Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Feature-Reduced IoT Intrusion Detection Using Clustering-Based Mutual Information and Meta-Heuristic CNN Optimization

S KUMAR REDDY MALLIDI

satya.cnis@gmail.com

Jawaharlal Nehru Technological University, Kakinada

Rajeswara Rao Ramisetty

Jawaharlal Nehru Technological University Gurajada-Vizianagaram

Research Article

Keywords: IoT Security, Intrusion Detection System (IDS), Bowerbird Courtship-Inspired Feature Selection (BBFS), Hunger Games Search (HGS), Feature Selection, Data Balancing, Compact IDS

Posted Date: August 11th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-8727074/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Feature-Reduced IoT Intrusion Detection Using Clustering-Based Mutual Information and Meta-Heuristic CNN Optimization

S Kumar Reddy Mallidi[1,2*] and Rajeswara Rao Ramisetty[3]

[1*]Computer Science and Engineering, Jawaharlal Nehru Technological University Kakinada, Kakinada, 533003, Andhra Pradesh, India.
[2]Computer Science and Engineering, Sri Vasavi Engineering College, Tadepalligudem, 534101, Andhra Pradesh, India.
[3]Computer Science and Engineering, Jawaharlal Nehru Technological University Gurajada, Vizianagaram, 535003, Andhra Pradesh, India.

*Corresponding author(s). E-mail(s): satya.cnis@gmail.com;
Contributing authors: raob4u.cse@jntugccev.edu.in;

Abstract

This work presents a compact and computationally efficient intrusion detection system (IDS) pipeline for Internet of Things (IoT) environments, integrating hybrid data balancing, filter-based feature reduction, and meta-heuristic hyperparameter optimization to construct an efficient one-dimensional Convolutional Neural Network (1D CNN). Class imbalance is addressed using an SMOTE–ENN–LOF sequence. At the same time, feature redundancy is mitigated by the Bowerbird Courtship-Inspired Feature Selection (BBFS) algorithm, which ranks features using a composite score that combines clustering-based Mutual Information (CMI), Fisher Score, and a correlation penalty. The Hunger Games Search (HGS) algorithm is employed to derive compact 1D CNN architectures by jointly considering attack-detection performance, trainable parameter count, and multiply–accumulate (MAC) operations. The proposed BBFS–HGS–1D CNN pipeline is evaluated on four diverse IoT intrusion detection datasets—ACI-IoT 2023, UQ-IoT 2021, Edge-IIoTset, and WUSTL-IIoT. The final models used compact feature subsets of 20/76, 23/76, 18/76, and 10/38 features, respectively, with trainable parameter counts ranging from 63,937 to 105,281 and MAC estimates ranging from 188,224 to 364,992. While the aggregate accuracy, precision, recall, and F1-score show only marginal gains on some saturated datasets, the proposed pipeline improves the reliability of attack detection by reducing

false negatives, with particular benefit for underrepresented attack categories. Comparative evaluation against six reimplemented IDS baselines shows that the proposed pipeline achieves competitive detection quality while using reduced feature subsets and compact CNN configurations, indicating its potential for future deployment-oriented IDS studies.

Keywords: IoT Security, Intrusion Detection System (IDS), Bowerbird Courtship-Inspired Feature Selection (BBFS), Hunger Games Search (HGS), Feature Selection, Data Balancing, Compact IDS

1 Introduction

The growing adoption of Internet of Things (IoT) technologies in healthcare, home automation, transportation, smart cities, and industrial systems has increased the need for reliable security and privacy mechanisms. IoT devices continuously generate and exchange data over heterogeneous networks while operating under limited processing, memory, and energy resources. These constraints make conventional security mechanisms difficult to deploy directly on IoT nodes, leaving such environments exposed to network- and software-based cyber threats. Consequently, designing effective and computationally efficient Intrusion Detection Systems (IDSs) for IoT has become an important research direction.

Although traditional IDS frameworks are effective in conventional enterprise networks, their computational and memory requirements often limit their suitability for IoT and edge environments. Machine learning (ML) and deep learning (DL)-based IDS models have therefore gained attention because they can learn traffic patterns and detect malicious activity from flow-based data. However, effective IoT IDS design is not merely a classifier-selection problem. Recent studies emphasize that IDS performance also depends on dataset quality, class imbalance handling, feature selection, computational efficiency, and deployment strategy. Hence, a practical IDS pipeline must jointly consider detection performance, feature compactness, class distribution, and model complexity.

1.1 Recent Literature and Research Gap

Recent years have witnessed a surge in research focusing on IDS tailored for IoT and Industrial IoT (IIoT) environments, with emphasis on feature engineering, ML/DL classification, optimization, and deployment-aware design. Kaushik et al. [1] introduced an IDS that uses a dedicated feature selection algorithm and reported high detection accuracy, thereby demonstrating the importance of targeted feature engineering in IoT security. Similarly, Ashraf et al. [2] developed a real-time IDS using ML techniques, where Random Forests (RF) produced strong detection performance on benchmark datasets. A modified RF-based IDS incorporating hybrid feature selection, one-hot encoding, and improved bootstrapping procedures has also been reported [3], with high accuracy on IoT-focused datasets such as IoTID20, UNSW-NB15, and IoT-23. In industrial settings, Particle Swarm Optimization-based feature selection has

been applied to IIoT intrusion detection [4], further demonstrating the relevance of swarm-based optimizers for reducing feature dimensionality in security datasets.

The integration of DL and hybrid models has also been widely explored for IoT intrusion detection. FogNet, a custom convolutional architecture designed for fog-based intrusion detection, achieved high accuracy on IoT traffic [5]. A systematic review of ML and DL techniques for IoT security highlighted the strengths, limitations, and research gaps that continue to shape IDS development [6]. Federated learning-based IDS frameworks have also been investigated to improve privacy and scalability in distributed IoT environments [7, 8]. These studies indicate that IoT IDS research is moving beyond centralized classification and toward more adaptive, privacy-preserving, and deployment-aware frameworks.

Several studies have also emphasized benchmarking, model selection, and optimization. Comparative analyses of ML classifiers show that the choice of algorithm and hyperparameter tuning substantially influence the performance of IoT intrusion detection [**?**]. Multi-layer IDS architectures using cloud and edge resources have been proposed to improve adaptability and responsiveness to emerging threats [9]. Additional work has focused on compact IDS designs for efficient, real-time operation in resource-constrained IoT settings [10, 11]. In parallel, explainable artificial intelligence models have been introduced to improve trust and interpretability in IDS decisions [12], while studies on feature selection and extraction have examined how dimensionality reduction affects both detection quality and computational cost [13]. Studies on IIoT anomaly detection further demonstrate the need for hybrid analytical approaches capable of handling complex cyber-physical traffic patterns [14]. Broader survey efforts have synthesized these trends and identified open challenges related to IoT IDS design, evaluation, and deployment [15, 16].

Although these studies demonstrate substantial progress, several methodological gaps remain. First, many IoT IDS studies report high aggregate accuracy, precision, recall, and F1-score, but such binary or aggregate metrics may hide per-attack weaknesses, particularly in imbalanced datasets. This is especially important in IoT IDS, where missing low-frequency attack categories may be more harmful than marginal gains in overall accuracy. Second, feature selection, data balancing, and model optimization are often studied independently, making it difficult to determine the contribution of each component to the final IDS performance. Third, several studies claim compact operation based mainly on reduced feature count or model size, without sufficient analysis of computational indicators such as parameter count, multiply–accumulate (MAC) operations, or statistical significance. Finally, some IoT datasets exhibit saturated binary classification performance, where simple baselines can achieve very high scores. Therefore, ablation analysis, per-class evaluation, statistical testing, and computational-efficiency assessment are necessary to establish the actual contribution of a proposed IDS pipeline.

The above gaps are summarized in Table 1, which maps common limitations in recent IoT IDS studies to the present work's corresponding focus.

Table 1: Summary of common limitations in recent IoT IDS studies and the corresponding focus of this work.

Research Direction	**Common Limitation**	**Focus of the Present Work**
ML-based IDS	Fast and interpretable models may still be affected by redundant flow features and class imbalance.	Applies hybrid data balancing and Bowerbird Courtship-Inspired Feature Selection (BBFS)-based feature reduction before classification.
DL-based IDS	High detection performance is often reported, but model compactness and computational cost are not always analyzed.	Uses Hunger Games Search (HGS) to obtain compact one-dimensional Convolutional Neural Network (1D CNN) configurations, and reports computational indicators such as parameters and MACs.
Feature-selection IDS	Many methods emphasize either feature relevance or redundancy, but nonlinear feature–label dependencies are not always adequately considered.	Uses clustering-based Mutual Information (CMI) to assess nonlinear feature–label relevance, with correlation-based redundancy control.
Balancing-based IDS	Oversampling may introduce synthetic bias or noisy samples if not combined with suitable cleaning mechanisms.	Combines Synthetic Minority Oversampling Technique (SMOTE) with Edited Nearest Neighbors (ENN) and Local Outlier Factor (LOF), and evaluates its effect through component-wise analysis.
Evaluation practices	Aggregate binary metrics may hide attack-wise weaknesses, especially in saturated IoT IDS datasets.	Extends evaluation beyond aggregate scores by including per-class analysis, with attention to false-negative reduction in underrepresented attack categories, ablation-based comparison of scoring variants, and confusion-matrix-based statistical testing where applicable.

1.2 Challenges in IoT IDS Implementation

Implementing IDS in IoT ecosystems poses several interrelated challenges:

- **Resource Constraints:** IoT devices often operate with limited processing power, memory, and energy, restricting the direct deployment of conventional security mechanisms [17, 18].
- **Data Imbalance:** IoT intrusion datasets commonly show highly uneven class distributions, where benign traffic and high-volume attacks such as DoS, DDoS, or reconnaissance dominate, while attacks such as brute force, malware, injection, or exploitation remain underrepresented [19, 20].
- **Feature Redundancy and Relevance:** Flow-based IoT security datasets often contain redundant, correlated, or weakly informative attributes, making feature selection important for improving detection accuracy and computational efficiency [19, 21].
- **Scalability and Adaptability:** IDS models must operate across heterogeneous and dynamically changing IoT environments while adapting to new and unseen attack patterns, making a universal "one-size-fits-all" solution difficult to realize [22, 23].
- **Privacy and Data Distribution:** The distributed and privacy-sensitive nature of IoT deployments complicates centralized data aggregation and model training [24, 25].

These challenges motivate an IDS pipeline that optimizes the complete data-to-model process rather than relying only on classifier selection. Accordingly, this work addresses class imbalance using hybrid balancing and cleaning, reduces redundant traffic attributes through BBFS-based feature selection, and tunes the 1D CNN classifier using HGS to obtain compact configurations while maintaining competitive detection performance.

1.3 Contributions and Organization of the Paper

This study introduces a computationally efficient IDS pipeline for flow-based IoT traffic classification by integrating SMOTE–ENN–LOF balancing, BBFS-based feature selection, and HGS-driven 1D CNN hyperparameter optimization. Unlike approaches that report only final classification scores, this work emphasizes component-wise validation to clarify the individual effects of balancing, feature selection, CMI-guided scoring, redundancy penalization, and CNN optimization.

The underlying search mechanism of BBFS is inherited from the authors' earlier work; therefore, the novelty of the present study lies not in the BBFS search process itself, but in its adaptation and evaluation using CMI-guided scoring variants within a balanced and HGS-optimized CNN IoT IDS pipeline. The key contributions of this work are summarized below:

- A multiclass hybrid balancing and cleaning strategy using SMOTE, ENN, and LOF is incorporated to reduce class imbalance and noisy synthetic samples before feature selection and model training.

- CMI-guided BBFS feature selection is investigated using CMI–correlation and CMI–Fisher–correlation scoring variants to evaluate the roles of nonlinear relevance, class separability, and redundancy control.
- A unified IDS pipeline is constructed by combining hybrid balancing, BBFS-based feature reduction, and HGS-driven 1D CNN optimization for compact flow-based traffic classification with improved sensitivity to attack instances.
- A component-wise ablation study is conducted to quantify the effects of balancing, BBFS feature selection, CMI-based scoring, correlation penalization, and HGS-based CNN tuning.
- The proposed pipeline is evaluated on four IoT/IIoT datasets using aggregate metrics, per-attack-class analysis, confusion-matrix-based statistical testing, and computational-efficiency indicators such as selected feature count, parameter count, and MACs.

The remainder of this paper is organized as follows. Section 2 presents the proposed IoT IDS pipeline, including data balancing, BBFS-based feature selection, and HGS-based optimization of the 1D CNN. Section 3 describes the datasets, preprocessing steps, evaluation metrics, and experimental setup. Section 4 presents the results, ablation studies, statistical analysis, and comparative discussion. Section 5 concludes the paper and outlines future research directions.

2 Proposed IoT-IDS Pipeline

2.1 Overview of the Proposed Pipeline

The overall workflow of the proposed IoT intrusion detection pipeline is illustrated in Fig. 1. The pipeline begins with flow-based traffic preparation. For datasets available as packet capture (PCAP) traces, traffic is converted into flow-level features using CICFlowMeter. In contrast, datasets already provided as flow records are processed according to their original feature format.

After data splitting, the training subset is processed using SMOTE–ENN–LOF to reduce class imbalance and remove noisy or density-inconsistent samples. Feature refinement is then performed using BBFS in filter mode with CMI-guided scoring variants that combine nonlinear feature–label relevance, class separability, and redundancy control. The resulting feature reduct is applied consistently to both training and test sets.

The reduced training set is used to construct a compact 1D CNN, whose key hyperparameters are optimized using HGS. The final model is evaluated on the held-out test set using detection metrics, selected feature count, trainable parameters, MAC operations, and attack-type-wise analysis. This modular design enables component-wise evaluation of balancing, feature reduction, and CNN optimization across multiple IoT/IIoT datasets.

