JOURNAL OF INFORMATION AND COMMUNICATION TECHNOLOGY
https://e-journal.uum.edu.my/index.php/jict

How to cite this article:
Sh. Daoud, M., Abualhaj, M. M., Al-Khatib, S. S., Al-Zyoud, M., Abu-Sareha, A., Al-Aqrabi, H., Alsharaiah, M., A., & Anbar, M. (2026). A unified machine learning-based IDS/IPS framework with bio-inspired feature selection for real-time detection of malware-laden URLs. *Journal of Information and Communication Technology, 25(3),* 1-30. https://doi.org/10.32890/jict2026.25.3.1

A Unified Machine Learning-Based IDS/IPS Framework with Bio-Inspired Feature Selection for Real-Time Detection of Malware-Laden URLs

[1]Mohammad Sh. Daoud, [2]Mosleh M. Abualhaj, [3]Sumaya S. Al-Khatib,
[4]Mahran Al-Zyoud, [5]Ahmad Abu-Shareha, [6]Hussain Al-Aqrabi,
[7]Mohammad A. Alsharaiah & [8]Mohamad Anbar
[1]College of Engineering, Al Ain University, United Arab Emirates
[2&4]Department of Networks and Cybersecurity,
Al-Ahliyya Amman University, Jordan
[3]Department of Computer Science, Al-Ahliyya Amman University, Jordan
[5]Department of Data Science and Artificial Intelligence,
Al-Ahliyya Amman University, Jordan
[6]Higher College of Technology, United Arab Emirates
[7]Department of Information Technology, King Abdullah II School of Information Technology, The University of Jordan, Jordan
[8]Cybersecurity Research Center (CYRES), Universiti Sains Malaysia, Malaysia

[*1]mohammad.daoud@aau.ac.ae
[2]m.abualhaj@ammanu.edu.jo
[3]sumayakh@ammanu.edu.jo
[4]m.zyoud@ammanu.edu.jo
[5]a.abushareha@ammanu.edu.jo
[6]halaqrabi@hct.ac.ae
[7]m.alsharaiah@ju.edu.jo
[8]anbar@cyres.usm.my
*Corresponding author

Received: 19/1/2026 Revised 9/6/2026 Accepted: 24/6/2026 Published: 31/7/2026

ABSTRACT

Cybersecurity threats have become increasingly sophisticated and dynamic due to the rapid evolution of malicious software and attack techniques. However, the reliance of traditional IDS/IPS implementations on signature-based detection limits their ability to identify novel, rapidly evolving, and adaptive malware, despite their effectiveness against known cyber threats. This paper presents a unified machine-learning framework for defence against malware-laden URLs, which simultaneously targets intrusion detection (accuracy first, out of band) and intrusion prevention (latency first, inline) through module-aware, bio-inspired feature selection. The detection module employs random forest (RF) with the bat algorithm (BA), achieving 99.52% accuracy under stratified fivefold validation. This approach yields thousands of additional correct decisions per million URLs and reduces false negatives and false positives without adverse throughput effects. The prevention module utilizes a decision tree (DT) with the HHO ∩ BA consensus subset, achieving an inference time of 1 ms per URL with competitive accuracy (98.96%), enabling more than 1,000 URLs per second per core and meeting strict inline decision budgets. Standard metrics, such as accuracy, precision, recall, and F1-score, along with timing analyses, confirm that aligning feature selection with module objectives strikes a practical balance between detection quality and latency. The resulting guidance is direct: employ RF with BA for intrusion detection when optimal discriminative performance is essential, and adopt DT with the HHO ∩ BA subset for intrusion prevention when millisecond-scale latency dictates deployment.

Keywords: Bat algorithm, feature selection, Harris Hawks Optimization, intrusion detection prevention systems, machine learning.

INTRODUCTION

The rapid proliferation of digital infrastructures has delivered unprecedented benefits to organisations and individuals, enabling seamless connectivity, efficient communication, and advanced service delivery. However, this growth has heightened the complexity, frequency, and sophistication of cyber threats targeting networks and information systems. Malware, including viruses, worms, ransomware, and spyware, remains a primary weapon for cybercriminals to infiltrate systems, extract sensitive data, and disrupt critical services (Kumi et al., 2021; Rafrastara et al., 2025; Abualhaj, Shambour, et al., 2025). The scale of this threat is alarming: in 2022 alone, 5.5 billion malware attacks were recorded (Abualhaj & Al-khatib, 2024), with over 560,000 new malware samples detected daily in 2023. The global malware landscape now encompasses over one billion variants, and incident rates have risen by 87% in the past decade (Hoang et al., 2023).

A particularly insidious malware delivery method involves compromised or deceptive URLs that entice users to execute malicious code or access harmful resources. These URLs are frequently integrated into phishing emails, social media posts, or compromised websites, and they often employ obfuscation and redirection techniques to evade standard detection. Consequently, effective identification and prevention of malware-laden URLs is essential to preserving the integrity, confidentiality, and availability of network resources (Almomani et al., 2025; Tung et al., 2022).

In modern network defense, an intrusion detection system (IDS) and an intrusion prevention system (IPS) play complementary roles. IDS passively monitors mirrored traffic, analyzing packet copies to identify suspicious activity with minimal impact on legitimate communications. IPS, by contrast,

operates inline, inspecting and blocking traffic in transit, which makes it sensitive to processing delays that can degrade performance in latency-critical environments. When implemented together, IDS and IPS provide a layered defense: IDS delivers near-real-time visibility and alerting, whereas IPS enforces real-time prevention, enabling the detection and mitigation of a broad range of network-based attacks (Mohammad et al., 2022; Sheikh, 2025; Scarfone & Mell, 2007).

Traditional IDS/IPS implementations, primarily based on signature matching, are effective against known threats but struggle against rapidly evolving and adaptive malware (Mohammad et al., 2022). This limitation has led to the adoption of machine learning (ML) techniques, which can automatically learn complex traffic patterns and detect known and zero-day threats. In the context of URL-based malware detection, ML models can analyze lexical and structural features to deliver high-precision, low-latency predictions (Mohammad et al., 2022; Al Saaidah et al., 2024). However, the effectiveness of ML models depends heavily on feature selection, as cybersecurity datasets often contain redundant or irrelevant attributes. High-dimensional feature spaces increase computational complexity, slow training and inference, and reduce detection accuracy. Feature selection techniques are used to retain only the most informative attributes, thereby improving model efficiency and performance (Al Saaidah et al., 2024, Hussain et al., 2025; Yab et al., 2024; Almomani et al., 2023). Bio-inspired metaheuristic algorithms, such as the Harris Hawks optimization (HHO) algorithm and the Bat Algorithm (BA), are particularly effective for this task, using nature-inspired search strategies to identify optimal feature subsets (Abualhaj, Hiari, et al., 2025; Saheed et al., 2023).

This research proposes a unified ML-based IDS/IPS framework for detecting malware-laden URLs, in which each module is tailored to its operational role: an accuracy-oriented IDS and a latency-optimized IPS. The framework integrates feature selection strategies and ML classifiers to balance detection accuracy with real-time responsiveness. Unlike conventional approaches focused solely on classification, this work explicitly targets the detection engine within the IDS/IPS architecture, where feature selection and model assignment influence both accuracy and response time. Thus, the framework extends beyond standalone classification by enabling efficient and accurate real-time detection within IDS/IPS environments. The specific contributions are as follows:

- Optimized IDS Module: A high-efficiency IDS design capable of executing computationally intensive ML-based malware detection, with and without feature selection, to identify subtle malicious patterns.
- Optimized IPS Module: A lightweight, latency-sensitive IPS designed with aggressive feature selection to maintain real-time responsiveness with minimal disruption to legitimate traffic.
- Comprehensive Feature Selection Evaluation: A systematic comparison of five scenarios, namely, no feature selection (NoFs), HHO, BA, HHO ∪ BA (union), and HHO ∩ BA (intersection), to identify the most effective feature subsets for each module.
- Multi-Classifier Implementation: Use of random forest (RF), k-nearest neighbors (KNN), decision tree (DT), and extra tree (ET) classifiers to evaluate detection performance across diverse learning paradigms.
- Hyperparameter Optimization: Application of grid search (GS) to fine-tune classifier parameters for optimal detection performance.
- Extensive Performance Evaluation: Rigorous testing on the ISCX-URL2016 dataset using K-fold cross-validation and multiple metrics, including accuracy, precision, recall, F1-score, training time, and inference time.

