aims to tackle these constraints if supplementary computing resources become available. Expanding the pool of optimization algorithms and conducting evaluations with larger population sizes and iterations could provide more robust insights and even stronger conclusions. Additionally, the altered metaheuristics described here could be further applied to address other pressing challenges, enhancing performance and equipping scientists with improved tools for hyperparameter optimization of ML models. The application of the developed methods in real-time or streaming environments, where data evolves continuously, represents another promising avenue for development.

Approach	Metric	Benign	Brute force	DDoS	DoS	Mirai	Recon	Spoofing	Web	Accuracy	Macro avg	Weighted avg
CNN-ISA-ChOA	Precision	0.721480	0.922479	0.623072	0.792727	0.000000	1.000000	0.840000	0.999433	0.862656	0.737399	0.868015
	Recall	0.955752	0.901860	0.682524	0.457023	0.000000	0.052632	0.570136	0.991840	0.862656	0.576471	0.862656
	f1-score	0.822255	0.912053	0.651444	0.579787	0.000000	0.100000	0.679245	0.995622	0.862656	0.592551	0.863891
CNN-CB-ISA-ChOA	Precision	0.717203	0.935205	0.845409	0.785953	0.000000	1.000000	0.890306	0.999433	0.917778	0.771689	0.916141
	Recall	0.956433	0.968480	0.721187	0.492662	0.000000	0.368421	0.526395	0.991840	0.917778	0.628177	0.917778
	f1-score	0.819720	0.951552	0.778372	0.605670	0.000000	0.538462	0.661611	0.995622	0.917778	0.668876	0.914704
CNN-LGBM-ISA-ChOA	Precision	0.998639	0.998647	0.996685	1.000000	0.968750	0.900000	0.998489	1.000000	0.998346	0.982651	0.998347
	Recall	0.999319	0.999214	0.994306	1.000000	0.968750	0.947368	0.996983	0.999718	0.998346	0.988207	0.998346
	f1-score	0.998979	0.998930	0.995494	1.000000	0.968750	0.923076	0.997735	0.999859	0.998346	0.985353	0.998346
Decision Tree:	Precision	0.905971	0.999782	0.999449	0.790850	0.500000	0.250000	0.799419	0.999718	0.993340	0.780649	0.993354
	Recall	0.898570	0.999913	0.999265	0.761006	0.593750	0.210526	0.829563	0.999156	0.993340	0.786469	0.993340
	f1-score	0.902256	0.999847	0.999357	0.775641	0.542857	0.228571	0.814212	0.999437	0.993340	0.782772	0.993339
Random Forest	Precision	0.887280	0.999411	0.999908	0.837379	0.500000	0.000000	0.839204	0.999718	0.993928	0.757862	0.993424
	Recall	0.959156	0.999869	0.998714	0.723270	0.031250	0.000000	0.826546	0.998593	0.993928	0.692175	0.993928
	f1-score	0.921819	0.999640	0.999311	0.776153	0.058824	0.000000	0.832827	0.999155	0.993928	0.698466	0.993506
KNN	Precision	0.733920	0.932715	0.813870	0.676471	0.000000	0.666667	0.734310	0.997182	0.909592	0.694392	0.906557
	Recall	0.908781	0.961015	0.709156	0.530398	0.000000	0.105263	0.529412	0.995498	0.909592	0.592440	0.909592
	f1-score	0.812044	0.946653	0.757913	0.594595	0.000000	0.181818	0.615250	0.996339	0.909592	0.613076	0.906777
XGBoost	Precision	0.905161	0.999804	0.999724	0.880460	0.944444	1.000000	0.869832	1.000000	0.995279	0.949928	0.995289
	Recall	0.955071	0.999935	0.999173	0.802935	0.531250	0.210526	0.856712	1.000000	0.995279	0.794450	0.995279
	f1-score	0.929447	0.999869	0.999449	0.839912	0.680000	0.347826	0.863222	1.000000	0.995279	0.832466	0.995147
AdaBoost	Precision	0.741230	0.855708	0.935484	0.115854	0.089286	0.000000	0.580952	0.230851	0.691107	0.443671	0.822392
	Recall	0.992512	0.836899	0.002663	0.199161	0.156250	0.000000	0.092006	0.982836	0.691107	0.407791	0.691107
	f1-score	0.848661	0.846199	0.005311	0.146492	0.113636	0.000000	0.158854	0.373883	0.691107	0.311630	0.660863
CatBoost	Precision	0.880832	0.999585	0.999632	0.800000	0.200000	0.666667	0.826284	1.000000	0.993420	0.796625	0.992997
	Recall	0.950987	0.999716	0.998898	0.687631	0.031250	0.105263	0.825038	0.998875	0.993420	0.699707	0.993420
	f1-score	0.914566	0.999651	0.999265	0.739572	0.054054	0.181818	0.825660	0.999437	0.993420	0.714253	0.993052
LightGBM	Precision	0.755736	0.989259	0.998032	0.588372	0.050847	0.021739	0.544910	0.976068	0.975952	0.615621	0.976088
	Recall	0.695031	0.997184	0.978143	0.530398	0.093750	0.052632	0.549020	0.963984	0.975952	0.607518	0.975952
	f1-score	0.724113	0.993206	0.987988	0.557883	0.065934	0.030769	0.546957	0.969989	0.975952	0.609605	0.975943
MLP precision		0.805572	0.998886	0.993402	0.715347	0.075000	0.068966	0.767658	0.993545	0.987745	0.677297	0.987784
recall		0.905378	0.998145	0.995500	0.605870	0.093750	0.105263	0.622926	0.996061	0.987745	0.665362	0.987745
f1-score		0.852564	0.998515	0.994450	0.656073	0.083333	0.083333	0.687760	0.994801	0.987745	0.668854	0.987581
	Support	1469	45812	10889	477	32	19	663	3554			