2.2 Data Preprocessing and Balancing

The raw or dataset-provided flow-based feature sets are preprocessed to improve data consistency before feature selection and model construction. Duplicate records and

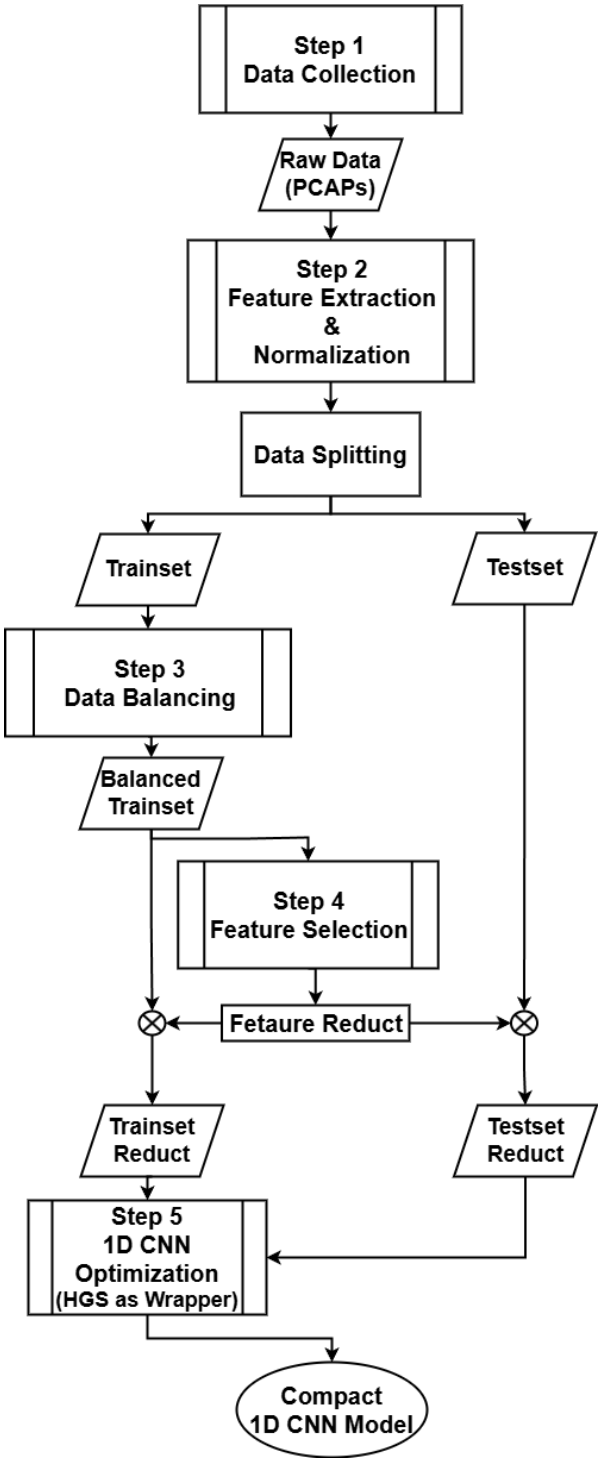


Fig. 1: Overview of the proposed IoT-IDS pipeline.

constant features are removed where applicable, and numerical features are scaled using parameters learned only from the training data. Since IoT intrusion datasets often contain noisy instances and highly imbalanced class distributions, preprocessing is performed carefully to avoid data leakage and to support stable classifier training.

The dataset is first partitioned into training and test subsets before balancing and feature selection. All data-dependent operations, including scaling, balancing, feature scoring, and hyperparameter optimization, are fitted or performed only on the training set. The test set remains untouched by oversampling, cleaning, feature scoring, and model selection, and is transformed only using parameters learned from the training data.

Class imbalance in the training subset is handled using the SMOTE–ENN–LOF sequence. SMOTE generates synthetic instances for minority attack classes to reduce bias toward majority classes, while ENN removes ambiguous or locally inconsistent samples after oversampling. LOF is then applied to remove density-inconsistent samples that may remain after SMOTE–ENN processing. Thus, the combined sequence reduces class imbalance while limiting the influence of noisy synthetic samples and abnormal original records before BBFS-based feature selection and CNN training.

The choice of SMOTE–ENN–LOF is motivated by the authors' earlier IIoT IDS study [26], in which multiple balancing strategies, including SMOTE, ADASYN, SMOTE–ENN, ADASYN–ENN, SMOTE–ENN–LOF, and ADASYN–ENN–LOF, were empirically evaluated. Since SMOTE–ENN–LOF provided balanced improvement in minority-class detection while reducing noisy or density-inconsistent samples, the present work adopts it as an empirically supported preprocessing sequence rather than as an arbitrary balancing choice.

2.3 Bowerbird Courtship-Inspired Feature Selection (BBFS)

Feature selection is important for high-dimensional IoT and IIoT intrusion detection because flow-based datasets often contain redundant, correlated, or weakly informative attributes that increase model complexity without necessarily improving detection performance. Existing feature-selection approaches generally include filter, wrapper, embedded, and hybrid methods. Nature-inspired optimization algorithms have also been widely used for this task; however, many were originally designed for continuous optimization and therefore require transfer functions, such as S-shaped and V-shaped mappings, to convert continuous position updates into binary feature-subset decisions [27].

In the authors' earlier work, BBFS was introduced as a feature-selection method inspired by the selective and iterative refinement behavior of bowerbirds [28]. The present study retains the existing BBFS search mechanism and adapts it to a filter-based IoT IDS setting using CMI-guided scoring variants. Specifically, BBFS is evaluated with CMI and correlation penalization, as well as with CMI, Fisher-score assistance, and correlation penalization. This formulation enables the feature-selection process to compare the contributions of nonlinear feature–label relevance, class separability, and inter-feature redundancy within the proposed balanced and CNN-optimized IoT IDS pipeline.

2.3.1 BBFS Algorithm in Filter Mode

In the filter configuration adopted in this work, each bower in BBFS represents a candidate feature subset, and subset quality is evaluated without invoking a classifier. The search operators are retained from the earlier BBFS formulation, while the filter-level scoring function is generalized to support Fisher–correlation, CMI–correlation, and CMI–Fisher–correlation variants. Thus, candidate subsets are ranked by jointly considering nonlinear feature–label relevance, class separability, and redundancy reduction.

For a candidate subset B_i, the generalized composite scoring function is shown in Equation 1. Here, f_j and f_k denote selected features in B_i, while α, β, and γ control the relative contributions of CMI, Fisher score, and correlation penalty, respectively. The correlation term penalizes absolute pairwise correlation among selected features.

$$\mathcal{F}(B_i) = \alpha \sum_{f_j \in B_i} \mathrm{CMI}(f_j) + \beta \sum_{f_j \in B_i} \mathrm{Fisher}(f_j) - \gamma \sum_{\substack{f_j, f_k \in B_i \\ j<k}} |\mathrm{Corr}(f_j, f_k)| \tag{1}$$

The weights satisfy $\alpha + \beta + \gamma = 1$, with $\gamma = 1 - (\alpha + \beta)$. By varying these weights, the same scoring function produces the Fisher–correlation, CMI–correlation, and CMI–Fisher–correlation variants used in the ablation study. CMI quantifies the dependency between the cluster representation of a continuous feature and the class label. For a feature f, clustering-based partitioning converts continuous values into discrete cluster states. Let c denote the cluster state obtained from feature f, and let y denote the class label. The CMI score is calculated using Equation 2.

$$\mathrm{CMI}(f) = \sum_{c,y} p(c,y) \log \frac{p(c,y)}{p(c)p(y)} \tag{2}$$

Higher CMI values indicate stronger nonlinear and distribution-aware relevance between the feature and the target class.

The Fisher score measures inter-class separability relative to intra-class variance. For a feature vector x, let n_c, μ_c, and σ_c^2 denote the number of samples, mean, and variance of class c, respectively, and let μ denote the global feature mean. The Fisher score is calculated using Equation 3.

$$\mathrm{Fisher}(x) = \frac{\sum_{c=1}^{C} n_c(\mu_c - \mu)^2}{\sum_{c=1}^{C} n_c \sigma_c^2} \tag{3}$$

Larger Fisher values indicate stronger discriminative ability. The correlation penalty in Equation 1 discourages the selection of highly correlated features within the same reduct, thereby promoting compact and less redundant feature subsets.

During the BBFS update process, candidate features from the guiding bower are added only when they produce a positive marginal improvement in subset fitness. For a candidate feature f_k, the temporarily updated subset is denoted as B_i^{new}. The positive marginal gain is calculated using Equation 4.

$$g_k = \max\left(0, \mathcal{F}(B_i^{new} \cup \{f_k\}) - \mathcal{F}(B_i^{new})\right) \tag{4}$$

If at least one candidate feature has a positive marginal gain, the feature-addition probability is calculated using Equation 5. Here, C_i denotes the set of candidate features obtained from the guiding bower.

$$P_k = \frac{g_k}{\sum_{f_l \in C_i} g_l} \tag{5}$$

Candidate features are sampled according to this probability distribution and added to the current bower. The complete filter-based BBFS procedure is presented in Algorithm 1. The final reduct $\mathbb{R}$ is applied consistently to both training and testing sets before CNN training and HGS-based hyperparameter tuning.

2.3.2 Computational Complexity of BBFS

The computational cost of the proposed BBFS filter stage includes one-time feature-score preprocessing and iterative population-based subset search. Let n denote the number of training samples used for feature selection, d the number of candidate features after constant-feature removal, N the BBFS population size, T the number of iterations, and $\bar{s}$ the average number of selected features in a candidate subset. The CMI preprocessing stage discretizes each continuous feature using K-Means clustering over candidate cluster values in the range $[K_{\min}, K_{\max}]$, giving an approximate cost of $O(d \cdot K \cdot n \cdot I_k)$, where K denotes the number of tested cluster values and I_k denotes the K-Means iteration cost. Fisher-score computation requires $O(nd)$, while constructing the absolute correlation matrix requires $O(nd^2)$. These preprocessing steps are performed once, and the resulting CMI scores, Fisher scores, and correlation matrix are reused during BBFS subset evaluation.

During the BBFS search, the CMI and Fisher relevance terms require $O(\bar{s})$ time for a candidate subset. In contrast, the correlation penalty requires $O(\bar{s}^2)$ time because it accumulates pairwise correlations among selected features. Thus, the dominant cost of one subset evaluation is $O(\bar{s}^2)$, and the iterative BBFS search cost is $O(TN\bar{s}^2)$. The overall BBFS complexity is summarized in Equation 6, where the first three terms correspond to CMI, Fisher-score, and correlation preprocessing, and the final term corresponds to iterative subset search.

$$O\left(d \cdot K \cdot n \cdot I_k + nd + nd^2 + TN\bar{s}^2\right) \tag{6}$$

Since BBFS is used as an offline feature-selection stage, this cost is incurred during model construction rather than during IDS inference. Once the reduct is selected, it remains fixed and is reused for CNN training and final testing, where the computational cost depends only on the reduced input dimensionality and the optimized CNN architecture.

2.4 Baseline 1D CNN Architecture

The BBFS-selected feature subset serves as the input representation for the 1D CNN model used in the proposed IDS pipeline. Since the input consists of flow-level numerical features rather than raw packet sequences, the selected features are arranged into

Algorithm 1 Filter-Based BBFS with Generalized CMI–Fisher–Correlation Scoring

Require: Full feature set $\mathbb{F}$, precomputed CMI scores, Fisher scores, absolute correlation matrix, weight parameters α, β, γ, and subset fitness function $\mathcal{F}(\cdot)$
Ensure: Feature reduct $\mathbb{R} \subseteq \mathbb{F}$

Set control parameters: population size N, maximum iterations T, top fraction ρ_t, worst fraction ρ_w, reinitialization window q, and Lévy parameter λ_L.
Generate the initial bower population $\mathbb{B} = \{B_1, B_2, \ldots, B_N\}$, where each $B_i \subseteq \mathbb{F}$ is formed by random feature inclusion.
If any $B_i = \emptyset$, randomly assign one feature from $\mathbb{F}$ to B_i.
Evaluate $\mathcal{F}(B_i)$ for each bower $B_i \in \mathbb{B}$ using Equation 1.
Initialize the best reduct as $\mathbb{R} \leftarrow \arg\max_{B_i \in \mathbb{B}} \mathcal{F}(B_i)$.
Initialize the worst-group counter of each bower to zero.
for $t = 1$ to T **do**
    Sort bowers based on fitness values.
    Assign the best $\rho_t N$ bowers to the top-performing group.
    Assign the lowest $\rho_w N$ bowers to the worst-performing group.
    Treat the remaining bowers as the neutral group.
    for each bower $B_i \in \mathbb{B}$ **do**
        Select a candidate guiding bower $B_j \neq B_i$ using Lévy-flight.
        Initialize $B_i^{new} \leftarrow B_i$.
        if B_j belongs to the top-performing group **then**
            Identify candidate features $C_i \leftarrow B_j \setminus B_i^{new}$.
            Compute the fitness gain g_k for each $f_k \in C_i$ using Equation 4.
            if at least one candidate feature has positive marginal gain **then**
                Compute feature-addition probabilities using Equation 5.
                Sample candidate feature(s) from C_i using the probabilities.
                Add the selected feature(s) to B_i^{new}.
            end if
        end if
        Set $base \leftarrow \mathcal{F}(B_i^{new})$.
        for each feature $f_k \in B_i^{new}$ **do**
            if removing f_k does not reduce the subset fitness **then**
                Remove f_k from B_i^{new}.
                Update $base \leftarrow \mathcal{F}(B_i^{new})$.
            end if
        end for
        Update $B_i \leftarrow B_i^{new}$ and recalculate $\mathcal{F}(B_i)$.
        if $\mathcal{F}(B_i) > \mathcal{F}(\mathbb{R})$ **then**
            Update the best reduct: $\mathbb{R} \leftarrow B_i$.
        else if $\mathcal{F}(B_i) = \mathcal{F}(\mathbb{R})$ and $|B_i| < |\mathbb{R}|$ **then**
            Update the best reduct using the more compact subset: $\mathbb{R} \leftarrow B_i$.
        end if
    end for
    for each bower B_i in the worst-performing group **do**
        Increase its worst-group counter.
        if the counter reaches q **then**
            Reinitialize B_i using random feature inclusion.
            If $B_i = \emptyset$, randomly assign one feature from $\mathbb{F}$ to B_i.
            Recalculate $\mathcal{F}(B_i)$ and reset its counter.
        end if
    end for
    Reset worst-group counters for top-performing and neutral bowers.
end for
return Best feature reduct $\mathbb{R}$

a fixed one-dimensional vector and reshaped to match the 1D CNN input format. The baseline architecture is used as a structured template for HGS-based hyperparameter optimization, rather than as the final manually fixed model.

The baseline 1D CNN consists of three convolutional blocks followed by fully connected layers. Each convolutional block includes a 1D convolutional layer, batch normalization, and max-pooling. The convolutional layers learn local interaction patterns among neighboring selected features, while max-pooling reduces the intermediate representation. The resulting feature maps are flattened and passed through two dense layers with batch normalization, ReLU activation, and dropout. A sigmoid output layer then produces the binary class probability for benign and attack traffic. Figure 2 illustrates the baseline architecture used as the starting point for HGS-driven optimization.

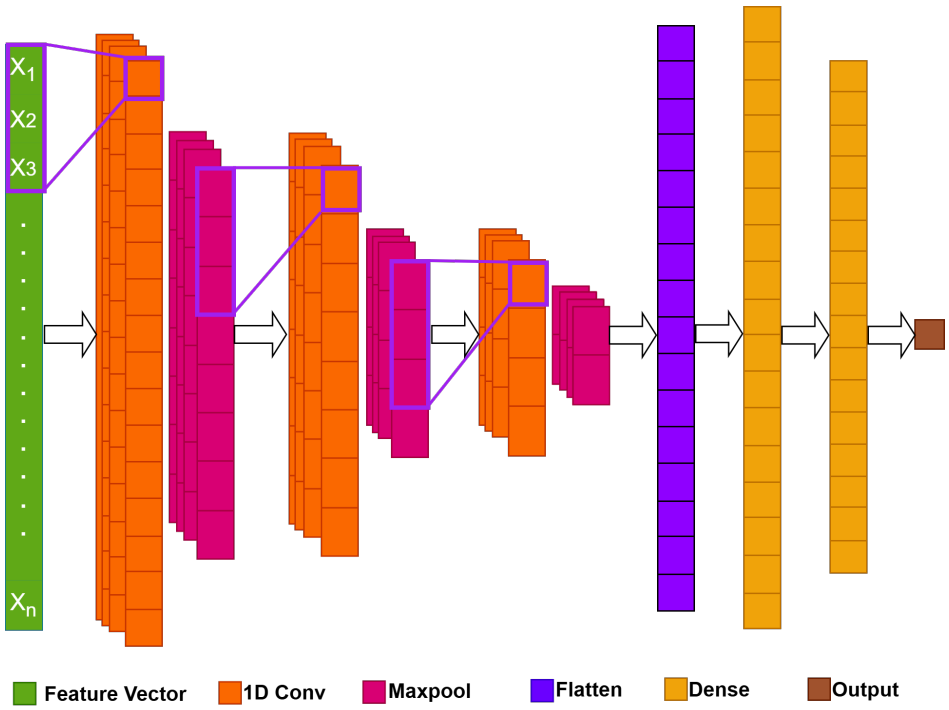


Fig. 2: Baseline 1D CNN architecture used as a benchmark and the starting point for HGS-driven optimization.