RELATED WORKS

This section outlines the key components of the proposed unified ML-based IDS/IPS framework. It briefly describes the ISCX-URL2016 dataset used for malware-laden URL detection, the HHO algorithm and BA for feature selection, and the four classifiers (i.e., RF, KNN, DT and ET) employed in this study. In addition, an overview of the current methods for detecting malware is provided, and the need that inspires the proposed unified ML-based IDS/IPS framework with bio-inspired feature selection is described.

Dataset: ISCX-URL2016

The ISCX-URL2016 Malware dataset (Mamun et al., 2016), released by the Canadian Institute for Cybersecurity (CIC), contains 14,493 records comprising 6,712 malicious and 7,781 benign URLs. Benign samples are drawn from Alexa-ranked domains, whereas malicious ones originate from the DNS-BH blacklist (Mamun et al., 2016). The dataset provides 79 lexical features from URL components (domain, path, arguments, and filename), excluding content-based attributes such as HTML or JavaScript to emphasize lightweight structural patterns. Table 1 lists the 79 features of the ISCX-URL2016 malware dataset (Abu Al-Haija & Al-Fayoumi, 2023). Some features require preprocessing through normalization and imputation because of missing values. Malicious URLs often employ obfuscation and encoding, so lexical analysis offers an efficient alternative to content-based inspection, particularly in latency-sensitive environments like IPSs. Given its balanced class distribution, comprehensive lexical design, and real-world applicability, ISCX-URL2016 has become a widely used benchmark for evaluating malware URL detection models, lightweight classifiers, adversarial robustness, and feature selection methods (Mamun et al., 2016; Abu Al-Haija & Al-Fayoumi, 2023; Abualhaj, Abu-Shareha, et al., 2025).

Table 1

Features of the ISCX-URL2016 Malware Dataset

#	Feature Name	#	Feature Name	#	Feature Name	#	Feature Name
1	Querylength	21	domainlength	41	Directory_ DigitCount	61	delimeter_ Count
2	domain_token_ count	22	pathLength	42	File_name_ DigitCount	62	NumberRate_ URL
3	path_token_ count	23	subDirLen	43	Extension_ DigitCount	63	NumberRate_ Domain
4	Avgdomaintok enlen	24	fileNameLen	44	Query_Digit Count	64	NumberRate_ DirectoryName
5	Longdomaintok enlen	25	this.fileExtLen	45	URL_Letter_ Count	65	NumberRate_ FileName
6	Avgpathtok enlen	26	ArgLen	46	host_letter_ count	66	NumberRate_ Extension
7	tld	27	pathurlRatio	47	Directory_Letter Count	67	NumberRate_ AfterPath
8	charcompvowels	28	ArgUrlRatio	48	Filename_Letter Count	68	SymbolCount_ URL
9	charcompace	29	argDomanRatio	49	Extension_Letter Count	69	SymbolCount_ Domain

(continued)

#	Feature Name	#	Feature Name	#	Feature Name	#	Feature Name
10	ldl_url	30	domainUrlRatio	50	Query_Letter Count	70	SymbolCount_ Directoryname
11	ldl_domain	31	pathDomainRatio	51	LongestPath TokenLength	71	SymbolCount_ FileName
12	ldl_path	32	argPathRatio	52	Domain_Longest WordLength	72	SymbolCount_ Extension
13	ldl_filename	33	executable	53	Path_Longest WordLength	73	SymbolCount_ Afterpath
14	ldl_getArg	34	isPortEighty	54	sub-Directory_ LongestWordLength	74	Entropy_ URL
15	dld_url	35	NumberofDotsin URL	55	Arguments_ Longest WordLength	75	Entropy_ Domain
16	dld_domain	36	ISIpAddressIn DomainName	56	URL_sensitive Word	76	Entropy_ DirectoryName
17	dld_path	37	CharacterContinuity Rate	57	URLQueries_ variable	77	Entropy_ Filename
18	dld_filename	38	LongestVariable Value	58	spcharUrl	78	Entropy_ Extension
19	dld_getArg	39	URL_DigitCount	59	delimeter_ Domain	79	Entropy_ Afterpath
20	urlLen	40	host_DigitCount	60	delimeter_ path		

Harris Hawks Optimization Algorithm

The Harris Hawks Optimization (HHO) algorithm is a population-based metaheuristic inspired by the cooperative hunting strategy of Harris hawks. The algorithm models the dynamic interactions between hawks (candidate solutions) and their prey (optimal solution) via adaptive exploration–exploitation mechanisms. The prey's escape energy parameter (E) determines the search phase:

- Exploration ($|E| \geq 1$) — Hawks search widely across the solution space to avoid premature convergence.
- Exploitation ($|E| < 1$) — Hawks focus on local search around promising regions to refine solutions.

HHO employs different attacking strategies, such as soft besiege, hard besiege, and progressive rapid dives, to simulate real hunting tactics. This flexibility enables HHO to balance global search capability and convergence accuracy, making it effective for high-dimensional feature selection tasks (Heidari et al., 2019; Tripathy, 2022).

Bat Algorithm

The Bat Algorithm (BA) is a swarm intelligence optimization method inspired by the echolocation behavior of microbats. Each bat represents a candidate solution and moves in the search space with velocity and position updated based on a frequency parameter, loudness, and pulse emission rate.

- Frequency controls the exploration range.
- Loudness decreases as solutions improve, simulating the bat approaching its prey.
- Pulse rate increases to intensify exploitation near promising regions.

BA adaptively switches from exploration to exploitation, enabling fast convergence while maintaining the ability to escape local optima. Its balance between randomness and adaptive intensification makes it suitable for selecting optimal feature subsets in complex datasets (Pethe et al., 2024; Sagagi et al., 2024).

Classifiers

The classifiers used in this research are selected for their complementary ability to deal with structured URL features and their efficiency in terms of time and space complexity while maintaining accuracy. These classifiers are RF, KNN, DT, and ET. On the other hand, although Support Vector Machine and Artificial Neural Network (ANN) are accurate (AlTalhi et al., 2021), they have higher inference latency and computational cost, which makes them unsuitable for real-time IPS deployment.

The RF is an ensemble learning algorithm that is based on the bagging paradigm, which trains multiple decision trees on bootstrapped subsets of the data. The final class is decided by the votes of each tree, which helps to reduce variance and prevent overfitting. RF naturally estimates feature importance, which is suitable for high-dimensional datasets, such as lexical attributes of URLs (Qaddara & Alraba'nah, 2025; Al-dabbas, 2024). On the other hand, KNN is a non-parametric, instance-based learning algorithm that classifies a sample based on the majority label among the k-nearest neighbors in the feature space. No model training is required, and it uses distance measures such as Euclidean or Manhattan distance. KNN is capable of capturing local decision boundaries, which makes it suitable for capturing subtle differences between benign and malicious URLs (Shambour et al., 2024; Abualhaj, Abu-Shareha, et al., 2024).

The DT is a supervised learning model that divides the data into segments according to feature-based conditions and constructs a decision tree structure. The algorithm chooses the split that maximizes the information gain or minimises the impurity, such as the Gini index or entropy, at each node. DTs are efficient and provide interpretable and transparent decision rules. They can overfit to the training data, but this issue can be addressed by implementing depth control, pruning, and cross-validation (Shambour et al., 2024; Abualhaj, Al-Shamayleh, et al., 2024). Besides than DT, this study also deploy ET which is an ensemble method, similar to RF, but with additional randomness in splitting, by choosing split thresholds randomly for each feature. This can further decrease variance, increase generalization, and train quickly. ET is also appropriate for fast detection situations, such as IPS modules, where computational efficiency is paramount (Afshar et al., 2022; Kocev et al., 2020). Table 2 summarizes the strengths, limitations, and rationale for selecting RF, KNN, DT, and ET in our framework.

Table 2

Strengths, Limitations, and Study-Specific Rational for RF, KNN, DT, and ET

Classifier	Strengths	Limitations	Rationale in this study
RF	High accuracy via variance reduction; robust to noise and redundant lexical features; provides feature-importance signals; stable across folds.	Large memory footprint; inference latency rises with number/depth of trees.	Strong candidate for IDS where accuracy is prioritized; also serves as a reliable wrapper evaluator and tuning baseline across feature-selection settings.
KNN	Non-parametric; captures fine local decision boundaries; benefits from reduced dimensionality after feature selection.	Sensitive to feature scaling; high inference cost due to distance computations; performance degrades with many irrelevant features.	Useful on compact subsets (e.g., HHO∩BA) to assess gains from local similarity; viable for IPS when dimensionality is aggressively reduced.
DT	Transparent rules; very fast inference; handles mixed feature types without complex preprocessing.	Prone to high variance; can underfit or overfit depending on depth/splits; less robust than ensembles.	Establishes interpretable baseline and latency floor; suitable for IPS-style deployment when simplicity and response time dominate.
ET	Fast training and inference; strong generalization from extreme randomization; competitive accuracy with fewer knobs to tune.	Excessive randomness can increase bias if trees are shallow or too few; feature importance less stable than RF.	Attractive speed–accuracy trade-off for both modules; a leading IPS candidate on reduced subsets and competitive for IDS on richer subsets.