Table 16. Detail metrics comparison between proposed framework models and state of the art baseline classifiers.

Approach	ISA-ChOA	ChOA	VNS	PSO	BA	ABC	WOA	RSA
Layer 1 CNN	0.028	0.036	0.029	0.041	0.024	0.028	0.032	0.041
Layer 2 CB	0.034	0.031	0.033	0.027	0.032	0.027	0.033	0.023
Layer 2 LGBM	0.029	0.031	0.032	0.037	0.029	0.031	0.024	0.029

Table 17. Shapiro-Wilk scores for forecasting experiments for normality condition evaluation.

ISA-ChOA vs. others	ChOA	VNS	PSO	BA	ABC	WOA	RSA
Layer 1 CNN	0.029	0.035	0.023	0.018	0.029	0.034	0.025
Layer 2 CB	0.035	0.029	0.025	0.044	0.037	0.035	0.028
Layer 2 LGBM	0.029	0.037	0.032	0.038	0.035	0.024	0.039

Table 18. Wilcoxon signed-rank test scores forecasting experiments.

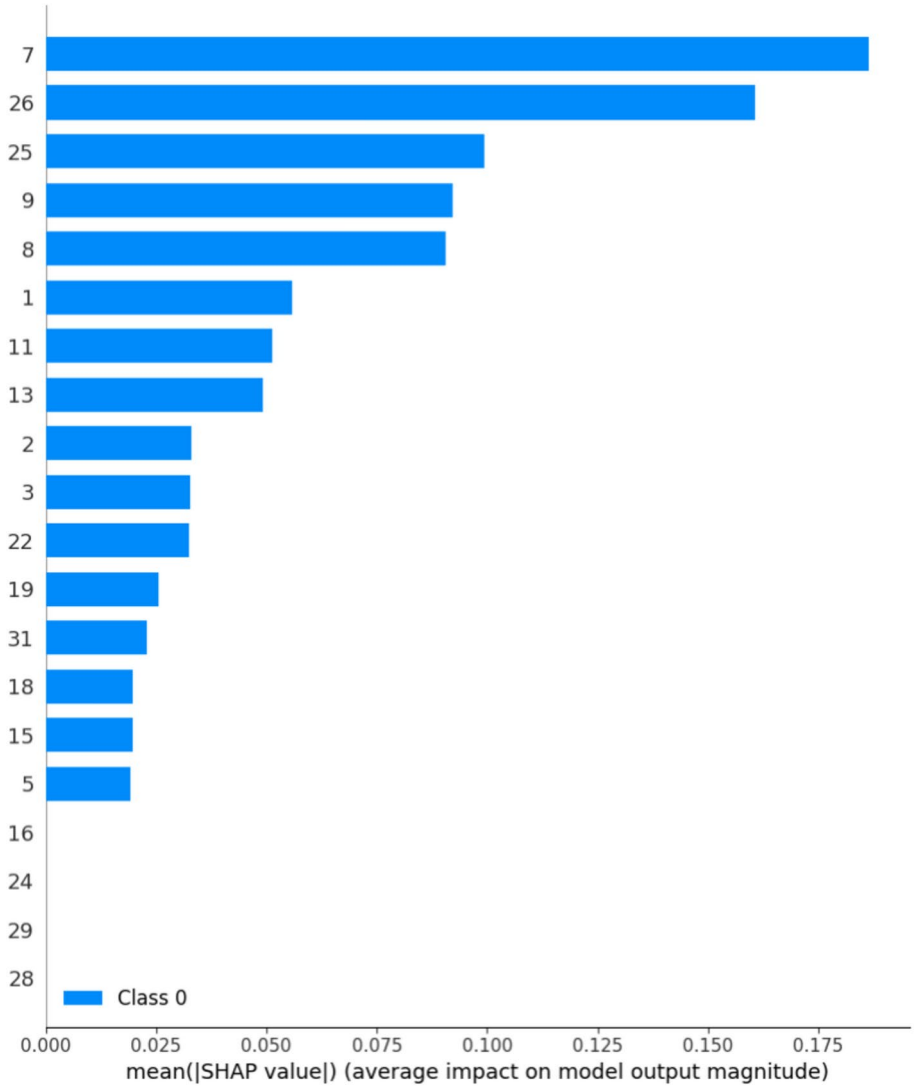

Fig. 18. Best performing model feature importance diagram.