The use of a 1D CNN is motivated by its ability to learn local and hierarchical feature-interaction patterns while maintaining a compact DL structure. Although flow-level IDS features are tabular and do not have the natural temporal ordering of packet sequences, the same fixed feature order is maintained during training and testing. Under this consistent representation, convolutional filters can capture local feature-neighborhood patterns, and deeper layers can combine them into higher-level

representations. Compared with fully connected MLPs, convolutional layers share weights across local windows and can reduce the number of trainable parameters for a given representation size. However, the baseline architecture is not assumed to be optimal; its filter counts, kernel sizes, pooling factor, dense-layer widths, dropout rates, ℓ_2 regularization strength, learning rate, and batch size are optimized using HGS. A systematic comparison with tree-based models, MLPs, attention-based models, and alternative feature-ordering strategies is left for future work.

2.5 HGS-Based Hyperparameter Optimization of the 1D CNN

Although BBFS reduces the input dimensionality, the final detection performance and computational complexity of the 1D CNN still depend strongly on its hyperparameter configuration. Important hyperparameters include convolutional filter counts, kernel sizes, pooling factor, dense-layer widths, dropout rates, ℓ_2 regularization strength, learning rate, and batch size. Since manual tuning or an exhaustive grid search becomes inefficient when these parameters are optimized jointly, the proposed pipeline employs HGS to tune the 1D CNN configuration automatically.

HGS is not introduced in this study as a new meta-heuristic optimizer or classifier. Instead, it is adopted as a population-based hyperparameter search strategy within the proposed IDS pipeline. The contribution of this stage lies in integrating HGS with BBFS-reduced feature sets, recall-aware threshold selection, and compactness-aware CNN fitness evaluation. Each HGS agent represents a candidate 1D CNN configuration encoded as a real-valued vector $\mathbf{z} \in [0,1]^D$, where D is the number of hyperparameters. This vector is decoded into a concrete CNN configuration through deterministic mappings: discrete hyperparameters are selected from predefined candidate sets using index-based mapping, whereas continuous hyperparameters are obtained using linear or log-uniform transformations. The exact search-space values, training settings, and dataset-specific optimization parameters are reported in Section 3.2.6.

2.5.1 Encoding, Evaluation, and Fitness Function

Each HGS agent represents a candidate 1D CNN configuration and is decoded into a concrete model architecture. During candidate evaluation, the decoded model is trained using binary cross-entropy loss and the Adam optimizer. Class weights are applied to account for the binary class distribution, while learning-rate decay and early stopping based on validation loss are used to reduce unnecessary training and limit overfitting during the HGS search. After training, validation probabilities are generated, and a threshold search is performed over a predefined grid to identify the decision threshold that maximizes the custom F_β-score.

The custom F_β-score is used instead of the ordinary F1-score because it allows attack recall to be emphasized by setting $\beta > 1$. For each candidate threshold, precision and recall are computed for the attack class. In Equation 7, β controls the relative emphasis on recall, and the dataset-specific value of β is reported in the experimental configuration.

$$F_\beta = \frac{(1+\beta^2) \cdot \text{Precision} \cdot \text{Recall}}{\beta^2 \cdot \text{Precision} + \text{Recall}} \tag{7}$$

The validation threshold that yields the highest F_β-score is retained as the candidate threshold. In case of equal F_β-scores, the threshold with higher recall is preferred; if recall is also tied, the threshold producing fewer false positives is selected. This threshold-aware evaluation is important because the final IDS decision depends on both the learned CNN probabilities and the operating threshold used to separate benign and attack flows.

In addition to validation F_β, the evaluation records the number of trainable parameters and the estimated multiply–accumulate (MAC) operations required for a forward pass. The MAC count is estimated analytically by propagating layer shapes through the convolutional and dense layers. These quantities are used to compute the compact-complexity score in Equation 8, where $\widetilde{\text{Params}}$ and $\widetilde{\text{MACs}}$ denote normalized parameter and MAC counts, respectively, and w_p and w_m control their relative importance.

$$\text{LW} = w_p \, \widetilde{\text{Params}} + w_m \, \widetilde{\text{MACs}} \tag{8}$$

The overall HGS fitness assigned to a candidate hyperparameter vector $\mathbf{z}$ is shown in Equation 9. Here, $\alpha_h \in [0, 1]$ controls the trade-off between recall-aware validation performance and architectural compactness. A higher α_h gives greater importance to F_β, whereas a lower value increases the influence of the compactness term. Since $(1 - \text{LW})$ increases as the model becomes smaller and computationally less expensive, maximizing $\Phi(\mathbf{z})$ favors configurations that achieve high validation F_β-score with lower parameter and MAC counts.

$$\Phi(\mathbf{z}) = \alpha_h \, F_\beta + (1 - \alpha_h)(1 - \text{LW}) \tag{9}$$

The values of α_h, w_p, w_m, and β are fixed before test-set evaluation and are reported in the experimental configuration for reproducibility. The compactness term is used only during hyperparameter optimization. Final model quality is reported separately using test-set classification metrics, selected feature count, parameter count, MAC estimates, model size, median test-set inference time, and attack-type detection analysis.

2.5.2 HGS Search Procedure

The HGS optimization process begins by sampling an initial population of agents uniformly within the normalized search range $[0, 1]^D$. Each agent is decoded into a 1D CNN configuration, trained, and evaluated using the fitness function in Equation 9. The best-performing candidate is stored as the current global best. During each iteration, agents are ranked by fitness, and an elite group is retained to guide the population update.

The position-update equations follow the standard HGS formulation, while this work customizes the decoding and fitness-evaluation stages for 1D CNN optimization. Candidate positions are generated using elite solutions, the elite center, the current global best, and a time-varying exploration–exploitation coefficient. The updated positions are clipped to $[0, 1]^D$, decoded into CNN configurations, trained, and evaluated. Greedy replacement and elitism are then applied to retain improved candidates. If the global best does not improve for a predefined number of iterations, a diversification

step perturbs or reinitializes selected low-performing agents. The complete HGS-based hyperparameter optimization procedure is summarized in Algorithm 2.

Algorithm 2 HGS-Based Hyperparameter Optimization for the 1D CNN

Require: BBFS-reduced training data $(X_{\text{tr}}, y_{\text{tr}})$, search dimension D, fitness function $\Phi(\cdot)$ in Equation 9

Ensure: Best hyperparameter vector $\mathbf{z}^*$, optimized threshold θ^*, and corresponding 1D CNN configuration

Choose population size N, maximum iterations T, elite fraction, and stagnation limit.
Sample initial agents $\{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N\}$ uniformly in $[0,1]^D$.
for each agent $\mathbf{z}_i$ **do**
    Decode $\mathbf{z}_i$ into a 1D CNN configuration.
    Train and validate the decoded 1D CNN.
    Select the validation threshold that maximizes F_β using Equation 7.
    Compute parameter count, MAC estimate, compact-complexity score, and fitness $\Phi(\mathbf{z}_i)$.
end for
Set the global best $\mathbf{z}^*$ as the agent with the maximum fitness and store threshold θ^*.
for $t = 1$ to T **do**
    Rank agents by fitness and identify elite and low-performing agents.
    Update candidate positions using the standard HGS hunger-driven position-update mechanism.
    Clip the updated candidates to $[0,1]^D$ and decode them into 1D CNN configurations.
    for each updated candidate **do**
        Train and validate the corresponding 1D CNN.
        Select the validation threshold that maximizes F_β.
        Compute the fitness value using Equation 9.
    end for
    Accept updated candidates that improve fitness and preserve elite candidates.
    if the global best does not improve for the predefined stagnation limit **then**
        Diversify a subset of low-performing agents.
    end if
    Update $\mathbf{z}^*$ and θ^* whenever a better candidate is found.
end for
return $\mathbf{z}^*$, θ^*, and its decoded 1D CNN configuration.

After HGS identifies the best hyperparameter configuration, the final 1D CNN is rebuilt using the selected configuration and trained on the BBFS-reduced balanced training data. The trained model is evaluated on the original test set after converting the multiclass labels into binary form using the same benign-versus-attack mapping. The original multiclass labels are retained only for attack-type-wise post-analysis and

are not used to train the binary classifier. In addition to aggregate binary metrics, the evaluation records the confusion matrix, attack-class detection analysis, selected feature count, trainable parameters, and MAC estimates.

3 Experimental Setup

This section describes the datasets, preprocessing procedures, and experimental conditions used to evaluate the proposed BBFS–HGS–1D CNN intrusion detection pipeline. The selected IoT/IIoT datasets represent heterogeneous environments, including smart-home networks, academic IoT deployments, industrial testbeds, and edge–cloud-oriented IoT scenarios. Their diversity enables evaluation of the proposed method under different traffic distributions, attack types, and operational conditions.

3.1 Datasets

3.1.1 Edge-IIoT Dataset

The Edge-IIoT dataset [29] was generated from a purpose-built IoT/IIoT cybersecurity testbed containing sensors, edge nodes, gateways, and cloud/edge configurations. It includes realistic industrial and consumer IoT communication patterns with 14 attack types covering DoS/DDoS, information gathering, man-in-the-middle, injection, and malware categories. Its heterogeneous testbed structure makes it suitable for evaluating IDS performance across mixed IoT and IIoT traffic conditions.

3.1.2 ACI-IoT 2023 Dataset

The ACI-IoT dataset [30] was developed in the IoT Research Lab (IoTRL) at the Army Cyber Institute using a realistic smart-home environment with wired and wireless IoT devices managed through Home Assistant. It contains benign traffic and attacks such as reconnaissance scans, OS probing, vulnerability scanning, DoS variants, dictionary-based brute force, and ARP spoofing. This controlled but realistic setup provides representative smart-home traffic for IoT intrusion detection.

3.1.3 UQ-IoT-IDS-2021 Dataset

The UQ-IoT dataset [31] was collected from a realistic IoT deployment at the University of Queensland. It includes benign traffic from devices such as smartphones, smart TVs, IP cameras, smart speakers, and Raspberry Pi nodes, along with attack traffic involving host discovery, port scanning, service detection, ARP spoofing, brute-force, and flooding attacks. The dataset provides high-volume traces from an operational academic IoT network.

3.1.4 WUSTL-IIoT Dataset

The WUSTL-IIoT dataset [32] originates from an IIoT testbed designed to emulate industrial control system communication. It includes benign traffic and attacks such as DoS, reconnaissance, command injection, and backdoor attempts. The dataset exhibits

natural class imbalance, making it useful for evaluating flow-based intrusion detection in industrial IoT environments.

3.2 Data Preprocessing

All datasets undergo a unified preprocessing workflow to support consistent evaluation. Since this study focuses on flow-based intrusion detection, PCAP-derived flow records are used where raw packet traces are available, and dataset-provided flow records are used where PCAP files are not publicly released.

3.2.1 Flow Generation

For Edge-IIoT, ACI-IoT 2023, and UQ-IoT, raw PCAP files were converted into bidirectional flow records using CICFlowMeter, producing 76 statistical traffic-flow features. Since the WUSTL-IIoT PCAP traces are not publicly released, the 41 Argus-derived flow features provided by the dataset creators were used.

3.2.2 Data Cleaning and Normalization

Duplicate flow records were removed to reduce repeated samples and prevent bias toward replicated traffic patterns. Non-numeric, identifier, and metadata fields were discarded, and missing values were handled using median imputation. Continuous features were scaled to stabilize 1D CNN training and ensure comparable feature ranges across heterogeneous datasets. Duplicate removal was applied before train–test partitioning. In contrast, imputation and normalization parameters were estimated only from the training split and then applied to the test split to avoid information leakage.

3.2.3 Class Label Consolidation and Train–Test Partitioning

Although the datasets contain multiple attack types, this study adopts a binary intrusion detection formulation. Benign flows are labeled as class 0, and attack flows are labeled as class 1. This formulation supports deployment-oriented decision-making, where the primary requirement is to distinguish normal and malicious behavior. The original multiclass labels are retained only for post-hoc attack-type-wise analysis.

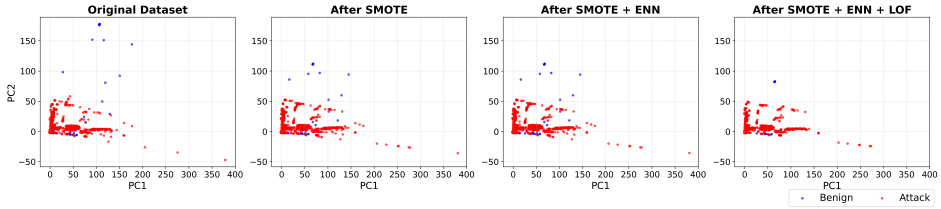
Each dataset is partitioned into training and testing splits using stratified 80:20 sampling to preserve class distribution. Only the training split is subjected to balancing and feature selection, while the test split remains untouched for unbiased evaluation of the final optimized model.

3.2.4 Hybrid Class Balancing

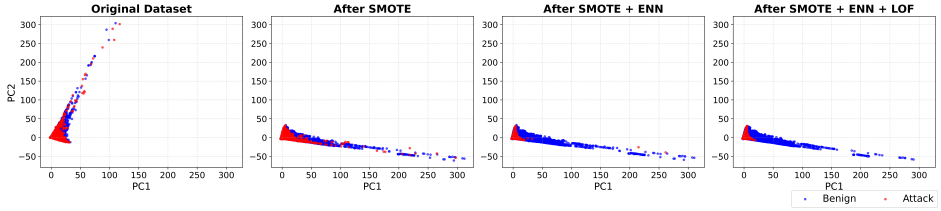
All four IoT datasets exhibit substantial class imbalance, with benign traffic dominating and several attack categories appearing sparsely. To obtain a stable training distribution for feature selection and 1D CNN optimization, the training split is processed using a three-stage SMOTE–ENN–LOF balancing pipeline.