Researchers have proposed several methods for malware detection. The hybrid model for real-time malicious URL detection proposed by Swetha et al. (2024) uses a combination of machine learning techniques. The model is a fusion of two models: self-organizing map-based radial movement optimization (SOM-RMO) and radial basis function network (RBFN) classification, in which radial movement optimization (RMO) is performed on SOM to optimise the representation of topological features. SOM-RMO performs dimensionality reduction and feature extraction. Meanwhile, tabu search is used to optimize the RBFN classifier to prevent it from getting stuck in local optima and overfitting. The method is based on 651,191 URLs provided by real-world datasets, including ISCX-URL-2016 and PhishTank. The dataset contains benign, phishing, malware, and defacement categories, and the input features include lexical, host-based, and content-based features. The proposed model achieved an accuracy of 96.5% and outperformed the En_kNN baseline.

In the case of Okunnuga (2023), supervised ML models are used for detecting malicious URLs. Defacement, phishing, spam, and malware attacks are all found in these types of URLs. This study uses the ISCX-URL2016 dataset, which consists of 36,707 labeled URLs, and tests classifiers such as RF, logistic regression (LR), support vector classifiers (SVCs), KNN, and DT. Data preprocessing includes handling NaN values and numerical encoding of labels, while dimensionality reduction and computational efficiency are achieved using principal component analysis (PCA). The best model,

which achieved the highest binary accuracy, was RF with a score of 98.9%, followed by KNN and DT, with accuracies of more than 96%. By contrast, SVC had the lowest performance, with 90.6% accuracy. Interestingly, RF had very few false positives and false negatives.

Anne and CarolinJeeva (2021) discuss and analyze the potential for enhancing ML algorithms to classify URLs as benign or malicious. The study compares the performance of Light Gradient Boosting Machine (LGBM), XGBoost, and gradient boosting (GBoost) algorithms for malicious URL detection using the ISCX-URL2016 dataset, which contains 18,982 labeled URL samples. For simplicity in classification, the malicious activities are grouped under phishing, spam, malware, and defacement. Data are preprocessed by eliminating irrelevant, duplicate, and missing data. Feature importance is calculated by taking the average score from the boosting models, and the 20 most important features for training the final RF model are chosen. These selected features are then used to train the RF algorithm to increase prediction accuracy. The accuracy of LGBM was 98%, while XGBoost and GBoost both achieved 94% overall classification accuracy. With just 29 misclassified instances, RF further enhanced performance with an accuracy of 99%.

Gopal et al. (2022) present a deep learning (DL)-based autoencoder–artificial neural network (ANN) model for phishing detection. The model is trained on the ISCX-URL2016 dataset, which consists of five URL categories: phishing, spam, malware, defacement, and benign. It preprocesses raw URL data, including missing and noisy data. Dimensionality reduction is achieved using an autoencoder, and the encoded features are fed into a deep ANN for classification. The study examines the influence of different ANN epochs and the number of hidden layers; it also examines how optimizers such as RMSprop, Nadam, and Adam can be used to tune ANN performance. The standalone ANN achieved 95.6% accuracy in phishing URL detection, while the autoencoder–ANN achieved 98.89% accuracy using the Adam optimizer. The overall performance of the proposed method improved by 3.2% compared to the baseline PCA-RF method.

Le et al. (2018) introduce a DL framework for detecting malicious URLs (URLNet), which directly processes raw URL strings without using hand-crafted features. The model combines character-level and word-level convolutional neural networks (CNNs) to facilitate representation learning. URLs are encoded using character and word embeddings, with the latter capturing structural patterns and being effective with rare or unseen words. The architecture consists of convolutional and fully connected layers for end-to-end binary classification, coupled with pooling layers to lower computational complexity while retaining important spatial characteristics. Experiments conducted on a large dataset of 15 million URLs from VirusTotal showed good results, with an accuracy of 97.07% and an AUC of 0.9929.

Lee et al. (2021) introduce a genetic algorithm (GA)-based feature selection model for ML-based Android malware detection. Androguard is used to extract static features, such as permissions and API calls, from APKs. The features are binary encoded and have a total dimension of 1104 before selection. The study involves three types of feature selection, namely GA, IG, and no selection. GA-based feature selection reduces the number of features from 1104 to 101, which increases the efficiency of the model. The highest accuracy was obtained by MLP, achieving 98.4% using the most optimal features selected by GA. Among classifiers using IG, the top five accuracies were IBk (97.0%), RFt (96.8%), multi-layer perceptron (MLP) (96.7%), LR (95.4%), and SMO (95.2%). By contrast, GA yielded superior results with MLP (98.1%), RF (97.8%), IBk (97.2%), J48 (96.8%), and SMO (96.4%).

In this study, Hasan et al. (2025) examine how feature selection and scaling affect malware detection using ML. CICMalDroid 2020 is the dataset used, with 11,598 malware-related samples and 139 features. To address class imbalance in the data, the SMOTE method is used. The study uses PCA and linear discriminant analysis (LDA) as dimensionality reduction techniques, and min–max and L2 normalization as feature scaling techniques. PCA maximizes variance, while LDA increases class separability. Experimental results show a significant improvement in ML model performance due to these preprocessing steps. For each configuration, ensemble models outperform traditional classifiers, with LGBM, ANN, and RF ranking among the top three. With PCA + min–max scaling, the top accuracies were achieved by LGBM (97.16%), ANN (96.94%), and RF (96.47%), slightly outperforming LDA + min–max scaling, where the top accuracies were achieved by LGBM (96.80%), ANN (96.72%), and RF (96.55%). Overall, PCA slightly improves the performance of the top classifiers compared to LDA.

Gu et al. (2024) introduce a hybrid feature selection approach that integrates mutual information (MI) with an improved (IHHO) algorithm for detecting malicious encrypted traffic. MI is first applied to eliminate irrelevant and redundant features by measuring statistical dependencies between features and the target variable. The subsequent IHHO stage refines feature selection through several enhancement strategies, including tent chaotic mapping, sine–cosine exploration, periodic escape energy adjustment, adaptive weighting, and a V-shaped transfer function for binary mapping. The method is evaluated using the Malware Capture Facility Project (MCFP) dataset, which initially comprises 431 traffic-related features. Feature selection reduces this set to 50 features, which are then used to train a deep learning classifier based on long short-term memory (LSTM) networks. The final LSTM-based model achieves a classification accuracy of 94.25%.

Table 3 summarises the previously discussed malware detection approaches. Although previous research has achieved high classification accuracy, most approaches in the literature focus on prediction and ignore deployment constraints, such as inference time and real-time requirements. This limitation is of paramount importance in real-world cybersecurity environments, especially for IPS. Furthermore, while many studies have reported high accuracy of nearly 99%, there are still opportunities to make the system more robust and generalisable and to minimise misclassifications at scale. Another notable limitation of previous studies is that they do not differentiate between IDS and IPS and usually evaluate feature selection solely by accuracy, without considering computational efficiency. Thus, a common standard is required that is accurate and suitable for real-time or near-real-time deployment. In this study, researcher address this gap by proposing a module-aware IDS/IPS framework based on feature selection and model design aligned with operational requirements.

Table 3

Malware Detection Techniques

References	Classification model	Feature reduction method	Dataset name	Results (Accuracy)
Swetha et al. (2024)	RBFN	SOM-RMO	ISCX-URL2016	96.5%
Okunnuga (2023)	RF	PCA		98.9%
Anne and Jeeva (2021)	LGBM	Feature importance		98%
	XGBoost			94%
	GBoost			94%

(continued)

References	Classification model	Feature reduction method	Dataset name	Results (Accuracy)
Gopal et al. (2022)	RF	Autoencoder	15M URLs (VirusTotal)	99%
	Autoencoder + ANN			98.89%
Le et al. (2018)	CNN	Implicitly using CNN		97.07%
Lee et al. (2021)	IBk	IG	Andro-AutoPsy	97.0%
	RF			96.8%
	MLP			96.7%
	LR			95.4%
	SMO			95.2%
	Tuned MLP			98.4%
	MLP	GA		98.1%
	RF			97.8%
	IBk			97.2%
	J48			96.8%
Hasan et al. (2025)	LGBM	PCA	CICMalDroid 2020	97.16%
	ANN			96.94%
	RF			96.47%
	LGBM			96.80%
	ANN	LDA		96.72%
	RF			96.55%
Gu et al. (2024)	LSTM	MI and IHHO	MCFP	94.25%

THE PROPOSED METHOD

This section presents the proposed unified ML-based IDS/IPS framework for malware detection. It covers data preprocessing, the development of a bio-inspired feature selection strategy that combines the HHO and BA algorithms, and the classification stage using selected ML models. Figure 1 clarifies the proposed unified ML-based IDS/IPS framework. Within the IDS/IPS architecture, the proposed method primarily enhances the detection engine, where feature selection and ML models are applied to classify URLs as benign or malicious. In the IDS module, the detection engine emphasizes accuracy using richer feature subsets, whereas in the IPS module, it prioritizes low-latency decision-making using compact feature subsets, enabling effective real-time operation.