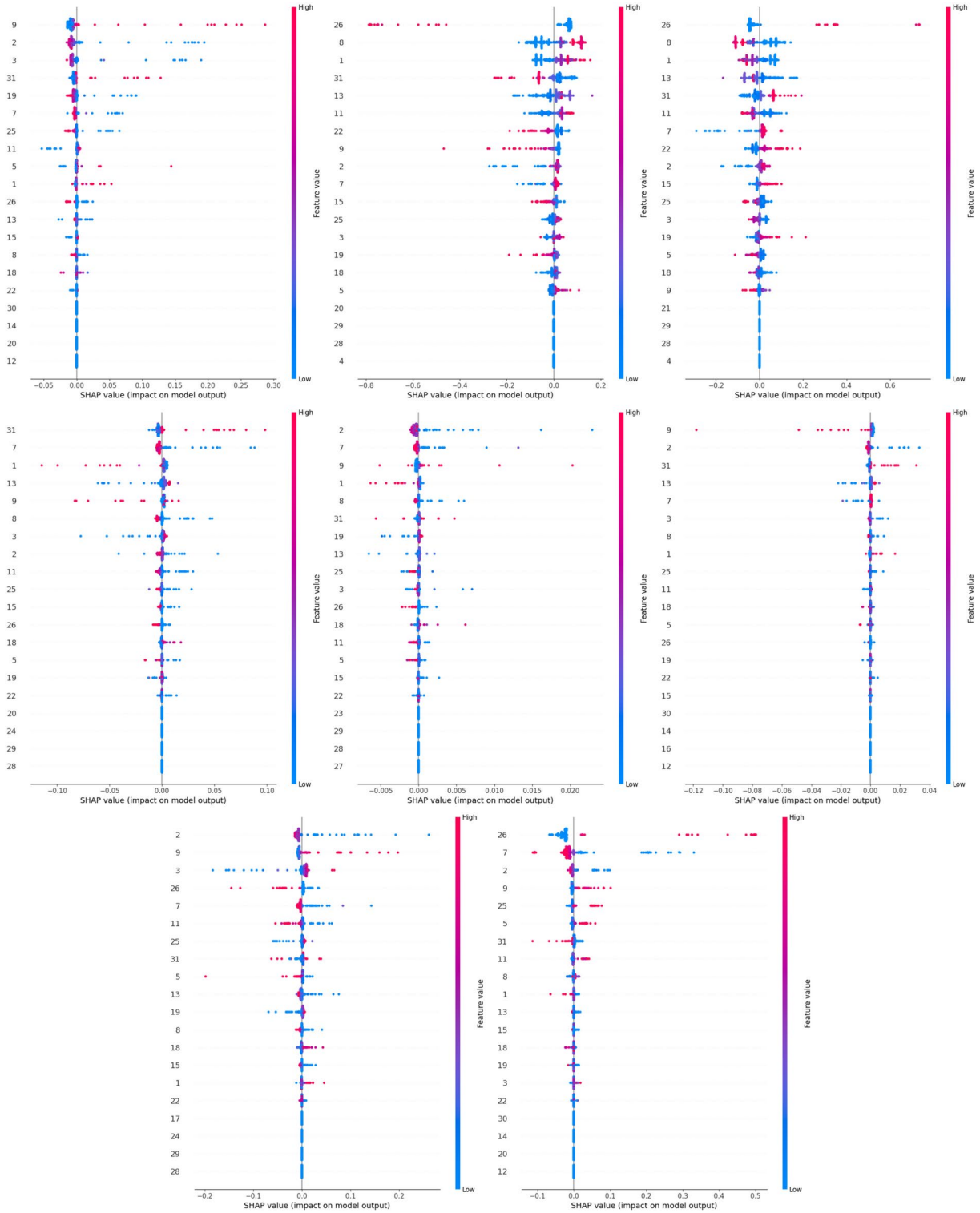

Fig. 19. Best performing model per class feature importance swarm diagrams.

Data availibility

The original dataset used in this study is freely available via URL: https://www.unb.ca/cic/datasets/index.html
Reduced dataset used in the experiments is available via URL: https://github.com/profzivkovic/CICIoT2023_IoT_Intrusion_reduced

Received: 2 November 2024; Accepted: 24 January 2025
Published online: 28 January 2025