SMOTE is first applied using a class-wise sampling strategy to increase minority-class representation. The oversampled data is then refined using ENN with $k = 5$,

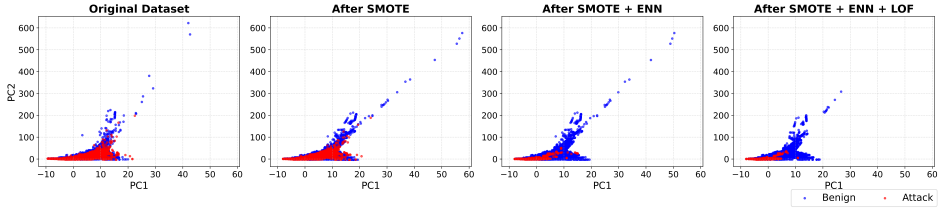
which removes locally inconsistent or borderline samples while retaining legitimate minority instances. Finally, LOF is applied with $k = 10$ to remove density-inconsistent samples that may remain after oversampling and neighborhood cleaning.



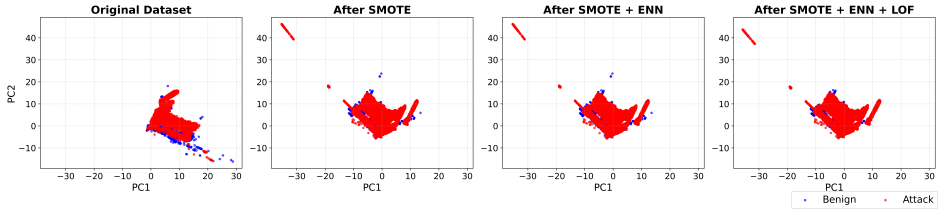
(a) Edge-IIoT



(b) ACI-IoT



(c) UQ-IoT-IDS



(d) WUSTL-IIoT

Fig. 3: PCA-based visualization of hybrid class balancing across the four datasets.

The resulting balanced and noise-reduced training set is used consistently for BBFS-based feature selection and HGS-driven 1D CNN training. Figure 3 illustrates the progressive effect of the balancing stages using two-dimensional principal component analysis (PCA) projections. These PCA projections are used only for visualization and are not used for feature selection or model training.

3.2.5 BBFS Filter Setup

BBFS was employed in a filter configuration to derive compact and informative flow-feature subsets across all four datasets. Each candidate subset was evaluated using the generalized scoring function in Equation 1. Two CMI-guided variants were considered: CMI with correlation penalization and CMI with Fisher-score-assisted correlation penalization. The first variant measures nonlinear feature–label relevance while penalizing inter-feature redundancy, whereas the second additionally incorporates Fisher score to account for class separability.

For feature selection, the balanced multiclass training labels were converted into binary labels, with benign flows assigned class 0 and all attack flows assigned class 1. The original multiclass labels were retained only for attack-type-wise post-analysis. CMI, Fisher score, and correlation values were computed exclusively from the training data to avoid information leakage. Continuous feature values were first converted into cluster states using K-means-based partitioning, and mutual information between the resulting cluster labels and binary class labels was used as the CMI relevance score.

BBFS was executed with fixed search-control settings across datasets and scoring variants. The search used population-based bower representation, top- and low-performing groups, Lévy-flight-based exploration, probabilistic feature inclusion based on positive marginal gain, and feature removal when deletion preserved or improved subset fitness. The scoring weights α, β, and γ were selected according to the evaluated variant, with $\gamma = 1 - (\alpha + \beta)$. In the CMI–correlation variant, $\beta = 0$, whereas in the CMI–Fisher–correlation variant, both α and β were assigned non-zero values. The exact values used for each dataset and variant are reported with the corresponding results. The Lévy-flight parameter was fixed at $\lambda_L = 1.5$.

The final BBFS reduct obtained for each dataset and scoring variant was applied unchanged to the corresponding training, validation, and test sets. This ensured dimensional consistency before HGS-driven 1D CNN optimization and prevented the use of test-set labels or statistics during feature scoring and subset selection.

3.2.6 HGS–1D CNN Search Space and Training Configuration

The HGS optimizer was used to tune the 1D CNN architecture and training hyperparameters over a mixed discrete–continuous search space. The search included convolutional filter counts, kernel sizes, pooling factor, dense-layer widths, dropout rates, ℓ_2 regularization strength, learning rate, and batch size. The candidate search space used in the experiments was as follows:

- Convolutional filters: $\texttt{filters1}, \texttt{filters2} \in \{64, 128, 256\}$, $\texttt{filters3} \in \{128, 256, 384\}$
- Kernel sizes: $\texttt{k1}, \texttt{k2}, \texttt{k3} \in \{3, 5, 7\}$

- Pooling size: `pool` $\in \{2, 3\}$
- Dense units: `dense1` $\in \{128, 256, 384\}$, `dense2` $\in \{64, 128, 256\}$
- Dropout rates: `drop1`, `drop2` $\in [0.20, 0.60]$
- ℓ_2 regularization strength: `l2` $\in [10^{-5}, 10^{-3}]$, log-uniform
- Learning rate: `lr` $\in [10^{-4}, 5 \times 10^{-3}]$, log-uniform
- Batch size: `batch` $\in \{256, 512, 1024\}$

Each HGS agent encoded these hyperparameters as a normalized vector in $[0, 1]^D$, which was decoded into a concrete 1D CNN configuration during evaluation. Candidate models were trained and validated using stratified partitions of the BBFS-reduced balanced training data to preserve the benign–attack class distribution. The validation objective used the custom F_β-score in Equation 7, with the attack class treated as the positive class. The value of β controlled the degree of recall emphasis and was set to 1.02 for Edge-IIoTset, 1.80 for ACI-IoT 2023, 1.05 for UQ-IoT, and 1.10 for WUSTL-IIoT. For each candidate model, the decision threshold was selected from a predefined threshold grid by maximizing validation F_β.

The HGS fitness followed Equation 9, combining validation F_β-score with the compactness term based on normalized trainable parameter count and MAC estimates. Equal compactness weights were used for all datasets, with $w_p = 0.5$ and $w_m = 0.5$. The dataset-specific α_h values were 0.95 for Edge-IIoTset, 0.99 for ACI-IoT 2023, 0.95 for UQ-IoT, and 0.9999 for WUSTL-IIoT. Thus, the search primarily prioritized recall-aware validation performance while using model compactness as a secondary selection criterion.

For all datasets, the HGS population size was set to 24 agents and the number of iterations to 16. The elite fraction was 0.10, and diversification was triggered after three non-improving iterations. During candidate evaluation, each 1D CNN was trained using the Adam optimizer, binary cross-entropy loss, class weighting, learning-rate decay, and early stopping based on validation loss. The HGS evaluation used 15 warmup epochs, early-stopping patience of 5, and a validation size of 0.15. After identifying the best hyperparameter configuration and validation threshold, the final 1D CNN was rebuilt and trained on the complete BBFS-reduced balanced training data. The final model was evaluated on the held-out original test set after converting multi-class labels into binary labels using the same benign-versus-attack rule. The reported outputs include accuracy, attack precision, attack recall, attack F1-score, custom F_β-score, confusion matrix, attack-type detection analysis, trainable parameter count, MAC estimate, model size, and median test-set inference time.

Although multiple experimental runs were conducted and the observed trends were consistent, the main result tables report a single representative run. This maintains one consistent prediction set for aggregate metrics, attack-type-wise analysis, false-negative examination, confusion-matrix-based comparison, and statistical testing. More extensive repeated-run reporting with mean–standard deviation summaries is left for future work.

3.2.7 Baseline Methods for Comparative Evaluation

To provide a rigorous assessment of the proposed BBFS–HGS–1D CNN pipeline, six representative IDS approaches from recent literature were re-implemented and evaluated on the same four IoT datasets used in this study: Edge-IIoTset, ACI-IoT 2023, UQ-IoT, and WUSTL-IIoT. Rather than relying only on the authors' reported results, each baseline was reproduced as closely as possible from the original methodological description and tested under the same preprocessing, flow-generation, balancing, and train–test partitioning protocols. Table 2 summarizes the selected baselines and their roles in the comparative evaluation.

Table 2: Reimplemented baseline IDS methods used for comparative evaluation.

Study and Method	Core Design	Baseline Role
Chalichalamala et al. [33] *RFE–LREC*	Recursive Feature Elimination combined with a Logistic Regression Ensemble Classifier for dimensionality reduction and ensemble-based decision fusion.	Feature-selection-based ensemble ML baseline
Rahman et al. [34] *Cognitive LR IDS*	Lightweight Logistic Regression-based IDS designed for resource-constrained IoT-enabled flying ad hoc network environments, including the reported cognitive weighting strategy.	Lightweight interpretable ML baseline
Sahu et al. [5] *FogNet*	Hybrid feature-selection chain using African Buffalo Optimization, Effective Seeker Optimization, and Weighted Gain Analysis, followed by a custom CNN.	Evolutionary–DL hybrid baseline
Alzahrani et al. [35] *CNN–LSTM*	Hybrid CNN–LSTM model in which CNN layers extract spatial patterns from flow vectors and LSTM layers capture sequential dependencies.	Fog-oriented hybrid DL baseline
Al Qathrady et al. [36] *SACNN-IDS*	Self-attention convolutional neural network using a self-attention layer to assign attention to input feature groups, followed by CNN layers for attack prediction.	Attention-based DL baseline
Kaushik et al. [1] *Chi-square–Pearson–DT*	Statistical feature-selection pipeline using Chi-square and Pearson correlation filtering, followed by a Decision Tree classifier.	Compact traditional ML baseline

By re-implementing all six approaches within a unified experimental framework, the reported comparative results reflect performance differences attributable primarily to model design rather than dataset inconsistencies or preprocessing variations.

3.3 Hardware and Software Environment

All experiments were executed on a workstation equipped with an NVIDIA RTX 3090 GPU with 24 GB VRAM, an AMD Ryzen 9 processor, and 64 GB system memory.

The DL components were implemented using TensorFlow 2.10, while scikit-learn was used for preprocessing, class balancing, and performance evaluation.

The BBFS feature-selection module and HGS hyperparameter optimization routines were implemented in Python 3.10 using NumPy, SciPy, and scikit-learn. The complete pipeline, including preprocessing, BBFS reduction, HGS-optimized 1D CNN training, and evaluation, was executed independently for each dataset under the same environment settings.

Since the experiments were conducted on a workstation, the computational analysis is limited to selected feature count, trainable parameter count, model size, MAC estimates, and test-set inference time. Device-level latency, memory usage, energy consumption, and throughput measurements are left for future deployment validation.

4 Results and Discussion

This section evaluates the proposed BBFS–HGS–1D CNN intrusion detection pipeline on four IoT/IIoT datasets: Edge-IIoTset, ACI-IoT 2023, UQ-IoT, and WUSTL-IIoT. The analysis is organized component-wise to clarify the individual effects of balancing, BBFS-based feature reduction, CMI-guided scoring, and HGS-based CNN optimization. First, feature-selection ablation compares the earlier Fisher-score–correlation BBFS formulation with the revised CMI–correlation and CMI–Fisher–correlation variants. Next, the effect of feature reduction on CNN compactness and detection performance is examined. The HGS-optimized 1D CNN is then analyzed in terms of detection metrics, convergence behavior, and computational indicators. Finally, the proposed method is compared with reimplemented baselines using aggregate binary metrics, attack-type-wise detection analysis, and statistical significance testing. All evaluations follow the same preprocessing, binary label consolidation, and hybrid balancing protocol to ensure consistency across datasets.

4.1 Dataset-wise Feature-Selection Ablation

To clarify the contributions of data balancing, BBFS-based feature reduction, and the revised CMI-guided scoring strategies, a dataset-wise ablation study was conducted before HGS-based CNN optimization. For each dataset, five configurations were evaluated: full-feature CNN training on the original imbalanced data, full-feature CNN training after SMOTE–ENN–LOF balancing, BBFS with the earlier Fisher-score–correlation formulation, BBFS with the revised CMI–correlation formulation, and BBFS with the revised CMI–Fisher–correlation formulation. This comparison separates the effect of class balancing from feature selection. Further, it quantifies the contribution of replacing or augmenting the earlier Fisher-score-based formulation with CMI-guided relevance estimation.

The generalized BBFS scoring function uses α, β, and γ to control the relative contributions of CMI relevance, Fisher-score separability, and correlation-based redundancy penalty, respectively. For the CMI–correlation variant, $\beta = 0$, whereas for the CMI–Fisher–correlation variant, all three components are used. The dataset-specific relative weights used for the CMI–Fisher–correlation variant are summarized in Table 3. These values were fixed before test-set evaluation.

Table 3: Dataset-specific BBFS scoring weights for the CMI–Fisher–correlation variant.

Dataset	α (CMI)	β (Fisher)	γ (Correlation penalty)
Edge-IIoTset	0.85	0.10	0.05
ACI-IoT 2023	0.80	0.15	0.05
UQ-IoT	0.40	0.55	0.05
WUSTL-IIoT	0.80	0.08	0.12

The results are presented separately for Edge-IIoTset, ACI-IoT 2023, UQ-IoT, and WUSTL-IIoT. For each dataset, the aggregate performance table reports the number of input features, accuracy, attack-class precision, attack-class recall, and attack-class F1-score. In addition, grouped bar plots compare attack-type-wise detection rates across the five configurations. This dataset-wise presentation is important because aggregate binary metrics may show only marginal differences on saturated datasets. In contrast, attack-type-wise analysis can reveal whether balancing and feature selection improve the recognition of minority attack categories.

4.1.1 Edge-IIoT

Table 4 presents the Edge-IIoTset ablation results, and Figure 4 shows the corresponding attack-type-wise detection rates. Although the full-feature imbalanced model achieved high aggregate performance, the attack-wise results show weaker detection for minority classes such as Malware and Recon. After SMOTE–ENN–LOF balancing, minority-class attack detection improved, and the number of missed attack samples was reduced. The BBFS variants then reduced the input dimensionality from 76 features to 22, 21, and 20 features while maintaining near-identical performance. For the CMI–Fisher–correlation variant, the dominant CMI weight ($\alpha = 0.85, \beta = 0.10, \gamma = 0.05$) indicates that nonlinear feature–label relevance was the main driver of feature selection. The CMI–correlation variant achieved the best feature-reduced F1-score, whereas the CMI–Fisher–correlation variant achieved perfect attack recall with the smallest subset.

Table 4: BBFS ablation results on Edge-IIoTset.