Data Preparation

The ISCX-URL2016 malware dataset contains 79 lexical features. The dataset is well-structured, but initial inspection revealed the presence of noisy and missing values that could adversely affect model training and evaluation. To ensure data integrity and maintain consistency (Fan et al., 2021), we applied the following preprocessing steps:

i. Handling Missing Values with Median Imputation: For continuous features such as avgpathtokenlen (Feature #6), Entropy_Filename (Feature #77), and Entropy_Extension (Feature #78), missing values were replaced with the median of the respective feature. Median imputation was chosen over mean imputation to reduce sensitivity to skewed distributions and outliers, preserving the robustness of the feature values.

ii. Row Removal for Isolated Missing or Infinite Values: For features such as argPathRatio (Feature #32), NumberRate_DirectoryName (Feature #64), NumberRate_FileName (Feature #65), and NumberRate_AfterPath (Feature #67), a single entry (< 0.01%) contained missing or infinite values. In this case, the affected row was removed entirely to prevent bias from imputation in extremely sparse cases.

iii. Feature Removal Due to High Missingness: NumberRate_Extension (Feature #66) and Entropy_DirectoryName (Feature #76) exhibited a substantial proportion of missing values (5927 [40.9%] and 1880 [13.0%] entries, respectively). Retaining these features could lead to biased imputation and reduced data quality; therefore, they were dropped entirely from the dataset.

Through these preprocessing measures, the dataset was cleaned, ensuring that the remaining features were free from critical noise, complete in coverage, and ready for normalisation. The ISCX-URL2016 malware dataset contains attributes with widely varying magnitudes. Such disparities in scale can negatively influence the effectiveness of ML models. Normalisation is applied to bring all features into a consistent range. This process ensures that each attribute contributes equally during model training and avoids favouring features with inherently large numeric values. In this work, min–max normalisation is employed to transform all feature values into the [0, 1] range, and their relative ranking and proportional differences are maintained. This approach improves training consistency and accelerates model convergence (Abu-Shareha, 2024). Table 4 shows a sample of the ISCX-URL2016 malware dataset before and after normalization.

Figure 1

Proposed Unified ML- Based IDS/IPS Framework

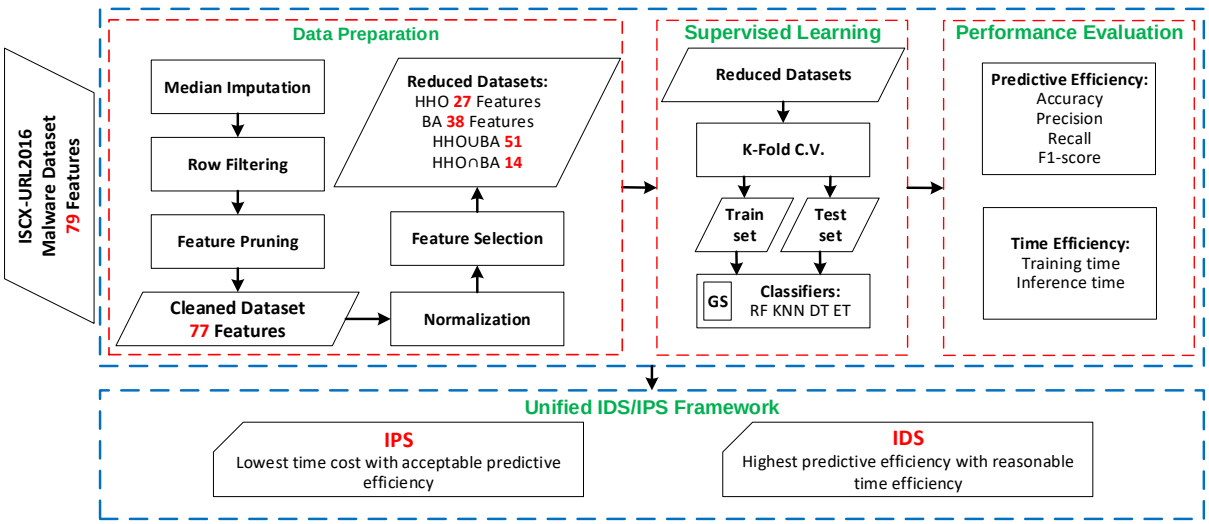


Table 4

Sample of the Dataset Before and After Normalization

Before normalization	After normalization
[2, 9, 11, 4.1, 15, 10, 5.5]	[0, 0.205882353, 0.177419355, 0.0484375, 0.159574468, 0.121212121, 0.256097555]
[10, 9, 2, 6, 2.25, 5, 9]	[0.121212121, 0.205882353, 0, 0.29268292, 0.01953125, 0.21951219, 0.095744681]
[2, 16, 4, 11, 11, 5, 4.75]	[0, 0.170212766, 0.14634146, 0.136363636, 0.177419355, 0.21951219, 0.05859375]
[12, 10, 3, 3.5833333, 12, 8, 5]	[0.151515152, 0.235294118, 0.166666667, 0.040364583, 0.127659574, 0.129032258, 0.21951219]
[7, 12, 8, 5.5, 2, 4.0833335, 15]	[0.112903226, 0.151515152, 0.176470588, 0.256097555, 0, 0.048177086, 0.159574468]
[9, 4.5, 2, 24, 12, 6, 5.3333335]	[0.14516129, 0.182926825, 0, 0.255319149, 0.151515152, 0.29268292, 0.067708336]
[9, 2, 2.5, 6, 3, 4.5555553, 3]	[0.106060606, 0, 0.036585365, 0.063829787, 0.166666667, 0.055555552, 0.048387097]
[7, 7, 4.5, 5.714286, 15, 10, 2]	[0.075757576, 0.147058824, 0.182926825, 0.073660719, 0.159574468, 0.161290323, 0]
[11, 2, 3.5, 4, 3.909091, 15, 6]	[0.136363636, 0, 0.109756095, 0.058823529, 0.045454547, 0.159574468, 0.096774194]

Following the normalization step, the HHO algorithm and BA are employed to perform feature selection, further refining the dataset. The subsequent section (proposed metaheuristic-based feature selection framework) provides a detailed explanation of these feature selection techniques. The resulting feature subsets are then utilised for training the classifiers. This preprocessing workflow produces a clean, standardised, and well-optimized feature space, thereby improving the predictive accuracy and interpretability of the resulting models.

Proposed Metaheuristic-Based Feature Selection Framework

In the proposed unified ML-based IDS/IPS framework for detecting malware-laden URLs, feature selection is essential for achieving the framework's dual objectives: high-accuracy detection in the IDS module and low-latency decision-making in the IPS module. The IDS can afford to process a rich feature set to capture subtle malicious patterns, whereas the IPS must operate on aggressively reduced feature subsets to minimize delays in legitimate traffic flow. Using only one feature selection method is computationally simple, but it can suffer from limited search diversity, overemphasis on either relevance or redundancy, and the potential omission of weak yet important features (Abualhaj, Al-Khatib, et al., 2025; Potharlanka & Nirupama, 2024; Liu et al., 2021). This limitation affects the predictive efficiency of IDS and the time efficiency of IPS. To overcome these challenges, the study adopts four feature selection methods along with no feature selection, resulting in a total of five configurations, as shown in the following subsection.