References

1. Mystakidis, S. Metaverse. *Encyclopedia* **2**, 486–497 (2022).
2. Veeraiah, V. *et al.* Enhancement of meta verse capabilities by iot integration. In *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, 1493–1498 (IEEE, 2022).
3. Wang, H. et al. A survey on the metaverse: The state-of-the-art, technologies, applications, and challenges. *IEEE Internet Things J.* **10**, 14671–14688 (2023).
4. Heidari, A., Amiri, Z., Jamali, M. A. J. & Jafari, N. Assessment of reliability and availability of wireless sensor networks in industrial applications by considering permanent faults. *Concurr. Comput. Pract. Exp.* **36**, e8252 (2024).
5. Hwang, G.-J. & Chien, S.-Y. Definition, roles, and potential research issues of the metaverse in education: An artificial intelligence perspective. *Comput. Educ. Artif. Intell.* **3**, 100082 (2022).
6. Li, K. et al. When internet of things meets metaverse: Convergence of physical and cyber worlds. *IEEE Internet Things J.* **10**, 4148–4173 (2022).
7. Wang, Y. et al. A survey on metaverse: Fundamentals, security, and privacy. *IEEE Commun. Surv. Tutor.* **25**, 319–352 (2022).
8. Mrabet, H., Belguith, S., Alhomoud, A. & Jemai, A. A survey of iot security based on a layered architecture of sensing and data analysis. *Sensors* **20**, 3625 (2020).
9. Tawalbeh, L., Muheidat, F., Tawalbeh, M. & Quwaider, M. Iot privacy and security: Challenges and solutions. *Appl. Sci.* **10**, 4102 (2020).
10. Zhao, R., Zhang, Y., Zhu, Y., Lan, R. & Hua, Z. Metaverse: Security and privacy concerns. *J. Metaverse* **3**, 93–99 (2023).
11. Cheng, R., Chen, S. & Han, B. Toward zero-trust security for the metaverse. *IEEE Commun. Mag.* **62**, 156–162 (2023).
12. Huang, Y., Li, Y. J. & Cai, Z. Security and privacy in metaverse: A comprehensive survey. *Big Data Min. Anal.* **6**, 234–247 (2023).
13. Asadi, M., Jamali, M. A. J., Heidari, A. & Navimipour, N. J. Botnets unveiled: A comprehensive survey on evolving threats and defense strategies. *Trans. Emerg. Telecommun. Technol.* **35**, e5056 (2024).
14. Al-Garadi, M. A. et al. A survey of machine and deep learning methods for internet of things (iot) security. *IEEE Commun. Surv. Tutor.* **22**, 1646–1685 (2020).
15. Hussain, F., Hussain, R., Hassan, S. A. & Hossain, E. Machine learning in iot security: Current solutions and future challenges. *IEEE Commun. Surv. Tutor.* **22**, 1686–1721 (2020).
16. Wolpert, D. & Macready, W. No free lunch theorems for optimization. *IEEE Trans. Evol. Comput.* **1**, 67–82. https://doi.org/10.1109/4235.585893 (1997).
17. Petrovic, A. et al. Exploring metaheuristic optimized machine learning for software defect detection on natural language and classical datasets. *Mathematics* **12**, 2918 (2024).
18. Zivkovic, M. et al. Hybrid cnn and xgboost model tuned by modified arithmetic optimization algorithm for covid-19 early diagnostics from x-ray images. *Electronics* **11**, 3798 (2022).
19. Salb, M. et al. Enhancing internet of things network security using hybrid cnn and xgboost model tuned via modified reptile search algorithm. *Appl. Sci.* **13**, 12687 (2023).
20. Jovanovic, L. et al. Improving phishing website detection using a hybrid two-level framework for feature selection and xgboost tuning. *J. Web Eng.* **22**, 543–574 (2023).