Configuration	Features	Accuracy	Precision	Recall	F1-score
Full features + imbalanced	76	0.999661	0.999992	0.999531	0.999762
Full features + balanced	76	0.999533	0.999346	0.999997	0.999672
BBFS (Fisher+Corr) + balanced	22	0.999532	0.999346	0.999995	0.999671
BBFS (CMI+Corr) + balanced	21	0.999536	0.999349	0.999998	0.999673
BBFS (CMI+Fisher+Corr) + balanced	20	0.999533	0.999343	1.000000	0.999672

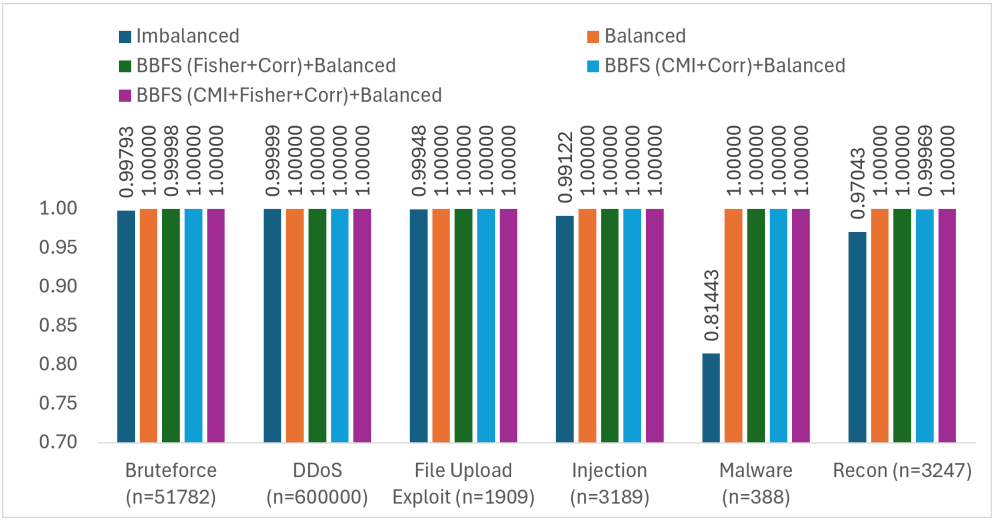


Fig. 4: Attack-type-wise detection rates for Edge-IIoTset under different balancing and BBFS feature-selection configurations.

4.1.2 ACI-IoT

Table 5 presents the ACI-IoT 2023 ablation results, and Figure 5 shows the attack-type-wise detection rates. Compared with Edge-IIoTset, ACI-IoT shows clearer differences among configurations, especially for minority attacks such as UDP Flood and Dictionary Attack. Although the imbalanced full-feature model achieved the highest aggregate F1-score, UDP Flood detection was only 0.797468. After applying SMOTE–ENN–LOF, UDP Flood detection increased to 0.981013, confirming the importance of balancing for sparse attack categories. Among the feature-reduced models, the CMI–correlation variant selected the smallest subset with 20 features and achieved the best F1-score among BBFS configurations. The CMI–Fisher–correlation variant achieved the highest attack recall among BBFS variants, but with slightly lower precision due to increased false positives.

Table 5: BBFS ablation results on ACI-IoT 2023.

Configuration	Features	Accuracy	Precision	Recall	F1-score
Full features + imbalanced	76	0.995302	0.994794	0.998814	0.996800
Full features + balanced	76	0.994165	0.994327	0.997728	0.996024
BBFS (Fisher+Corr) + balanced	29	0.992509	0.992319	0.997495	0.994900
BBFS (CMI+Corr) + balanced	20	0.993004	0.992699	0.997789	0.995237
BBFS (CMI+Fisher+Corr) + balanced	23	0.991753	0.990298	0.998526	0.994395

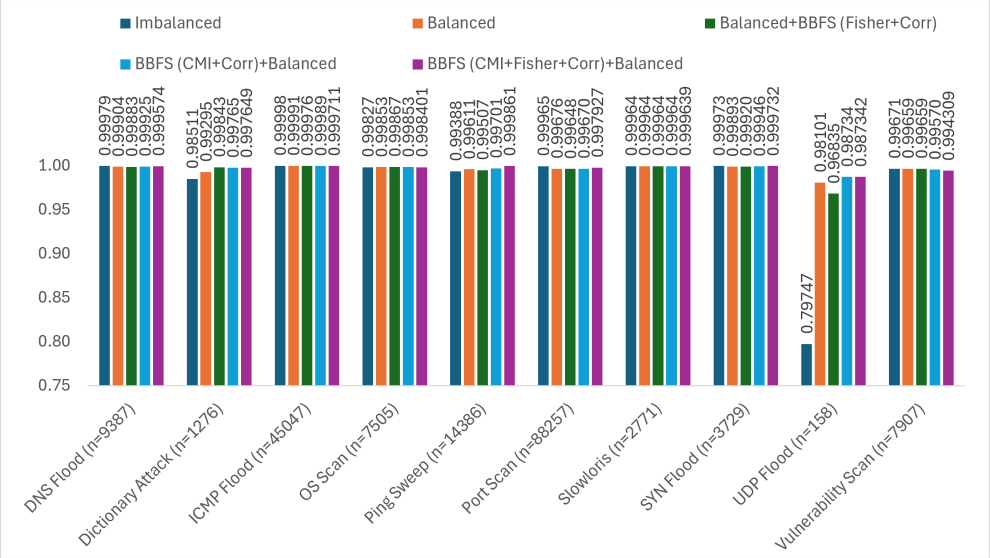


Fig. 5: Attack-type-wise detection rates for ACI-IoT 2023 under different balancing and BBFS feature-selection configurations.

4.1.3 UQ-IoT

Table 6 presents the UQ-IoT ablation results, and Figure 6 shows the corresponding attack-type-wise detection rates. The imbalanced full-feature model achieved the highest aggregate metrics. Still, the attack-wise analysis reveals poor detection of minority classes such as ARP Spoofing, Host Discovery, and Telnet Brute Force. After SMOTE–ENN–LOF balancing, ARP Spoofing increased from 0.441860 to 0.608527, Host Discovery from 0.483957 to 0.660428, and Telnet BruteForce from 0.446078 to 0.605392. The BBFS variants reduced the feature count substantially while preserving high attack recall. For UQ-IoT, the CMI–Fisher–correlation setting assigned greater weight to Fisher score than to CMI ($\alpha = 0.40, \beta = 0.55, \gamma = 0.05$), suggesting that class separability was more influential than nonlinear relevance alone. This variant achieved the highest F1 score among the feature-reduced configurations.

Table 6: BBFS ablation results on UQ-IoT.

Configuration	Features	Accuracy	Precision	Recall	F1-score
Full features + imbalanced	76	0.997665	0.999526	0.997295	0.998409
Full features + balanced	76	0.989608	0.988227	0.997742	0.992962
BBFS (Fisher+Corr) + balanced	16	0.986853	0.984156	0.998175	0.991116
BBFS (CMI+Corr) + balanced	20	0.985086	0.981902	0.998097	0.989933
BBFS (CMI+Fisher+Corr) + balanced	18	0.987141	0.984558	0.998154	0.991309

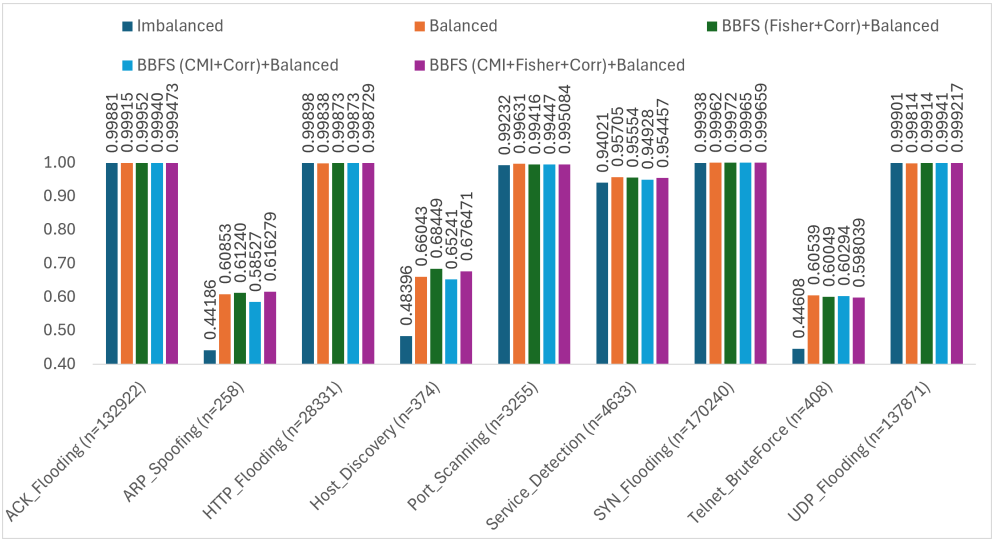


Fig. 6: Attack-type-wise detection rates for UQ-IoT under different balancing and BBFS feature-selection configurations.

4.1.4 WUSTL-IIoT

Table 7 presents the WUSTL-IIoT ablation results, and Figure 7 shows the corresponding attack-type-wise detection rates. The imbalanced full-feature model achieved the highest aggregate accuracy and F1-score, but the minority Backdoor class showed a lower detection rate of 0.857143. After SMOTE–ENN–LOF balancing, Reconn, CommInj, and Backdoor detection reached 1.000000, while DoS detection remained nearly unchanged, showing that balancing mainly improved low-frequency attack recognition. The BBFS variants preserved near-saturated detection while reducing the feature count from 39 to 13 features for Fisher–correlation and to 10 features for both CMI-guided variants. For WUSTL-IIoT, the CMI–Fisher–correlation setting ($\alpha = 0.80, \beta = 0.08, \gamma = 0.12$) reflects the need to retain nonlinear relevance while controlling redundancy in the smaller Argus-derived feature space.

Table 7: BBFS ablation results on WUSTL-IIoT.

Configuration	Features	Accuracy	Precision	Recall	F1-score
Full features + imbalanced	39	0.999958	0.999943	0.999483	0.999713
Full features + balanced	39	0.999221	0.989537	0.999885	0.994684
BBFS (Fisher+Corr) + balanced	13	0.999213	0.989480	0.999828	0.994627
BBFS (CMI+Corr) + balanced	10	0.999175	0.989029	0.999770	0.994371
BBFS (CMI+Fisher+Corr) + balanced	10	0.999092	0.987851	0.999828	0.993803

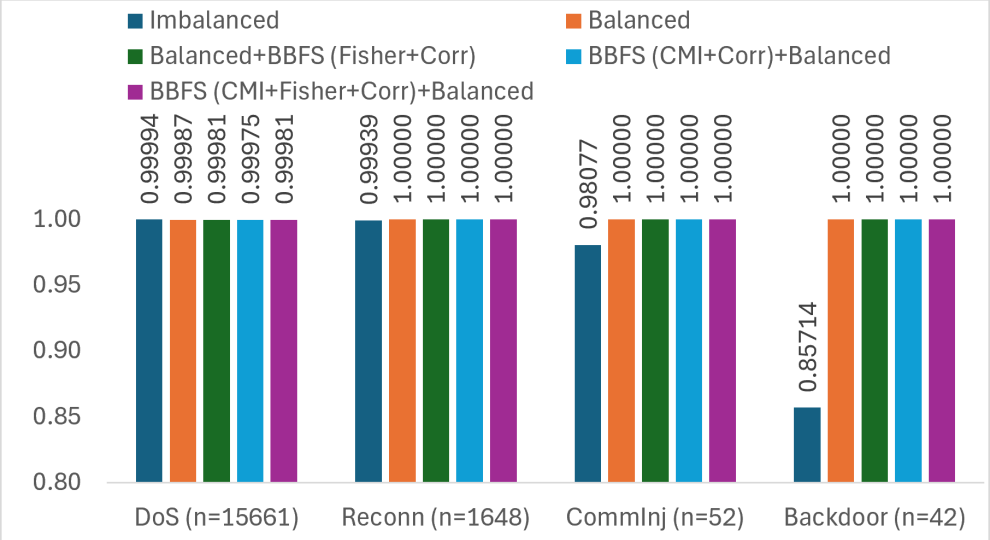


Fig. 7: Attack-type-wise detection rates for WUSTL-IIoT under different balancing and BBFS feature-selection configurations.

Overall, the ablation results show that SMOTE–ENN–LOF primarily improves the recognition of minority attack categories. At the same time, BBFS substantially reduces feature dimensionality with only marginal changes in aggregate binary performance. The CMI-guided variants generally produce more compact reducts than the earlier Fisher–correlation formulation, confirming the value of nonlinear feature–label relevance in the proposed filter-based BBFS setting.

4.2 Performance of the HGS-Optimized 1D CNN

After the BBFS-based feature-selection ablation, the CMI–Fisher–correlation BBFS reduct was used as the final input representation for HGS-based 1D CNN optimization across all four datasets. The HGS stage optimized the CNN architecture and decision threshold using the balanced training data and validation partition. Final evaluation was performed on the original untouched test set after converting multiclass labels into binary benign-versus-attack labels. Thus, the reported test results reflect binary intrusion detection performance on the original test distribution.

The HGS fitness function jointly considered recall-aware detection performance and architectural compactness. Detection performance was measured using the attack-class F_β-score, while compactness was represented using trainable parameter count and MAC estimates. These values are interpreted as indicators of architecture-level complexity. Since all experiments were conducted on a workstation, device-level latency, memory consumption, energy usage, and throughput are not claimed and are left for future deployment validation.

4.2.1 HGS-Selected Architectures and Complexity

Table 8 summarizes the final HGS-selected configuration for each dataset in terms of selected feature count, optimized decision threshold, F_β setting, trainable parameters, and MAC estimates. All models used the BBFS (CMI–Fisher–Corr) selected features. The selected feature count ranged from 10 for WUSTL-IIoT to 23 for ACI-IoT 2023, showing that BBFS produced compact input representations before CNN optimization.

Table 8: HGS-selected 1D CNN configuration and complexity using BBFS (CMI–Fisher–Corr) features.

Dataset	Features	Thr.	β	Params	MACs
Edge-IIoTset	20 / 76	0.05	1.02	63,937	188,224
ACI-IoT 2023	23 / 76	0.10	1.80	105,281	364,992
UQ-IoT	18 / 76	0.17	1.05	96,705	294,336
WUSTL-IIoT	10 / 38	0.72	1.10	105,281	235,520

The Edge-IIoTset model produced the smallest CNN, with 63,937 trainable parameters and 188,224 MACs. WUSTL-IIoT used the smallest feature subset, with only 10 selected features, although its selected CNN retained 105,281 parameters because of the chosen convolutional and dense-layer configuration. ACI-IoT 2023 required the highest MAC count, consistent with its larger selected feature subset and more diverse attack composition, while UQ-IoT produced an intermediate architecture.

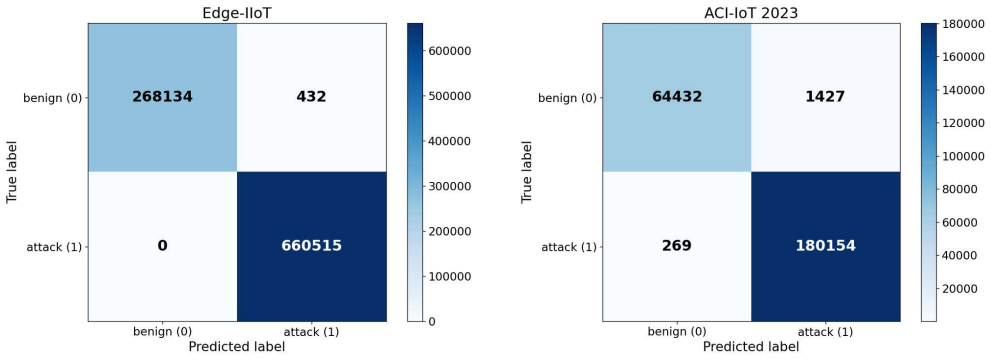
4.2.2 Detection Performance on Untouched Test Sets

Table 9 reports the final binary detection performance of the HGS-optimized 1D CNN models on the untouched test sets. The original multiclass test labels were converted into binary benign-versus-attack labels, with attack traffic treated as the positive class. The corresponding confusion matrices are shown in Figure 8.