Feature Selection Framework

The study evaluates five configurations: no feature selection, the two adopted base metaheuristics HHO and BA, and the two proposed hybrid strategies HHO∪BA and HHO∩BA. On the 77-feature processed ISCX-URL2016 dataset, HHO selected 27 features, which corresponded to a reduction of 64.9 percent,

whereas BA selected 38 features, which corresponded to a reduction of 50.7 percent. The union strategy is defined in Equation 1 and implemented in Algorithm 1. It constructs a subset that contains every distinct feature selected by either HHO or BA. Applied to the dataset, HHO∪BA yielded 51 features, a reduction of approximately 33.8 percent. The intersection strategy is defined in Equation 2 and implemented in Algorithm 1. It retains only those features selected by HHO and BA. Applied to the dataset, HHO∩BA produced 14 features, which is a reduction of 81.8 percent. All five configurations are evaluated. Feature subsets are not evaluated exhaustively; instead, HHO and BA explore the search space to identify optimised subsets, and their resulting combinations (HHO, BA, HHO ∪ BA, HHO ∩ BA) are evaluated using classification performance metrics. For IDS, the configuration with the highest predictive efficiency under reasonable time efficiency is selected. For IPS, the configuration with the lowest time cost under acceptable predictive efficiency is chosen. Table 5 shows the features selected by each method, and the resulting subsets are then used for training and evaluation.

$$S_{union} = S_{HHO} \cup S_{BA} = \{f_j \mid f_j \in S_{HHO} \text{ or } f_j \in S_{BA}\} \quad (1)$$

$$S_{inter} = S_{HHO} \cap S_{BA} = \{f_j \mid f_j \in S_{HHO} \text{ and } f_j \in S_{BA}\} \quad (2)$$

Where,

- S_{HHO} is the feature selected by the HHO algorithm.
- S_{BA} is the feature selected by BA.
- S_{union} is the feature selected by the HHO ∪ BA method.
- S_{inter} is the feature selected by the HHO ∩ BA method.

Algorithm 1. HHO ∪ BA and HHO ∩ BA

Input:
S_{HHO}, Best feature subset from HHO
S_{BA}, Best feature subset from BA
Output:
Final selected feature subset S_{union}
Final selected feature subset S_{inter}
Begin

1. Receive S_{HHO} = {f_j | features selected by HHO}
2. Receive S_{BA} = {f_j | features selected by BA}
3. Compute the union of features:
   a. Initialize S_{union} and S_{inter} as empty sets
   b. For each feature f in S_{HHO} do
      Add f to S_{union}
      End For
   c. For each feature f in S_{BA} do
      If f not in S_{union} then
      Add f to S_{union}
      End If
      If f in S_{HHO} then
      Add f to S_{inter}
      End If
   End For
4. Return S_{union} and S_{inter}

End

Table 5

Selected Features

Method	No. of features	Selected features
HHO	27	2, 3, 5, 6, 15, 23, 24, 29, 30, 41, 45, 46, 50, 52, 53, 56, 57, 60, 61, 63, 69, 70, 71, 73, 74, 75, 78
BA	38	1, 2, 4, 5, 7, 8, 9, 10, 16, 19, 20, 25, 29, 32, 33, 37, 38, 40, 43, 44, 45, 46, 49, 50, 52, 55, 56, 58, 59, 61, 68, 69, 70, 71, 72, 75, 78, 79
HHO ∪ BA	51	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 16, 19, 20, 23, 24, 25, 29, 30, 32, 33, 37, 38, 40, 41, 43, 44, 45, 46, 49, 50, 52, 53, 55, 56, 57, 58, 59, 60, 61, 63, 68, 69, 70, 71, 72, 73, 74, 75, 78, 79
HHO ∩ BA	14	2, 5, 29, 45, 46, 50, 52, 56, 61, 69, 70, 71, 75, 78

Rationale for Combining HHO and BA in Feature Selection

The combination of complementary search characteristics and metaheuristic algorithms is effective for HHO and BA. HHO, with its prey–predator modelling approach and Lévy flight dynamics, provides excellent global exploration, while BA, with its frequency tuning and velocity adjustments, provides excellent local refinement. Together, these complementary behaviours allow the union strategy to benefit from feature diversity, while the intersection strategy achieves stability through consensus. The following are the reasons why HHO and BA are chosen as a complementary pair (Liu et al., 2021; Spooner et al., 2023; Zhang & Jonassen, 2019):

- Exploration behaviour: HHO's Lévy flight-driven diversification paired with BA's frequency-tuned local search yields a balanced exploration–exploitation dynamic and broad discovery of informative features.
- Mathematical update strategy: HHO's energy-coefficient updates and BA's frequency/velocity-modulated updates are orthogonal, resulting in diverse candidate subsets and reducing selection bias.
- Position update mechanism: Energy-driven, prey-escape trajectories in HHO complement BA's frequency-based random walks around the best solution, expanding the regions visited in the search space.
- Feature subset selection behaviour: HHO emphasizes high-impact features, whereas BA preserves weak-but-cumulative contributors via local refinement; their union captures strong and subtle signals.
- Search-landscape interaction: HHO adapts rapidly to dynamic fitness landscapes through energy modulation, whereas BA offers smooth, fine-grained refinement; together they improve robustness to diverse landscapes.
- Fitness function compatibility: Both align with classification-driven fitness (and multi-objective formulations), enabling unified evaluation and consistent subset merging while leveraging BA's local sensitivity and HHO's global reach.

Classification with Optimized Machine Learning Models

The classification step represents the classification layer in the pipeline. Both the full lexical feature set and one of the selected subsets, namely HHO, BA, HHO ∪ BA, or HHO ∩ BA, are fed into each model, which then generates a binary decision, malware versus benign, for each URL. This stage operationalizes the framework's two deployment modes: accuracy-oriented and latency-oriented. The final assignment is presented in the Performance Evaluation section.

Experiments are conducted using stratified five-fold cross-validation, with the training set consisting of 80% of the data and the testing set consisting of 20% of the data in each fold. Stratified five-fold cross-validation is used to preserve the class distribution and ensure a reliable and unbiased estimate of system performance while maintaining computational efficiency. For each fold, all transformations are performed on the training split only and then applied to the held-out split to avoid information leakage. Each fold is used once for validation, while the other four folds are used for training. The performance metrics, namely accuracy, precision, recall, and F1-score, are calculated per fold and then averaged across the folds. Training time and per-sample inference time are also calculated per fold and then averaged. The classifiers evaluated are Random Forest (RF), KNN, Decision Tree (DT), and Ensemble Tree (ET). The hyperparameter values selected by the GS algorithm for these models are shown in Table 6.

Table 6

Hyperparameters of RF, KNN, DT, and ET

Classifier	Hyperparameters
RF	n_estimators=300, max_depth=30, min_samples_split=2, min_samples_leaf=1, max_features='sqrt', bootstrap=True, criterion='gini', class_weight=None, n_jobs=-1, random_state=42
KNN	n_neighbors=7, weights='distance', metric='minkowski', p=2, algorithm='auto', leaf_size=30
DT	criterion='gini', max_depth=16, min_samples_split=4, min_samples_leaf=2, max_features='sqrt', splitter='best', ccp_alpha=0.0, random_state=42
ET	n_estimators=300, max_depth=30, min_samples_split=2, min_samples_leaf=1, max_features='sqrt', bootstrap=False, criterion='gini', n_jobs=-1, random_state=42

EVALUATION

This section presents the evaluation of the proposed malware detection framework. It describes the implementation setup, outlines the performance metrics used for assessment, and provides a detailed analysis of experimental results, including classification performance, comparisons with existing methods, and runtime efficiency.

Implementation

These experiments were performed on a high-performance workstation, the Dell Precision 7865 Tower. It is powered by an AMD Ryzen Threadripper PRO 5955WX processor (16 cores, 32 threads) running at 4.00 GHz and an NVIDIA Quadro P5000 graphics card. The system comes with 512 GB of DDR4 ECC memory and 4 TB of NVMe SSD storage. It is equipped with Windows 11 Pro as its operating system, which ensures a stable and optimized computing environment for heavy-duty applications.

The experiment was carried out in four phases. First, the URL data were preprocessed by filling in missing values, and the numerical features were min–max normalized to place them on the same scale. Second, the processed ISCX-URL2016 dataset containing malware and benign URLs was used for bio-inspired feature selection. Five different configurations were tested: NoFS, HHO, BA, HHO ∪ BA (union), and HHO ∩ BA (intersection), using Equations 1 and 2 and Algorithm 1. Third, each feature subset was used to train the classifiers (RF, KNN, DT, and ET) to generate malware-versus-benign decisions. Finally, stratified five-fold cross-validation was performed, where each fold was used once for validation and the remaining folds were used for training. The metrics were then averaged across the folds.