21. Khishe, M. & Mosavi, M. Chimp optimization algorithm. *Expert Syst. Appl.* **149**, 113338. https://doi.org/10.1016/j.eswa.2020.113338 (2020).
22. Jia, H., Rao, H., Wen, C. & Mirjalili, S. Crayfish optimization algorithm. *Artif. Intell. Rev.* **56**, 1919–1979. https://doi.org/10.1007/s10462-023-10567-4 (2023).
23. Połap, D. & Woźniak, M. Red fox optimization algorithm. *Expert Syst. Appl.* **166**, 114107. https://doi.org/10.1016/j.eswa.2020.114107 (2021).
24. Abualigah, L., Elaziz, M. A., Sumari, P., Geem, Z. W. & Gandomi, A. H. Reptile search algorithm (rsa): A nature-inspired meta-heuristic optimizer. *Expert Syst. Appl.* **191**, 116158. https://doi.org/10.1016/j.eswa.2021.116158 (2022).
25. Thapa, S. & Mailewa, A. The role of intrusion detection/prevention systems in modern computer networks: A review. *In Conference: Midwest Instruction and Computing Symposium (MICS)* **53**, 1–14 (2020).
26. Amiri, Z., Heidari, A., Navimipour, N. J., Esmaeilpour, M. & Yazdani, Y. The deep learning applications in iot-based bio-and medical informatics: a systematic literature review. *Neural Comput. Appl.* **36**, 5757–5797 (2024).
27. Saheed, Y. K., Abdulganiyu, O. H., Majikumna, K. U., Mustapha, M. & Workneh, A. D. Resnet50-1d-cnn: A new lightweight resnet50-one-dimensional convolution neural network transfer learning-based approach for improved intrusion detection in cyber-physical systems. *Int. J. Crit. Infrastruct. Prot.* **45**, 100674 (2024).
28. Tsai, C.-F., Hsu, Y.-F. & Yen, D. C. Hybrid machine learning model for intrusion detection. *Appl. Sci.* **10**, 6620. https://doi.org/10.3390/app10196620 (2020).
29. Alamro, H. *et al.* Modelling of blockchain assisted intrusion detection on iot healthcare system using ant lion optimizer with hybrid deep learning. *IEEE Access* (2023).
30. Goran, R. *et al.* Identifying and understanding student dropouts using metaheuristic optimized classifiers and explainable artificial intelligence techniques. *IEEE Access* (2024).
31. Bacanin, N. *et al.* Improving performance of extreme learning machine for classification challenges by modified firefly algorithm and validation on medical benchmark datasets. *Multimedia Tools and Applications* 1–41 (2024).
32. Saheed, Y. K., Abdulganiyu, O. H. & Tchakoucht, T. A. Modified genetic algorithm and fine-tuned long short-term memory network for intrusion detection in the internet of things networks with edge capabilities. *Appl. Soft Comput.* **155**, 111434 (2024).
33. Saheed, Y. K., Omole, A. I. & Sabit, M. O. Ga-madam-iiot: A new lightweight threats detection in the industrial iot via genetic algorithm with attention mechanism and lstm on multivariate time series sensor data. *Sens. Int.* **6**, 100297 (2025).
34. Neto, E. C. P. et al. Ciciot 2023: A real-time dataset and benchmark for large-scale attacks in iot environment. *Sensors* **23**, 5941 (2023).
35. Krizhevsky, A., Sutskever, I. & Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Commun. ACM* **60**, 84–90 (2017).
36. Albawi, S., Bayat, O., Al-Azawi, S. & Ucan, O. N. Social touch gesture recognition using convolutional neural network. *Comput. Intell. Neurosci.* **2018**, 6973103 (2018).
37. Nair, V. & Hinton, G. E. Rectified linear units improve restricted boltzmann machines. In *Proc. of the 27th international conference on machine learning (ICML-10)*, 807–814 (2010).