Table 9: Final test performance of HGS-optimized 1D CNN using BBFS (CMI–Fisher–Corr) features.

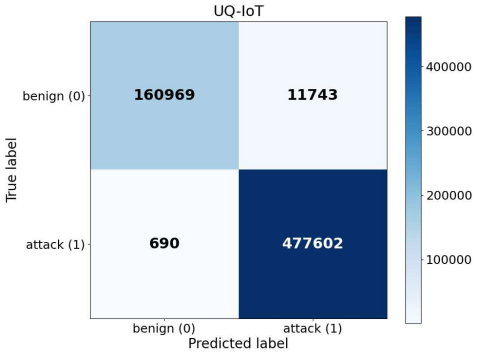
Dataset	Accuracy	Precision	Recall	F1-score	F_β
Edge-IIoTset	0.999535	0.999346	1.000000	0.999673	0.999680
ACI-IoT 2023	0.993114	0.992141	0.998509	0.995315	0.997000
UQ-IoT	0.980902	0.976003	0.998557	0.987151	0.987701
WUSTL-IIoT	0.999163	0.989083	0.999540	0.994284	0.994781

The Edge-IIoTset model achieved the strongest attack-recall behavior, with no attack samples misclassified as benign, as shown in Figure 8a. This produced an attack

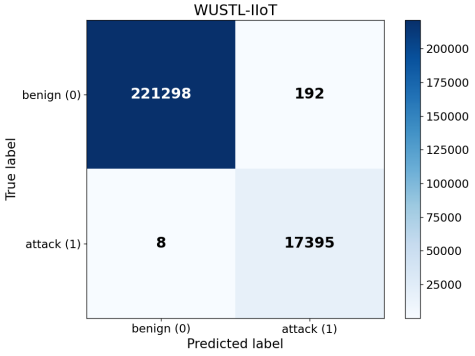


(a) Edge-IIoTset

(b) ACI-IoT 2023



(c) UQ-IoT



(d) WUSTL-IIoT

Fig. 8: Confusion matrices of the HGS-optimized 1D CNN using BBFS (CMI–Fisher–Corr) features on the untouched test sets.

recall of 1.000000 and an F1-score of 0.999673 using only 20 selected features. ACI-IoT 2023 also achieved high recall, correctly detecting 180,154 out of 180,423 attack samples, resulting in an attack recall of 0.998509 and an F1-score of 0.995315. The higher β value used for ACI-IoT 2023 reflects the stronger recall emphasis during HGS optimization.

UQ-IoT was the most challenging dataset. Although the model maintained a high attack recall of 0.998557, attack precision decreased to 0.976003 because 11,743 benign flows were classified as attacks, as shown in Figure 8c. This indicates that the HGS-tuned model prioritized attack coverage, while the benign–attack boundary in UQ-IoT remained more difficult than in the other datasets. For WUSTL-IIoT, the model missed only eight attack samples, producing an attack recall of 0.999540 and an F1-score of 0.994284.

Overall, the HGS-optimized 1D CNN models maintained high attack recall across all datasets while using compact BBFS-selected feature representations. These results

indicate that the proposed BBFS–HGS combination can reduce input dimensionality and generate compact CNN architectures without substantially degrading binary attack-detection performance. However, the reported complexity values should be interpreted as architecture-level indicators; deployment-specific claims require further validation on physical IoT or edge hardware.

4.2.3 Attack-Type-Wise Detection Analysis

Although the final HGS-optimized 1D CNN models were trained as binary classifiers, the original multiclass test labels were retained for post-hoc attack-type-wise analysis. This analysis was performed only after binary prediction and was not used during training, balancing, feature selection, threshold selection, or HGS optimization. For each attack category, the detection rate is calculated as the proportion of samples from that attack class correctly classified as an attack.

The Edge-IIoTset model detected all evaluated attack categories, with no missed attack samples. This confirms that the aggregate recall of 1.000000 was not caused only by the dominant DDoS class, but was also maintained for lower-frequency categories such as Malware and File upload exploit.

For ACI-IoT 2023, all attack categories achieved high detection rates. UDP Flood was detected perfectly, while DNS Flood, ICMP Flood, Slowloris, and SYN Flood were detected at rates close to 1.000000. The lowest detection rate was observed for Vulnerability Scan, with 32 missed samples out of 7,907. Port Scan contributed the largest number of missed attack samples because it had the largest test support among the ACI-IoT 2023 attack categories.

UQ-IoT was the most challenging dataset in the attack-type-wise analysis. Dominant flooding attacks such as ACK Flooding, HTTP Flooding, SYN Flooding, and UDP Flooding achieved near-perfect detection rates. However, minority and boundary-overlapping attacks such as ARP Spoofing, Host Discovery, and Telnet BruteForce showed lower detection rates of 0.686047, 0.721925, and 0.642157, respectively. This explains why UQ-IoT achieved high aggregate attack recall while still showing weaknesses for specific minority attack types.

For WUSTL-IIoT, DoS, Reconn, and CommInj were detected with high or perfect detection rates. The Backdoor class had one missed sample out of 42, producing a detection rate of 0.976190. Since CommInj and Backdoor contain very few test samples, their detection rates should be interpreted cautiously. Nevertheless, the results indicate strong attack coverage across the available WUSTL-IIoT attack categories.

Overall, the attack-type-wise analysis provides a more detailed interpretation of the binary detection results. Edge-IIoTset, ACI-IoT 2023, and WUSTL-IIoT showed consistently strong detection across attack categories, whereas UQ-IoT revealed that some minority attack types remain difficult even when aggregate binary recall is high. These observations confirm the importance of reporting attack-type-wise post-analysis in addition to standard binary metrics.

4.2.4 HGS Convergence Analysis

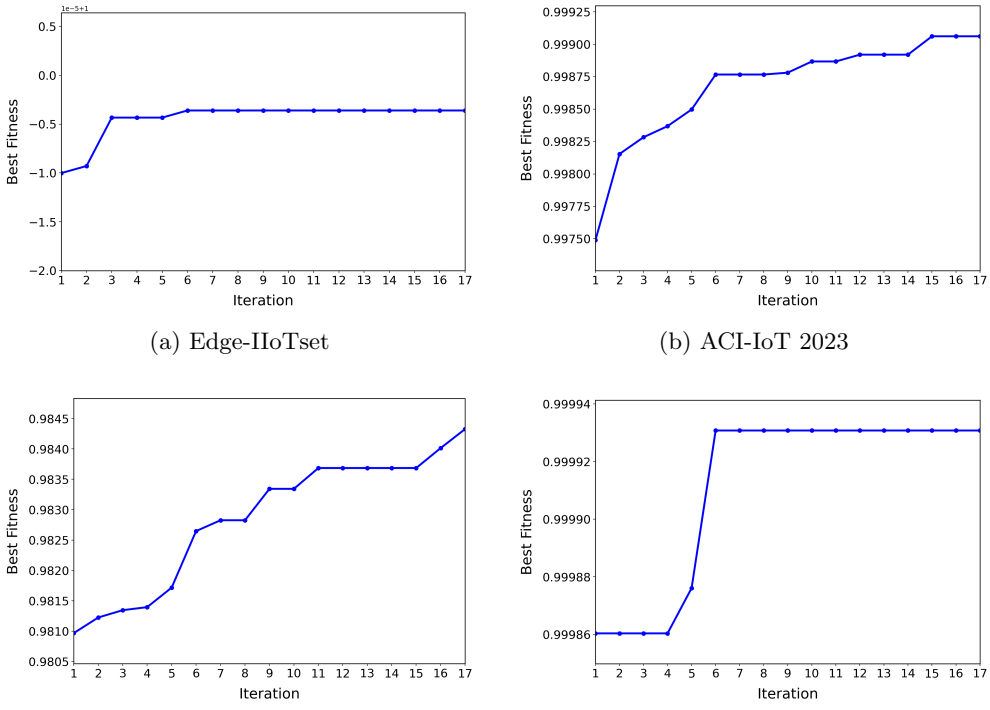
To examine the optimization behavior of HGS, the best-so-far fitness value was recorded during the initial population evaluation and the subsequent 16 HGS update

Table 10: Attack-type-wise detection analysis of the final HGS-optimized 1D CNN models.

Dataset	Attack type	Test samples	Detected	Missed	Detection rate
Edge-IIoTset	Bruteforce	51,782	51,782	0	1.000000
	DDoS	600,000	600,000	0	1.000000
	File upload exploit	1,909	1,909	0	1.000000
	Injection	3,189	3,189	0	1.000000
	Malware	388	388	0	1.000000
	Recon	3,247	3,247	0	1.000000
ACI-IoT 2023	DNS Flood	9,387	9,384	3	0.999680
	Dictionary Attack	1,276	1,274	2	0.998433
	ICMP Flood	45,047	45,043	4	0.999911
	OS Scan	7,505	7,495	10	0.998668
	Ping Sweep	14,386	14,346	40	0.997220
	Port Scan	88,257	88,083	174	0.998028
	Slowloris	2,771	2,770	1	0.999639
	SYN Flood	3,729	3,726	3	0.999195
	UDP Flood	158	158	0	1.000000
	Vulnerability Scan	7,907	7,875	32	0.995953
UQ-IoT	ACK Flooding	132,922	132,865	57	0.999571
	ARP Spoofing	258	177	81	0.686047
	HTTP Flooding	28,331	28,295	36	0.998729
	Host Discovery	374	270	104	0.721925
	Port Scanning	3,255	3,248	7	0.997849
	Service Detection	4,633	4,497	136	0.970645
	SYN Flooding	170,240	170,197	43	0.999747
	Telnet BruteForce	408	262	146	0.642157
	UDP Flooding	137,871	137,791	80	0.999420
WUSTL-IIoT	DoS	15,661	15,654	7	0.999553
	Reconn	1,648	1,648	0	1.000000
	CommInj	52	52	0	1.000000
	Backdoor	42	41	1	0.976190

iterations. The convergence curves are shown in Figure 9. Since the HGS fitness combines recall-aware validation F_β-score with model compactness, the convergence trend reflects how the optimizer progressively identifies CNN configurations that maintain high attack-detection performance while also considering trainable parameter count and MAC estimates.

The convergence curves show that HGS identified high-quality CNN configurations within the selected search budget. Edge-IIoTset and WUSTL-IIoT reached high fitness values early, indicating that the optimizer quickly found configurations that achieved near-saturated validation performance while satisfying the compactness-aware fitness criterion. This is consistent with the final test results, where both datasets achieved very high attack recall using compact BBFS-selected feature representations.



(a) Edge-IIoTset

(b) ACI-IoT 2023

(c) UQ-IoT

(d) WUSTL-IIoT

Fig. 9: Convergence behavior of HGS during 1D CNN hyperparameter optimization.

For ACI-IoT 2023, the convergence curve reflects the stronger recall-oriented setting used during optimization. Since this dataset used $\beta = 1.80$, the HGS search emphasized attack recall more strongly than in the other datasets. UQ-IoT showed more gradual improvement, consistent with its lower final precision and the attack-type-wise analysis, where some minority attack categories remained harder to detect.

Overall, the convergence analysis indicates that HGS obtained effective CNN configurations within a compact search budget. However, the curves represent the optimization behavior of the recorded experimental runs rather than a complete statistical stability study. Therefore, broader stability claims would require multiple independent runs with different random seeds and additional sensitivity analysis.

4.3 Model Size and Test-Time Efficiency

To complement the theoretical BBFS complexity analysis and the parameter/MAC-based CNN compactness indicators, Table 11 reports the model size and median full-test-set inference time for the baseline balanced CNN, BBFS-reduced CNN, and final BBFS–HGS-optimized CNN. The reported time corresponds to the median inference

time over the complete test set under the same workstation environment used for all experiments.

Table 11: Model size and median full-test-set inference time for the baseline balanced CNN, BBFS-reduced CNN, and HGS-optimized CNN pipelines.

Dataset	Pipeline	Features	Model Size (MB)	Median Full-Test-Set Inference Time (s)
Edge-IIoTset	Baseline Balanced	72	10.161	1.511
	BBFS-reduced	20	4.656	0.591
	BBFS–HGS-optimized	20	0.848	0.425
ACI-IoT 2023	Baseline Balanced	76	10.161	0.439
	BBFS-reduced	23	4.656	0.173
	BBFS–HGS-optimized	23	1.150	0.143
UQ-IoT	Baseline Balanced	76	10.161	1.061
	BBFS-reduced	18	4.656	0.392
	BBFS–HGS-optimized	18	1.438	0.372
WUSTL-IIoT	Baseline Balanced	38	6.229	0.263
	BBFS-reduced	10	3.869	0.158
	BBFS–HGS-optimized	10	1.344	0.117

The results show that BBFS-based feature reduction reduced both model size and median full-test-set inference time compared with the baseline balanced CNN. The additional HGS optimization further reduced model size and runtime by selecting compact CNN configurations while preserving recall-oriented detection behavior.

Compared with the baseline balanced CNN, the final BBFS–HGS-optimized CNN reduced model size by approximately 91.65% on Edge-IIoTset, 88.68% on ACI-IoT 2023, 85.85% on UQ-IoT, and 78.42% on WUSTL-IIoT. The corresponding median full-test-set inference time reductions were approximately 71.87%, 67.43%, 64.94%, and 55.51%, respectively. These results support the computational compactness of the proposed pipeline. However, because the timings were measured on a workstation, they should be interpreted as controlled implementation-level efficiency indicators rather than real edge-device latency, memory, energy, or throughput measurements.

4.4 Comparison with Benchmark Pipelines

To further evaluate the proposed Balanced + BBFS + HGS–1D CNN pipeline, six recent IDS pipelines were reimplemented and compared on the same test sets. The evaluated benchmarks include Rahman et al. [34], Chalichalamala et al. [33], Alzahrani et al. [35], Sahu et al. [5], Kaushik et al. [1], and Al-Qathrady et al. [36]. For Kaushik et al., the best-performing classifier variant was selected for each dataset based on the highest F1-score. All methods were evaluated under the same binary benign-versus-attack test setting, while the original multiclass attack labels were retained only for post-hoc attack-type-wise detection analysis. Tables 12–15 report aggregate binary

performance, and Figures 10–13 compare the proposed model with the strongest recall-oriented benchmark for each dataset.

4.4.1 Edge-IIoTset Comparison

Table 12 presents the benchmark comparison on Edge-IIoTset. The proposed method achieved the highest attack recall of 1.000000, using only 20 selected features. Although Sahu et al. achieved a marginally higher F1-score, the proposed method produced zero missed attack samples in the binary test setting.

Table 12: Comparison with benchmark pipelines on Edge-IIoTset.