Performance Evaluation Metrics

To comprehensively evaluate the proposed framework, we adopt a suite of performance metrics based on the confusion matrix, as shown in Figure 2: accuracy (Equation 3), precision (Equation 4), recall (Equation 5), and F1-score (Equation 6). We also report training time and per-sample inference time to quantify the accuracy and latency trade-off that influences IDS and IPS deployments. All metrics are computed per fold and averaged over a stratified fivefold cross-validation. Table 7 lists the evaluation metrics (Yeop et al., 2025; Hersyaputra et al., 2026; Alraba'nah & Toghuj, 2024; Alraba'nah et al., 2025).

Figure 2

Confusion Matrix

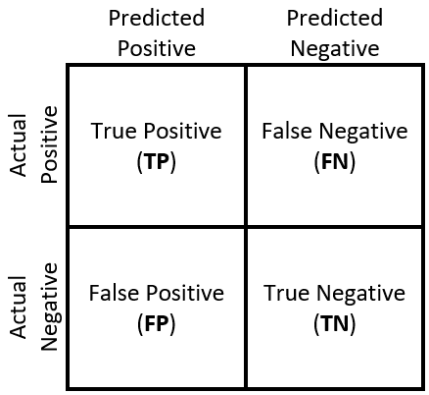


Table 7

List of Evaluation Metrics

Metric	Definition	Purpose	Indication
Accuracy	Proportion of correctly classified URLs among all URLs	Overall correctness on a balanced dataset	Higher is better. Values close to 1.0 indicate consistently correct decisions. Useful for quick comparison but can mask tradeoffs between precision and recall.
Precision	Proportion of predicted malware that are truly malware	Limit false positives and avoid blocking benign traffic	Higher is better. Higher values mean fewer false alarms. Especially important for IPS where unnecessary blocking degrades user experience and service availability.

(continued)

Metric	Definition	Purpose	Indication
Recall (TPR)	Proportion of malware correctly detected	Limit false negatives and avoid missed attacks	Higher is better. Higher values mean fewer missed malware instances. Especially important for IDS where comprehensive detection is prioritized.
F1 score	Harmonic mean of precision and recall	Single measure that balances false positives and false negatives	Higher is better. High F1 requires precision and recall to be strong. Useful for selecting models that balance detection and false alarms.
Training time	Wall clock time required to train the model	Measure computational cost of model building and retraining	Lower is better. Lower values enable faster experimentation and more frequent retraining. Not a deployment latency metric but affects operational agility.
Per sample inference time	Average time to classify one URL	Measure deployment latency for IDS and IPS	Lower is better. Lower values support real time decisions. IPS typically requires very low latency, while IDS can tolerate higher latency if accuracy is improved.

$$Accuracy = \frac{(TP+TN)}{(TP+TN+FP+FN)} \quad (3)$$

$$Recall = \frac{TP}{(TP+FN)} \quad (4)$$

$$Precision = \frac{TP}{(TP+FP)} \quad (5)$$

$$F1-score = 2 \times \frac{Pre \times Rec}{Pre+Rec} \quad (6)$$

Experimental Results

Insights and Interpretation of Classification Results

Figure 3 presents the classification accuracy for RF, KNN, DT, and ET under five feature–selection settings: HHO ∪ BA, HHO ∩ BA, HHO, BA, and NoFs. The models achieve consistently high accuracy. For ET, the union method attains the best result at 99.52%, exceeding BA and NoFs at 99.48% and HHO at 99.45%, and surpassing the intersection by 0.21 percentage points. For KNN, BA achieves the highest accuracy at 98.59%, followed by the union at 98.55%. BA improves upon HHO and NoFs (both 98.48%) by 0.11 points, whereas the union improves upon them by 0.07 points, and the intersection yields the lowest value at 98.31%. For RF, BA achieves the highest accuracy at 99.52%, edging the union by 0.07 points and NoFs by 0.07 points; the intersection is lowest at 99.24%. For DT, BA is again best at 99.00%, tied with NoFs at 99.00%, slightly above the union at 98.97% and intersection at 98.96%, and clearly above HHO at 98.69%. Overall, the union strategy is best for ET and remains competitive for the other models, whereas BA leads for RF and KNN and is co-best for DT; the intersection tends to trade a small amount of accuracy for a compact feature set. The consistently high values indicate strong separability between malware and benign URLs with minimal misclassification.

Although RF-BA and ET-Union achieve similar accuracy, they differ in feature size and purpose. The union subset improves feature coverage and supports higher robustness, benefiting accuracy-oriented IDS, while smaller subsets (e.g., BA or HHO ∩ BA) enhance efficiency and reduce latency, which is critical for IPS. This trade-off explains the role of union and intersection strategies in the proposed framework.

Table 8 reports pairwise accuracy gaps in percentage points for each classifier to make the differences explicit. Values are "row minus column." Positive numbers indicate that the row configuration is more accurate than the column. Overall, these results show that ensembles benefit most from the union strategy, intersection sacrifices a small amount of accuracy likely in exchange for latency, and the no-selection baseline remains competitive. Section Findings and Deployment Guidance for the proposed unified ML-based IDS/IPS framework discusses these patterns concerning the accuracy and latency requirements for IDS and IPS.

Figure 3

Proposed Unified Framework Classification Accuracy

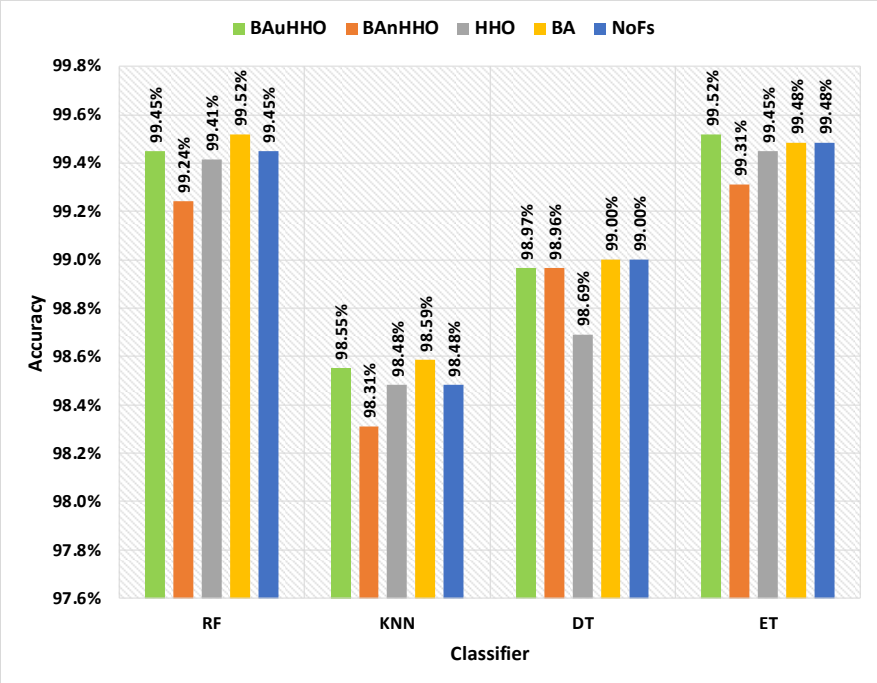


Table 8

Pairwise Accuracy Differences across Classifiers

Method	BA ∪ HHO	BA ∩ HHO	HHO	BA	NoFs
RF					
BA ∪ HHO	0.00	0.21	0.04	−0.07	0.00
BA ∩ HHO	−0.21	0.00	−0.17	−0.28	−0.21
HHO	−0.04	0.17	0.00	−0.11	−0.04
BA	0.07	0.28	0.11	0.00	0.07
NoFs	0.00	0.21	0.04	−0.07	0.00

(continued)

Method	BA ∪ HHO	BA ∩ HHO	HHO	BA	NoFs
KNN					
BA ∪ HHO	0.00	0.24	0.07	−0.04	0.07
BA ∩ HHO	−0.24	0.00	−0.17	−0.28	−0.17
HHO	−0.07	0.17	0.00	−0.11	0.00
BA	0.04	0.28	0.11	0.00	0.11
NoFs	−0.07	0.17	0.00	−0.11	0.00
DT					
BA ∪ HHO	0.00	0.01	0.28	−0.03	−0.03
BA ∩ HHO	−0.01	0.00	0.27	−0.04	−0.04
HHO	−0.28	−0.27	0.00	−0.31	−0.31
BA	0.03	0.04	0.31	0.00	0.00
NoFs	0.03	0.04	0.31	0.00	0.00
ET					
BA ∪ HHO	0.00	0.21	0.07	0.04	0.04
BA ∩ HHO	−0.21	0.00	−0.14	−0.17	−0.17
HHO	−0.07	0.14	0.00	−0.03	−0.03
BA	−0.04	0.17	0.03	0.00	0.00
NoFs	−0.04	0.17	0.03	0.00	0.00