38. Hinton, G. et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Process. Mag.* **29**, 82–97. https://doi.org/10.1109/MSP.2012.2205597 (2012).
39. Ranjan, R., Sankaranarayanan, S., Castillo, C. D. & Chellappa, R. An all-in-one convolutional neural network for face analysis. In *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, 17–24 (IEEE, 2017).
40. Lombardi, F. & Marinai, S. Deep learning for historical document analysis and recognition-a survey. *J. Imaging* **6**, 110 (2020).
41. Cai, L., Gao, J. & Zhao, D. A review of the application of deep learning in medical image classification and segmentation. *Ann. Transl. Med.***8** (2020).
42. Chattopadhyay, A., Hassanzadeh, P. & Pasha, S. Predicting clustered weather patterns: A test case for applications of convolutional neural networks to spatio-temporal climate data. *Sci. Rep.* **10**, 1317 (2020).
43. Bjekic, M. et al. Wall segmentation in 2d images using convolutional neural networks. *PeerJ Comput. Sci.* **9**, e1565 (2023).
44. Bukumira, M. et al. Carrot grading system using computer vision feature parameters and a cascaded graph convolutional neural network. *J. Electron. Imaging* **31**, 061815–061815 (2022).
45. Micci-Barreca, D. A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems. *ACM SIGKDD Explor. Newsl.* **3**, 27–32 (2001).
46. Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V. & Gulin, A. Catboost: unbiased boosting with categorical features. *Advances in Neural Information Processing Systems***31** (2018).
47. Ke, G. *et al.* Lightgbm: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems***30** (2017).
48. Sinha, B. B., Ahsan, M. & Dhanalakshmi, R. Lightgbm empowered by whale optimization for thyroid disease detection. *Int. J. Inf. Technol.* **15**, 2053–2062 (2023).
49. Guo, X. et al. Critical role of climate factors for groundwater potential mapping in arid regions: Insights from random forest, xgboost, and lightgbm algorithms. *J. Hydrol.* **621**, 129599 (2023).
50. Li, L. et al. A lightgbm-based strategy to predict tunnel rockmass class from tbm construction data for building control. *Adv. Eng. Inform.* **58**, 102130 (2023).
51. Lao, Z. et al. Intelligent fault diagnosis for rail transit switch machine based on adaptive feature selection and improved lightgbm. *Eng. Fail. Anal.* **148**, 107219 (2023).
52. Emambocus, B. A. S., Jasser, M. B. & Amphawan, A. A survey on the optimization of artificial neural networks using swarm intelligence algorithms. *IEEE Access* **11**, 1280–1294 (2023).
53. Tawhid, M. A. & Ibrahim, A. M. An efficient hybrid swarm intelligence optimization algorithm for solving nonlinear systems and clustering problems. *Soft. Comput.* **27**, 8867–8895 (2023).
54. Tang, J., Liu, G. & Pan, Q. A review on representative swarm intelligence algorithms for solving optimization problems: Applications and trends. *IEEE/CAA J. Autom. Sin.* **8**, 1627–1643 (2021).
55. Rostami, M., Berahmand, K., Nasiri, E. & Forouzandeh, S. Review of swarm intelligence-based feature selection methods. *Eng. Appl. Artif. Intell.* **100**, 104210 (2021).
56. Kennedy, J. & Eberhart, R. Particle swarm optimization. In *Proc. of ICNN'95 - International Conference on Neural Networks*, vol. 4, 1942–1948. https://doi.org/10.1109/ICNN.1995.488968 (1995).
57. Mirjalili, S. *Genetic Algorithm* 43–55 (Springer International Publishing, 2019).
58. Mladenović, N. & Hansen, P. Variable neighborhood search. *Comput. Oper. Res.* **24**, 1097–1100. https://doi.org/10.1016/S0305-0548(97)00031-2 (1997).
59. Karaboga, D. & Basturk, B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. *J. Glob. Optim.* **39**, 459–471. https://doi.org/10.1007/s10898-007-9149-x (2007).
60. Yang, X.-S. & He, X. Firefly algorithm: recent advances and applications. *Int. J. Swarm Intell.* **1**, 36–50 (2013).