Method	Features	Accuracy	Precision	Recall	F1-score
Rahman et al.	76	0.959362	0.997854	0.944870	0.970640
Chalichalamala et al.	10	0.789653	0.999888	0.704205	0.826394
Alzahrani et al.	76	0.999648	0.999906	0.999599	0.999752
Sahu et al.	18	0.999537	0.999383	0.999967	0.999675
Al-Qathrady et al.	72	0.999638	0.999909	0.999582	0.999746
Kaushik et al.	21	0.999146	0.999165	0.999635	0.999400
Proposed	20	0.999535	0.999346	1.000000	0.999673

Figure 10 compares the proposed method with Sahu et al., the highest-recall benchmark on Edge-IIoTset. The proposed method achieved a detection rate of 1.000000 for all evaluated attack types, including Bruteforce, DDoS, File Upload Exploit, Injection, Malware, and Recon. This confirms that the perfect aggregate recall was consistently maintained across both high- and low-support attack categories.

4.4.2 ACI-IoT 2023 Comparison

Table 13 reports the benchmark comparison on ACI-IoT 2023. The proposed method achieved the best accuracy, recall, and F1-score among the compared methods, while Al-Qathrady et al. achieved slightly higher precision. The result indicates that the proposed model improved attack coverage while maintaining competitive precision.

Table 13: Comparison with benchmark pipelines on ACI-IoT 2023.

Method	Features	Accuracy	Precision	Recall	F1-score
Rahman et al.	76	0.912657	0.965663	0.913248	0.938724
Chalichalamala et al.	10	0.976673	0.984640	0.983500	0.984070
Alzahrani et al.	76	0.985988	0.986181	0.994812	0.990478
Sahu et al.	30	0.986958	0.986312	0.996020	0.991142
Al-Qathrady et al.	70	0.992756	0.992186	0.997971	0.995070
Kaushik et al.	19	0.967294	0.990630	0.964478	0.977379
Proposed	23	0.993114	0.992141	0.998509	0.995315

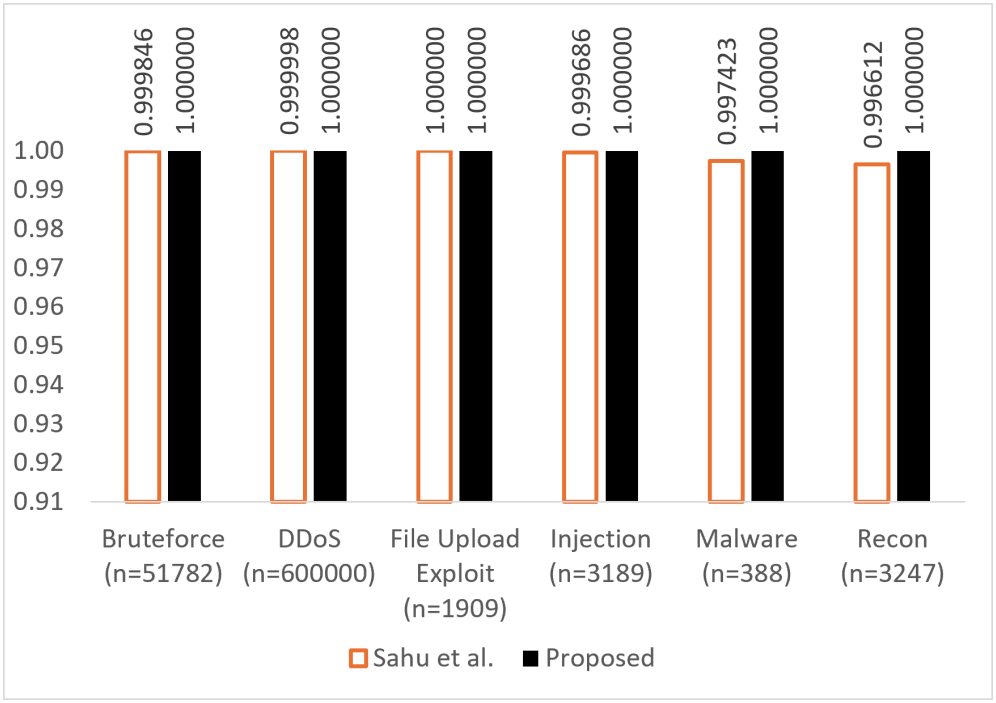


Fig. 10: Attack-type-wise detection-rate comparison on Edge-IIoTset against the highest-recall benchmark.

Figure 11 compares the proposed method with Al-Qathrady et al., the highest-recall benchmark on ACI-IoT 2023. The proposed model maintained strong detection across all attack categories and achieved 1.000000 detection for UDP Flood. The most visible gains were observed for difficult categories such as Dictionary Attack and UDP Flood, where the benchmark detection rates were lower.

4.4.3 UQ-IoT Comparison

Table 14 presents the benchmark comparison on UQ-IoT. Al-Qathrady et al. achieved the highest aggregate F1-score, whereas the proposed method achieved the highest recall. This reflects the recall-oriented design of the proposed pipeline, which prioritizes reducing missed attacks even at the cost of additional false positives.

Figure 12 compares the proposed method with Kaushik et al., the highest-recall benchmark on UQ-IoT. The proposed method retained near-perfect detection for major flooding categories and improved detection for difficult low-support classes such as ARP Spoofing, Host Discovery, and Telnet BruteForce. This shows that the higher aggregate recall of the proposed method is supported by improved coverage of several minority attack categories.

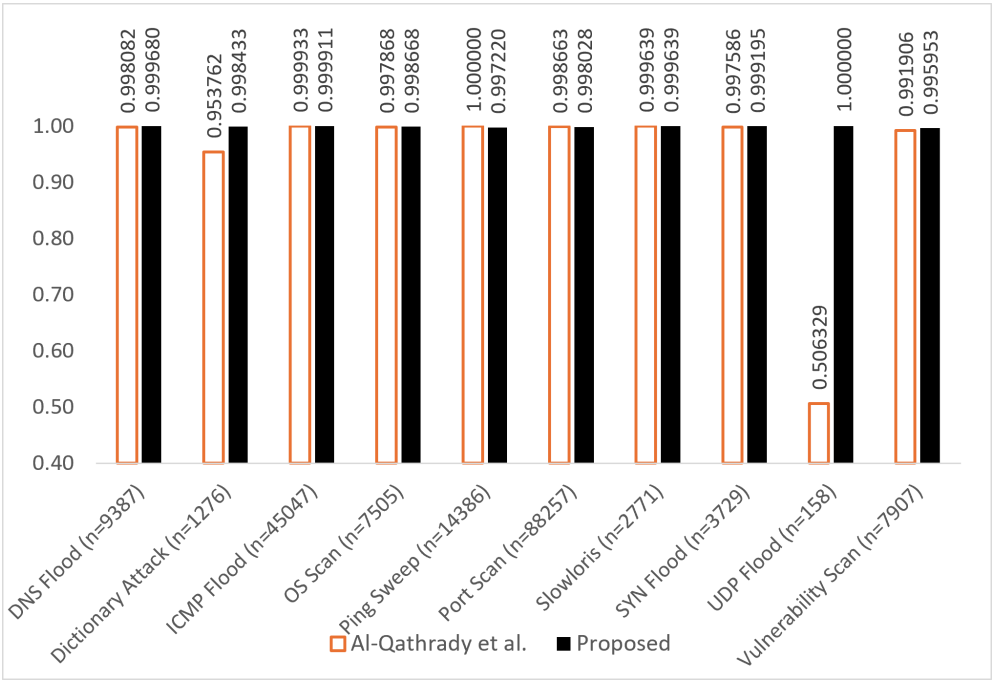


Fig. 11: Attack-type-wise detection-rate comparison on ACI-IoT 2023 against the highest-recall benchmark.

Table 14: Comparison with benchmark pipelines on UQ-IoT.

Method	Features	Accuracy	Precision	Recall	F1-score
Rahman et al.	76	0.952695	0.983648	0.951429	0.967270
Chalichalamala et al.	10	0.989346	0.998960	0.986525	0.992704
Alzahrani et al.	76	0.995528	0.999047	0.994863	0.996950
Sahu et al.	20	0.994845	0.997895	0.995083	0.996487
Al-Qathrady et al.	65	0.995986	0.999249	0.995285	0.997263
Kaushik et al.	18	0.995883	0.997762	0.996632	0.997197
Proposed	18	0.980902	0.976003	0.998557	0.987151

4.4.4 WUSTL-IIoT Comparison

Table 15 shows the benchmark comparison on WUSTL-IIoT. Al-Qathrady et al. achieved the highest accuracy, precision, and F1-score, while the proposed model achieved the highest recall using only 10 selected features. This result highlights the recall-oriented behavior of the proposed model under a compact feature representation.

Figure 13 compares the proposed method with Al-Qathrady et al., the highest-recall benchmark on WUSTL-IIoT. The proposed model achieved perfect detection for Reconn and CommInj and detected 41 out of 42 Backdoor samples. Although

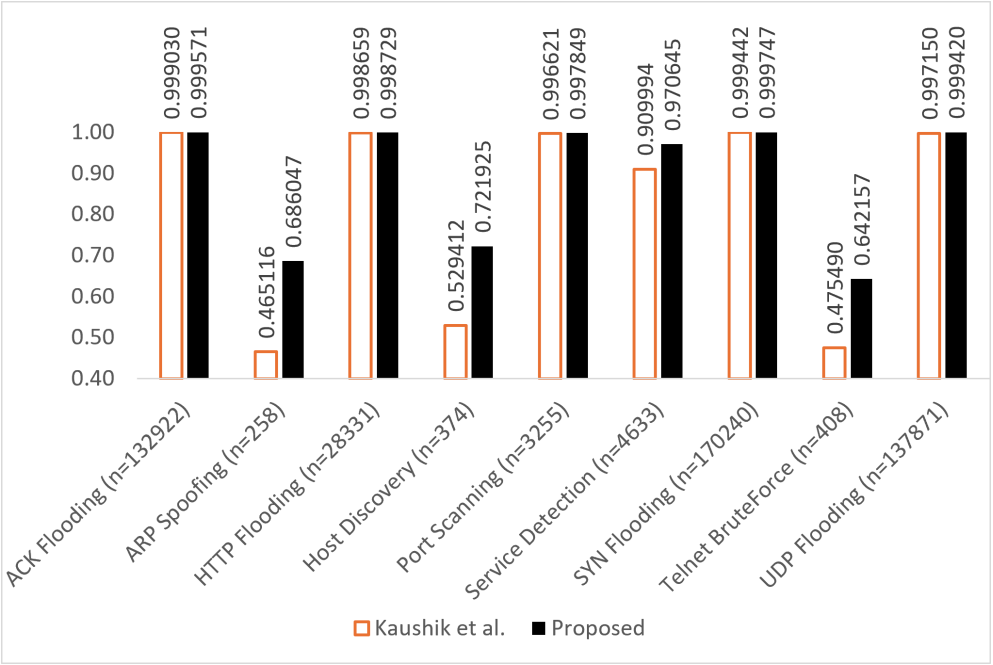


Fig. 12: Attack-type-wise detection-rate comparison on UQ-IoT against the highest-recall benchmark.

Table 15: Comparison with benchmark pipelines on WUSTL-IIoT.

Method	Features	Accuracy	Precision	Recall	F1-score
Rahman et al.	38	0.982377	0.822299	0.967075	0.888830
Chalichalamala et al.	10	0.991695	0.906512	0.987876	0.945447
Alzahrani et al.	38	0.999837	0.999482	0.998276	0.998879
Sahu et al.	6	0.982553	0.981588	0.775039	0.866170
Al-Qathrady et al.	38	0.999895	0.999942	0.998621	0.999281
Kaushik et al.	16	0.999623	0.997815	0.997012	0.997413
Proposed	10	0.999163	0.989083	0.999540	0.994284

Backdoor remained difficult due to its very small test support, the proposed method clearly improved its detection rate over the benchmark. Since Backdoor and CommInj contain very few test samples, these attack-wise rates should be interpreted cautiously.

Overall, the proposed Balanced + BBFS + HGS–1D CNN pipeline does not uniformly dominate all benchmark methods across every aggregate metric. However, it consistently provides a recall-oriented and feature-reduced detection pipeline, achieving the highest attack recall on all four datasets, including perfect recall on Edge-IIoTset. The attack-type-wise analysis further confirms that the proposed method improves or maintains detection for several difficult and low-support attack categories,

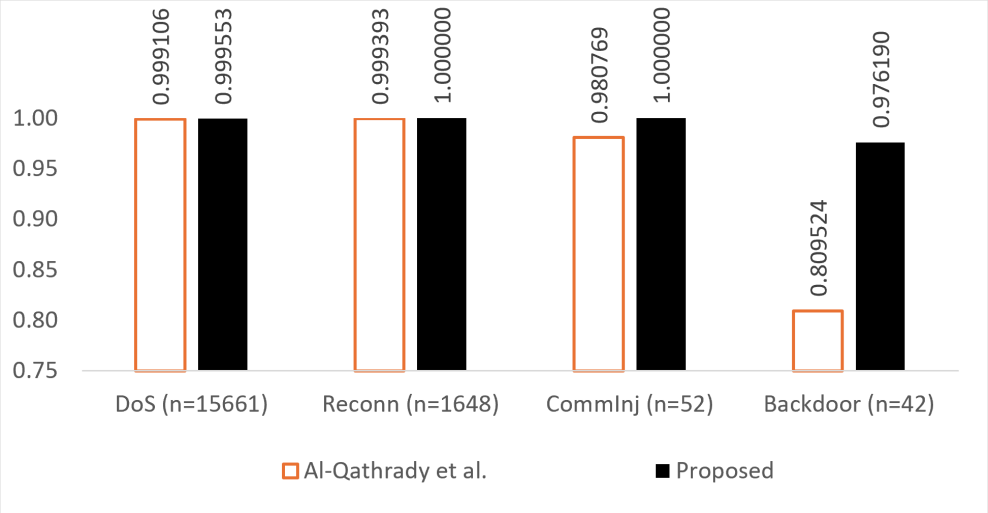


Fig. 13: Attack-type-wise detection-rate comparison on WUSTL-IIoT against the highest-recall benchmark.

making it suitable for IDS scenarios where missed attacks are more critical than a moderate increase in false positives.

4.4.5 Statistical Significance Analysis

To further validate the recall-oriented behavior of the proposed IDS, McNemar's test was conducted on the true attack records. For each dataset, the proposed model was compared with the strongest reimplemented benchmark in terms of attack recall, because the purpose of this analysis is to assess whether the reduction in missed attack samples is statistically meaningful. In this analysis, b denotes the number of attack samples correctly detected by the proposed model but missed by the benchmark, whereas c denotes the number of attack samples missed by the proposed model but correctly detected by the benchmark.

Table 16: McNemar statistical significance analysis between the proposed model and the strongest recall-oriented benchmark pipeline.

Dataset	Comparator	Attack Samples	b	c	p-value
Edge-IIoTset	Sahu et al.	660,515	22	0	4.79×10^{-7}
ACI-IoT 2023	Al-Qathrady et al.	180,423	252	206	3.55×10^{-2}
UQ-IoT	Kaushik et al.	478,292	1,245	93	2.50×10^{-217}
WUSTL-IIoT	Al-Qathrady et al.	17,403	25	5	5.22×10^{-4}

The results in Table 16 show that the proposed model provides a statistically significant recall advantage over the strongest recall-oriented benchmark on all four

datasets at the 0.05 significance level. On Edge-IIoTset, the proposed model correctly detected 22 attack flows that were missed by Sahu et al., while the reverse case did not occur, resulting in $p = 4.79 \times 10^{-7}$. On ACI-IoT 2023, the proposed model detected 252 attack samples missed by Al-Qathrady et al., compared with 206 reverse cases, yielding $p = 3.55 \times 10^{-2}$. Similarly, on UQ-IoT, the proposed model detected 1,245 attack samples missed by Kaushik et al., while only 93 attack samples were detected by the benchmark and missed by the proposed model. This produced a highly significant $p = 2.50 \times 10^{-217}$, confirming a strong reduction in missed attacks.