Figure 4 presents recall for RF, KNN, DT, and ET across five feature selection settings: HHO ∪ BA, HHO ∩ BA, HHO, BA, and NoFs. The models achieve consistently high recall. For ET, the HHO ∪ BA, HHO, and NoFs configurations share the highest recall at 99.55%, exceeding BA (99.48%) by 0.07% and surpassing the intersection by 0.30%. For KNN, HHO and BA are tied at 98.29%, improving over the intersection (98.21%) by 0.08% and over the union and NoFs (both 98.15%) by 0.14%; the union and NoFs record the lowest values. For RF, HHO achieves the highest recall at 99.55%, edging BA and NoFs (both 99.48%) by 0.07% and the union (99.40%) by 0.15%, whereas the intersection is lowest at 99.25%. For DT, BA and NoFs share the best recall at 98.74%, slightly above the intersection at 98.66% and the union at 98.52%, with HHO lowest at 98.22% (a 0.52% gap to the best). Overall, the leading configuration varies by model. ET peaks with union/HHO/NoFs, KNN with HHO/BA, RF with HHO, and DT with BA/NoFs. By contrast, the intersection tends to yield the lowest or near-lowest recall, consistent with its aggressive dimensionality reduction. High recall here indicates strong sensitivity to malware and a low rate of missed detections.

Figure 4

Proposed Unified Framework Classification Recall

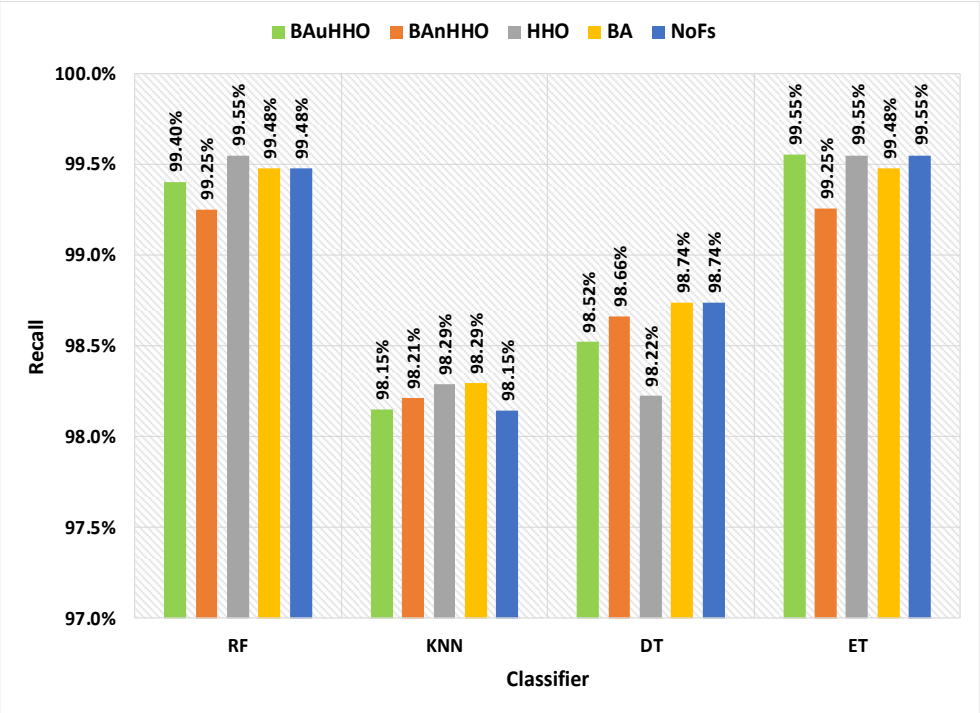


Figure 5 presents precision for RF, KNN, DT, and ET across five feature selection settings: HHO ∪ BA, HHO ∩ BA, HHO, BA, and NoFs. The models deliver consistently high precision. For RF, BA achieves the highest precision at 99.48%, exceeding BA∪HHO at 99.40% by 0.08%, NoFs at 99.33% by 0.15%, HHO at 99.18% by 0.30%, and the intersection at 99.11% by 0.37%. For KNN, BA ∪ HHO leads at 98.73%, narrowly ahead of BA at 98.66% by 0.07%, and above HHO at 98.44% by 0.29%, NoFs at 98.58% by 0.15%, and the intersection at 98.14% by 0.59%. For DT, the union setting attains the best precision at 99.26%, slightly above BA, BAnHHO, and NoFs (all 99.11%) by 0.15%, whereas HHO records the lowest value at 98.96% with a 0.30% gap to the best. For ET, the union and BA settings are tied for best at 99.40%, surpassing NoFs at 99.33% by 0.07% and HHO and the intersection at 99.25% by 0.15%. Overall, the union strategy is strongest for KNN and DT and equally effective for ET, whereas BA is superior for RF and shares the top position for ET; the intersection yields the lowest or near-lowest precision, consistent with its compact feature subset. High precision here indicates a low false positive rate, which is particularly important for IPS-oriented deployment.

Figure 5

Proposed Unified Framework Classification Precision

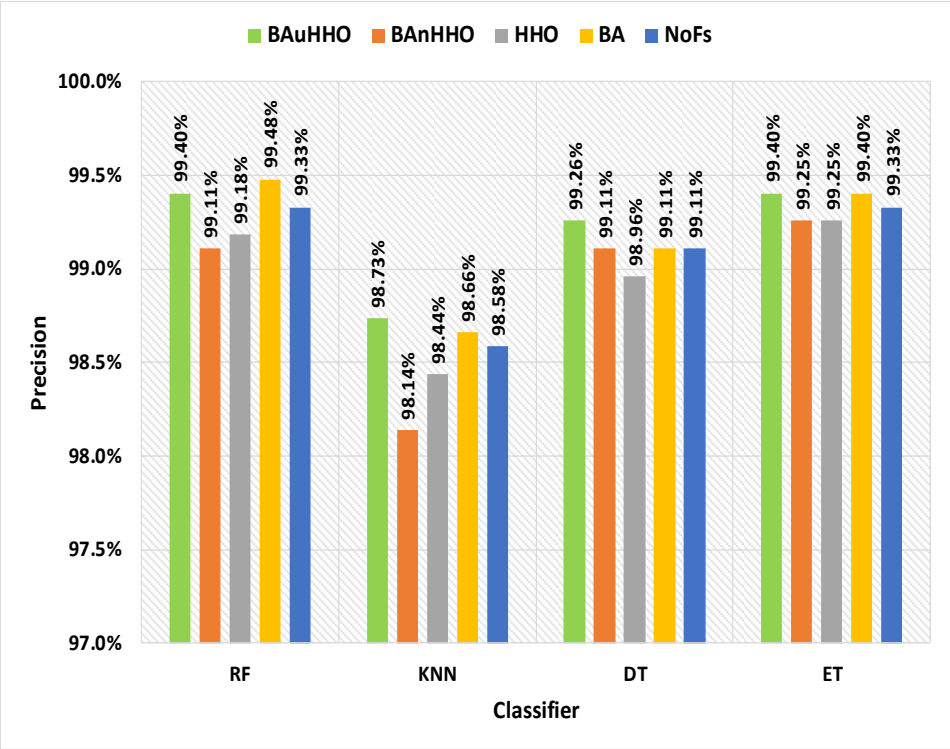
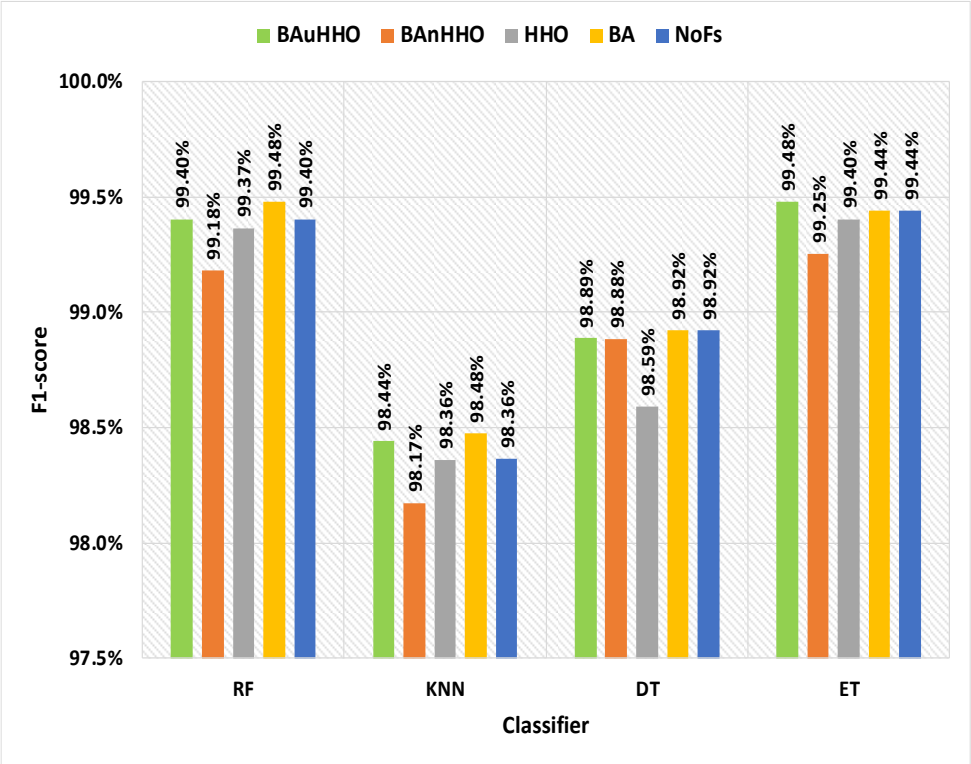