61. Yang, X.-S. & He, X. Bat algorithm: literature review and applications. *Int. J. Bio-Inspired Comput.* **5**, 141–149. https://doi.org/10.1504/IJBIC.2013.055093 (2013).
62. Gurrola-Ramos, J., Hernàndez-Aguirre, A. & Dalmau-Cedeño, O. Colshade for real-world single-objective constrained optimization problems. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, 1–8, https://doi.org/10.1109/CEC48606.2020.9185583 (2020).
63. Bai, J. et al. A sinh cosh optimizer. *Knowl.-Based Syst.* **282**, 111081. https://doi.org/10.1016/j.knosys.2023.111081 (2023).
64. Damaševičius, R. *et al.* Decomposition aided attention-based recurrent neural networks for multistep ahead time-series forecasting of renewable power generation. *PeerJ Comput. Sci.***10** (2024).
65. Zivkovic, T., Nikolic, B., Simic, V., Pamucar, D. & Bacanin, N. Software defects prediction by metaheuristics tuned extreme gradient boosting and analysis based on shapley additive explanations. *Appl. Soft Comput.* **146**, 110659 (2023).
66. Predić, B. et al. Cloud-load forecasting via decomposition-aided attention recurrent neural network tuned by modified particle swarm optimization. *Complex Intell. Syst.* **10**, 2249–2269 (2024).
67. Vakili, A. et al. A new service composition method in the cloud-based internet of things environment using a grey wolf optimization algorithm and mapreduce framework. *Concurr. Comput. Pract. Exp.* **36**, e8091 (2024).
68. Stoean, C. et al. Metaheuristic-based hyperparameter tuning for recurrent deep learning: application to the prediction of solar energy generation. *Axioms* **12**, 266 (2023).
69. Velasco, L., Guerrero, H. & Hospitaler, A. A literature review and critical analysis of metaheuristics recently developed. *Arch. Comput. Methods Eng.* **31**, 125–146 (2024).
70. Babic, L. *et al.* Leveraging metaheuristic optimized machine learning classifiers to determine employee satisfaction. In *International Conference on Multi-Strategy Learning Environment*, 337–352 (Springer, 2024).
71. Pavlov-Kagadejev, M. et al. Optimizing long-short-term memory models via metaheuristics for decomposition aided wind energy generation forecasting. *Artif. Intell. Rev.* **57**, 45 (2024).
72. Dobrojevic, M. et al. Cyberbullying sexism harassment identification by metaheurustics-tuned extreme gradient boosting. *Comput. Mater. Contin.* **80**, 4997–5027 (2024).
73. Amiri, Z., Heidari, A., Zavvar, M., Navimipour, N. J. & Esmaeilpour, M. The applications of nature-inspired algorithms in internet of things-based healthcare service: A systematic literature review. *Trans. Emerg. Telecommun. Technol.* **35**, e4969 (2024).
74. Heidari, A., Shishehlou, H., Darbandi, M., Navimipour, N. J. & Yalcin, S. A reliable method for data aggregation on the industrial internet of things using a hybrid optimization algorithm and density correlation degree. *Cluster Computing* 1–19 (2024).
75. Zanbouri, K. et al. A gso-based multi-objective technique for performance optimization of blockchain-based industrial internet of things. *Int. J. Commun Syst.* **37**, e5886 (2024).
76. Savanović, N. et al. Intrusion detection in healthcare 4.0 internet of things systems via metaheuristics optimized machine learning. *Sustainability* **15**, 12563 (2023).
77. Dakic, P. et al. Intrusion detection using metaheuristic optimization within iot/iiot systems and software of autonomous vehicles. *Sci. Rep.* **14**, 22884 (2024).
78. Luo, W., Lin, X., Li, C., Yang, S. & Shi, Y. Benchmark functions for cec 2022 competition on seeking multiple optima in dynamic environments (2022).
79. Rahnamayan, S., Tizhoosh, H. R. & Salama, M. M. Quasi-oppositional differential evolution. In *2007 IEEE Congress on Evolutionary Computation*, 2229–2236 (IEEE, 2007).
80. Chicco, D. & Jurman, G. The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC Genom.* **21**, 1–13 (2020).