For WUSTL-IIoT, the comparison with Al-Qathrady et al. also indicates a statistically significant recall-oriented improvement. However, the absolute number of discordant attack samples is smaller than in the larger datasets. The proposed model correctly detected 25 attack samples missed by the benchmark, whereas only 5 reverse cases were observed. This produced $p = 5.22 \times 10^{-4}$. Since both models already detected most attack flows in this highly saturated test setting, the practical difference is numerically small; however, the proposed model achieved this recall-oriented behavior using only 10 selected features, compared with the full 38-feature representation used by the SACNN benchmark.

These statistical findings are consistent with the attack-type-wise results discussed in the previous subsection. The proposed pipeline is particularly effective in recall-sensitive IDS settings, where missed attack flows are more critical than a moderate increase in false alarms. This behavior is most evident in UQ-IoT, where the proposed model improved the detection of difficult minority categories such as ARP Spoofing, Host Discovery, and Telnet BruteForce while maintaining near-perfect detection for high-volume flooding attacks. Overall, the statistical analysis supports the claim that the proposed Balanced + BBFS + HGS–1D CNN provides a compact and recall-oriented IDS pipeline with strong attack-detection consistency across diverse IoT datasets.

Conclusion and Future Work

This study presented a compact and modular IoT intrusion detection pipeline that integrates SMOTE–ENN–LOF balancing, BBFS-based feature selection, and HGS-driven 1D CNN optimization. The proposed Balanced + BBFS + HGS–1D CNN pipeline was evaluated on ACI-IoT 2023, UQ-IoT 2021, Edge-IIoTset, and WUSTL-IIoT. The results showed that the proposed method achieved strong binary attack-detection performance using compact feature subsets: 20 features on Edge-IIoTset, 23 on ACI-IoT 2023, 18 on UQ-IoT, and 10 on WUSTL-IIoT. The model achieved perfect attack recall on Edge-IIoTset, high recall on ACI-IoT 2023 and WUSTL-IIoT, and recall-oriented detection on UQ-IoT, where improved attack coverage was obtained at the cost of additional false positives. The model-size and test-time analysis further showed that BBFS reduction and HGS optimization substantially reduced CNN size and full-test-set inference time under the controlled workstation environment.

The benchmark comparison showed that the proposed method does not uniformly outperform all competing pipelines across every aggregate metric. Some benchmark

methods achieved higher precision or F1-score on specific datasets. However, the proposed pipeline consistently provided a recall-oriented and feature-reduced IDS design, achieving the highest attack recall across the evaluated datasets. The attack-type-wise analysis confirmed that aggregate binary metrics alone are insufficient, as some methods with strong overall scores still showed weaker detection of minority attack categories. The McNemar analysis further supported the recall-oriented advantage of the proposed model over the strongest recall-focused benchmarks. Overall, the results indicate that the proposed integration of hybrid balancing, CMI-guided BBFS feature reduction, and HGS-based CNN optimization provides a compact and effective IDS pipeline for diverse IoT/IIoT traffic settings.

Future work will extend the current binary formulation to multiclass and hierarchical attack classification for finer-grained intrusion analysis. The BBFS and HGS components can also be explored in federated or privacy-preserving IDS settings. In addition, real deployment studies on IoT and edge hardware are needed to measure latency, throughput, memory usage, and energy consumption under practical operating conditions. Model-compression techniques such as quantization-aware training, structured pruning, and knowledge distillation may further reduce the computational footprint while preserving strong detection performance.

Declarations

Data Availability

All datasets used in this study are publicly available and can be accessed through their respective repositories. The ACI-IoT 2023 dataset is hosted on IEEE Dataport and can be obtained at doi:10.21227/qacj-3x32. The University of Queensland provides the UQ-IoT-IDS 2021 dataset and is accessible at doi:10.48610/17b44bb. The Edge-IIoTset dataset is available on IEEE Dataport at doi:10.21227/mbc1-1h68. The WUSTL-IIoT 2021 dataset is publicly distributed by Washington University in St. Louis and can be accessed at http://www.cse.wustl.edu/~jain/iiot2/index.html. The processed datasets generated in this study, including the train–test splits and balanced NumPy files used for experimentation, are publicly available on Zenodo at doi:10.5281/zenodo.21158877. The implementation code and compact 1D CNN training pipelines are available in the GitHub repository at https://github.com/mskrcnis/Lightweight-BBFS-HGS-IoT-IDS, supporting reproducibility of the proposed pipeline.

Funding

No funding was received to support the preparation of this manuscript.

Competing Interests

The authors declare that they have no relevant financial or non-financial interests to disclose.

References

[1] Kaushik, S., Bhardwaj, A., Almogren, A., bharany, S., Altameem, A., Rehman, A.U., Hussen, S., Hamam, H.: Robust machine learning based intrusion detection system using simple statistical techniques in feature selection. Scientific Reports **15**(1), 3970 (2025) https://doi.org/10.1038/s41598-025-88286-9

[2] Ashraf, J., Raza, G.M., Kim, B.-S., Wahid, A., Kim, H.-Y.: Making a real-time iot network intrusion-detection system (inids) using a realistic bot–iot dataset with multiple machine-learning classifiers. Applied Sciences **15**(4) (2025) https://doi.org/10.3390/app15042043

[3] Akif, O.Z., Ali, S.M., Sabih, A.F., Sadiq, A.T., Subramaniam, S.K.: Intrusion detection system for iot based on modified random forest algorithm. Iraqi Journal for Computer Science and Mathematics **6**(2) (2025)

[4] Tiwari, R.S., Lakshmi, D., Das, T.K., Tripathy, A.K., Li, K.-C.: A lightweight optimized intrusion detection system using machine learning for edge-based iiot security. Telecommunication Systems **87**(3), 605–624 (2024) https://doi.org/10.1007/s11235-024-01200-y

[5] Sahu, D.P., Tripathy, B., Samantaray, L.: Fognet: Custom cnn with optimal feature selection-based combat model for secured fog computing environment. e-Prime - Advances in Electrical Engineering, Electronics and Energy **8**, 100604 (2024) https://doi.org/10.1016/j.prime.2024.100604

[6] Al Mukhaini, G., Anbar, M., Manickam, S., Al-Amiedy, T.A., Al Momani, A.: A systematic literature review of recent lightweight detection approaches leveraging machine and deep learning mechanisms in internet of things networks. Journal of King Saud University - Computer and Information Sciences **36**(1), 101866 (2024) https://doi.org/10.1016/j.jksuci.2023.101866

[7] Karunamurthy, A., Vijayan, K., Kshirsagar, P.R., Tan, K.T.: An optimal federated learning-based intrusion detection for iot environment. Scientific Reports **15**(1), 8696 (2025) https://doi.org/10.1038/s41598-025-93501-8

[8] Olanrewaju-George, B., Pranggono, B.: Federated learning-based intrusion detection system for the internet of things using unsupervised and supervised deep learning models. Cyber Security and Applications **3**, 100068 (2025) https://doi.org/10.1016/j.csa.2024.100068

[9] Aldaej, A., Ahanger, T.A., Ullah, I.: Deep learning-inspired iot-ids mechanism for edge computing environments. Sensors **23**(24) (2023) https://doi.org/10.3390/s23249869

[10] Misrak, S.F., Melaku, H.M.: Lightweight intrusion detection system for iot with improved feature engineering and advanced dynamic quantization. Discover

Internet of Things **5**(1), 97 (2025) https://doi.org/10.1007/s43926-025-00203-8

[11] Fenanir, S., Semchedine, F., Baadache, A.: A machine learning-based lightweight intrusion detection system for the internet of things. Rev. d'Intelligence Artificielle **33**(3), 203–211 (2019)

[12] Patil, S., Varadarajan, V., Mazhar, S.M., Sahibzada, A., Ahmed, N., Sinha, O., Kumar, S., Shaw, K., Kotecha, K.: Explainable artificial intelligence for intrusion detection system. Electronics **11**(19) (2022) https://doi.org/10.3390/electronics11193079

[13] Li, J., Othman, M.S., Chen, H., Yusuf, L.M.: Optimizing iot intrusion detection system: feature selection versus feature extraction in machine learning. Journal of Big Data **11**(1), 36 (2024) https://doi.org/10.1186/s40537-024-00892-y

[14] Jeffrey, N., Tan, Q., Villar, J.R.: A hybrid methodology for anomaly detection in cyber–physical systems. Neurocomputing **568**, 127068 (2024) https://doi.org/10.1016/j.neucom.2023.127068

[15] Hozouri, A., Mirzaei, A., Effatparvar, M.: A comprehensive survey on intrusion detection systems with advances in machine learning, deep learning and emerging cybersecurity challenges. Discover Artificial Intelligence **5**(1), 314 (2025) https://doi.org/10.1007/s44163-025-00578-1

[16] Isong, B., Kgote, O., Abu-Mahfouz, A.: Insights into modern intrusion detection strategies for internet of things ecosystems. Electronics **13**(12) (2024) https://doi.org/10.3390/electronics13122370

[17] Xinyu, T., Pekka, K., Ismo, H.: A design and modeling approach for resource-constrained internet of things devices. Design Automation for Embedded Systems **29**(3), 4 (2025) https://doi.org/10.1007/s10617-025-09294-6

[18] Pandey, V.K., Sahu, D., Prakash, S., Rathore, R.S., Dixit, P., Hunko, I.: A lightweight framework to secure iot devices with limited resources in cloud environments. Scientific Reports **15**(1), 26009 (2025) https://doi.org/10.1038/s41598-025-09885-0

[19] Ma, H., Zhang, W., Zhang, D., Chen, B.: An iot intrusion detection framework based on feature selection and large language models fine-tuning. Scientific Reports **15**(1), 21158 (2025) https://doi.org/10.1038/s41598-025-08905-3

[20] Shanmugam, V., Razavi-Far, R., Hallaji, E.: Addressing class imbalance in intrusion detection: A comprehensive evaluation of machine learning approaches. Electronics **14**(1) (2025) https://doi.org/10.3390/electronics14010069

[21] Nimbalkar, P., Kshirsagar, D.: Feature selection for intrusion detection system in internet-of-things (iot). ICT Express **7**(2), 177–181 (2021) https://doi.org/10.

1016/j.icte.2021.04.012

[22] Hossain, M.A.: Deep learning-based intrusion detection for iot networks: a scalable and efficient approach. EURASIP Journal on Information Security **2025**(1), 28 (2025) https://doi.org/10.1186/s13635-025-00202-w

[23] Sebestyen, H., Popescu, D.E., Zmaranda, R.D.: A literature review on security in the internet of things: Identifying and analysing critical categories. Computers **14**(2) (2025) https://doi.org/10.3390/computers14020061

[24] Mallidi, S.K.R., Ramisetty, R.R.: Advancements in training and deployment strategies for ai-based intrusion detection systems in iot: a systematic literature review. Discover Internet of Things **5**(1), 8 (2025) https://doi.org/10.1007/s43926-025-00099-4

[25] Mallidi, S.K.R., Ramisetty, R.R.: Optimizing intrusion detection for iot: A systematic review of machine learning and deep learning approaches with feature selection and data balancing. WIREs Data Mining and Knowledge Discovery **15**(2), 70008 (2025) https://doi.org/10.1002/widm.70008 https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/widm.70008. e70008 DMKD-00800.R1

[26] Mallidi, S.K.R., Ramisetty, R.R.: A multi-level intrusion detection system for industrial iot using bowerbird courtship-inspired feature selection and hybrid data balancing. Discover Computing **28**(109) (2025) https://doi.org/10.1007/s10791-025-09632-z

[27] Kristiyanti, D.A., Sitanggang, I.S., Annisa, Nurdiati, S.: Feature selection using new version of v-shaped transfer function for salp swarm algorithm in sentiment analysis. Computation **11**(3) (2023) https://doi.org/10.3390/computation11030056

[28] Mallidi, S.K.R., Ramisetty, R.R.: Bowerbird courtship-inspired feature selection for efficient high-dimensional data analysis using a novel meta-heuristic. Discover Computing **28**(1), 6 (2025) https://doi.org/10.1007/s10791-025-09497-2

[29] Ferrag, M.A., Friha, O., Hamouda, D., Maglaras, L., Janicke, H.: Edge-iiotset: A new comprehensive realistic cyber security dataset of iot and iiot applications for centralized and federated learning. IEEE Access **10**, 40281–40306 (2022) https://doi.org/10.1109/ACCESS.2022.3165809

[30] Nack, E.A., McKenzie, M.C., Bastian, N.D.: Aci-iot-2023: A robust dataset for internet of things network security analysis. In: MILCOM 2024 - 2024 IEEE Military Communications Conference (MILCOM), pp. 1–6 (2024). https://doi.org/10.1109/MILCOM61039.2024.10773916

[31] He, K., Kim, D., Zhang, Z., Ge, M., Lam, U.: UQ IoT IDS dataset 2021. The

University of Queensland (2022)

[32] Zolanvari, M., Teixeira, M.A., Gupta, L., Khan, K.M., Jain, R.: Machine learning-based network vulnerability analysis of industrial internet of things. IEEE Internet Things J. **6**(4), 6822–6834 (2019)

[33] Chalichalamala, S., Govindan, N., Kasarapu, R.: Logistic regression ensemble classifier for intrusion detection system in internet of things. Sensors **23**(23) (2023) https://doi.org/10.3390/s23239583

[34] Rahman, K., Aziz, M.A., Usman, N., Kiren, T., Cheema, T.A., Shoukat, H., Bhatia, T.K., Abdollahi, A., Sajid, A.: Cognitive lightweight logistic regression-based ids for iot-enabled fanet to detect cyberattacks. Mobile Information Systems **2023**(1), 7690322 (2023) https://doi.org/10.1155/2023/7690322 https://onlinelibrary.wiley.com/doi/pdf/10.1155/2023/7690322

[35] Alzahrani, H., Sheltami, T., Barnawi, A., Imam, M., Yaser, A.: A lightweight intrusion detection system using convolutional neural network and long short-term memory in fog computing. Computers, Materials & Continua **80**(3), 4703–4728 (2024) https://doi.org/10.32604/cmc.2024.054203

[36] Qathrady, M.A., Ullah, S., Alshehri, M.S., Ahmad, J., Almakdi, S., Alqhtani, S.M., Khan, M.A., Ghaleb, B.: Sacnn-ids: A self-attention convolutional neural network for intrusion detection in industrial internet of things. CAAI Transactions on Intelligence Technology **9**(6), 1398–1411 (2024) https://doi.org/10.1049/cit2.12352 https://ietresearch.onlinelibrary.wiley.com/doi/pdf/10.1049/cit2.12352