81. Mirjalili, S. & Lewis, A. The whale optimization algorithm. *Adv. Eng. Softw.* **95**, 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008 (2016).
82. de Ville, B. Decision trees. *WIREs Comput. Stat.* **5**, 448–455. https://doi.org/10.1002/wics.1278 (2013).
83. Breiman, L. Random forests. *Mach. Learn.* **45**, 5–32 (2001).
84. Kramer, O. *K-Nearest Neighbors* 13–23 (Springer, 2013).
85. Chen, T. & Guestrin, C. Xgboost: A scalable tree boosting system. In *Proc. of the 22nd ACM Sigkdd International Conference on Knowledge Discovery and Data Mining*, 785–794 (2016).
86. Hastie, T., Rosset, S., Zhu, J. & Zou, H. Multi-class adaboost. *Stat. Interface* **2**, 349–360 (2009).
87. Zou, J., Han, Y. & So, S.-S. *Overview of Artificial Neural Networks* 14–22 (Humana Press, 2009).
88. LaTorre, A. et al. A prescription of methodological guidelines for comparing bio-inspired optimization algorithms. *Swarm Evol. Comput.* **67**, 100973. https://doi.org/10.1016/j.swevo.2021.100973 (2021).
89. Schultz, B. B. Levene's test for relative variation. *Syst. Biol.* **34**, 449–456. https://doi.org/10.1093/sysbio/34.4.449 (1985).
90. Shapiro, S. S. & Francia, R. S. An approximate analysis of variance test for normality. *J. Am. Stat. Assoc.* **67**, 215–216. https://doi.org/10.1080/01621459.1972.10481232 (1972).
91. Woolson, R. F. *In Encyclopedia of Biostatistics* (Wiley, 2005). https://doi.org/10.1002/0470011815.b2a15177.
92. Saheed, Y. K. & Chukwuere, J. E. Xaiensembletl-iov: A new explainable artificial intelligence ensemble transfer learning for zero-day botnet attack detection in the internet of vehicles. *Results Eng.* **24**, 103171 (2024).
93. Lundberg, S. & Lee, S.-I. *A unified approach to interpreting model predictions* **1705**, 07874 (2017).

Acknowledgements
This research was supported by the Science Fund of the Republic of Serbia, grant No. 7373, characterizing crises-caused air pollution alternations using an artificial intelligence-based framework (crAIRsis), and grant No. 7502, Intelligent Multi-Agent Control and Optimization applied to Green Buildings and Environmental Monitoring Drone Swarms (ECOSwarm).

Author contributions
M.A. and M.Z. conceived the experiments, L.J. and N.B. conducted the experiments, M.D.J. and B.N. analyzed the results, J.P., M.A.S. and M.M. wrote first draft of the manuscript. All authors were included in writing the final version of the manuscript. All authors reviewed the manuscript.

Declarations

Competing interests
The authors declare no competing interests.

Additional information
Correspondence and requests for materials should be addressed to N.B.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.