Figure 6 presents the F1 score for RF, KNN, DT, and ET across five feature selection settings: HHO ∪ BA (union), HHO ∩ BA, HHO, BA, and NoFs. The models achieve consistently high F1 values. For ET, the union setting attains the highest F1 at 99.48%, exceeding BA and NoFs at 99.44%, HHO at 99.40%, and the intersection at 99.25%, with a maximum gap of 0.23% relative to union. For KNN, BA is best at 98.48%, narrowly ahead of union at 98.44%, improving over HHO and NoFs (both 98.36%) by 0.12%, whereas the intersection yields the lowest value at 98.17%. For RF, BA achieves the top F1 at 99.48%, ahead of union and NoFs at 99.40% by 0.08%, HHO at 99.37% by 0.11%, and the intersection at 99.18% by 0.30%. For DT, BA and NoFs lead at 98.92%, slightly above union at 98.89% and intersection at 98.88%, with HHO lowest at 98.59% representing a 0.33% gap to the best. Overall, union is best for ET and remains competitive elsewhere, BA is strongest for RF and KNN and co-best for DT, and the intersection is the lowest, consistent with its compact feature subset. High F1 in this context indicates that precision and recall are both strong, yielding balanced detection with minimal trade-offs.

Figure 6

Proposed Unified Framework Classification F1-Score



Comparison with Existing Approaches

Figure 7 reports accuracy on the ISCX-URL2016 dataset for the proposed variants and prior baselines evaluated on the same dataset. Among our methods, RF-BA and ET-Union attain the highest accuracy at 99.52%, followed by ET-NoFs (99.48%), ET-HHO (99.45%), and ET-Intersection (99.31%). All proposed variants exceed the strongest published reference. Using RF-BA as a reference point, the margins over prior work are +0.52% versus Anne and CarolinJeeva (2021)-RF 99.00%, +0.62% versus Okunnuga (2023) 98.90%, +0.63% versus Anne and CarolinJeeva (2021) 98.89%, +1.52% versus Anne and CarolinJeeva (2021)-LGBM 98.00%, +3.02% versus Swetha et al. (2024) 96.50%, and +5.52% versus Anne and CarolinJeeva (2021)-XGBoost 94.00% and Anne and CarolinJeeva (2021)-GradBoost 94.00%. Even the most compact variant, ET-Intersection (99.31%), surpasses the strongest baseline Anne and CarolinJeeva (2021)-RF by 0.31%.

Figure 7

Accuracy of Proposed versus Existing Models on ISCX-URL2016

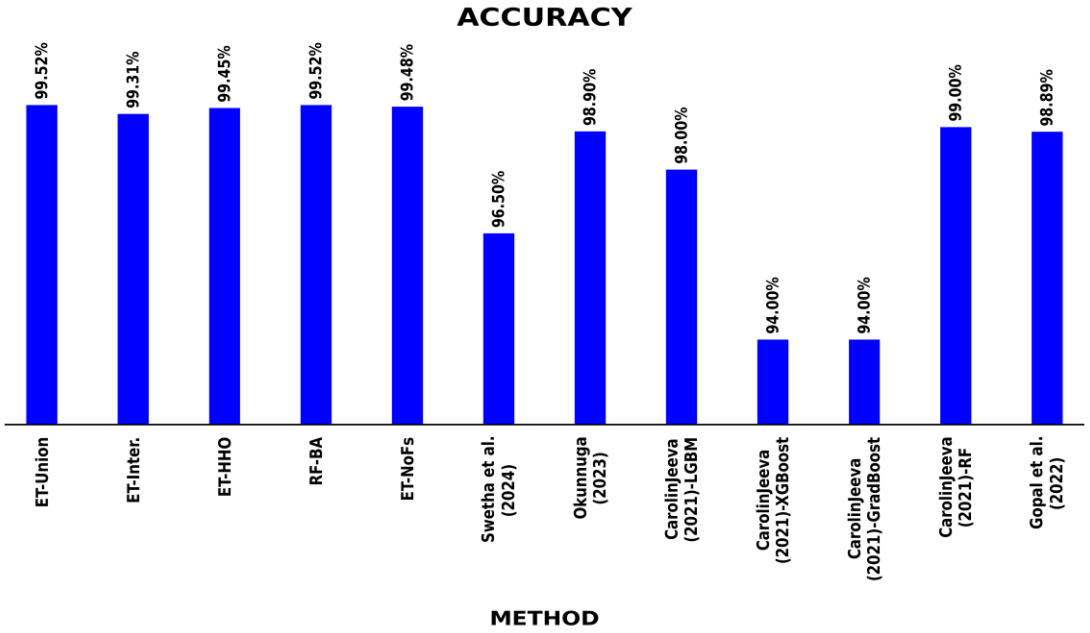


Figure 8 presents an accuracy comparison between our variants and prior studies evaluated on different datasets. Among the proposed methods, RF-BA and ET-Union attain 99.52%, followed by ET-NoFs (99.48%), ET-HHO (99.45%), and ET-Intersection (99.31%). All proposed variants exceed external references: taking RF-BA as the anchor, the margins are +1.12% over Lee et al. (2021)-Tuned MLP (98.40%), +2.45% over Le et al. (2018) (97.07%), +2.52% over Lee et al. (2021)-IBk (97.00%), +2.36% over Hasan et al. (2025)-PCA-LGBM (97.16%), +2.72% over Hasan et al. (2025)-LDA-LGBM (96.80%), and +5.27% over Gu et al. (2024) (94.25%). Even the most compact variant, ET-Intersection (99.31%), surpasses the strongest reference (98.40%) by 0.91%. Cross-study comparisons across different datasets demonstrate comparable margins, reinforcing the general effectiveness of the proposed bio-inspired feature selection within our integrated IDS/IPS framework.

Tables 9 and 10 present the training and per-sample inference times for RF, KNN, DT, and ET under five feature selection settings (NoFs, HHO ∪ BA, HHO ∩ BA, BA, and HHO). Across classifiers, the intersection subset (HHO ∩ BA) yields the minimal computational expense: RF training decreases from 2.139 s to 0.687 s (−67.9%) and inference from 0.010 s to 0.007 s (−30%); DT training reduces from 0.424 s to 0.030 s (−92.9%) and inference from 0.002 s to 0.001 s (−50%); KNN inference diminishes from 0.004 s to 0.002 s (−50%); and ET training declines from 0.913 s to 0.503 s (−44.9%) with inference at 0.009–0.010 s (−10%). BA has the next-best latency profile and, as previously noted, is the leader in accuracy for RF; for ET, latency improves with compact subsets while accuracy peaks with the union (HHO ∪ BA). These patterns support deployment choices: IPS should favor HHO ∩ BA (or BA where intersection ties) to meet strict latency budgets, whereas IDS should prefer RF with BA or ET with HHO ∪ BA when modest latency is acceptable in exchange for enhanced accuracy. In throughput terms, reducing RF inference from 10 ms to 7 ms increases per-core capacity from ~100 to ~143 URLs/s, with analogous gains for KNN and DT. Table 11 presents the best-performing feature selection methods and their corresponding training/inference times and reductions over NoFs.

Although compact subsets such as HHO ∩ BA yield the lowest training time, they may slightly reduce accuracy due to feature reduction. In contrast, richer subsets (e.g., BA or HHO ∪ BA) improve detection performance at a moderate computational cost. Therefore, the proposed framework balances accuracy and efficiency by assigning larger feature subsets to accuracy-oriented IDS and compact subsets to latency-sensitive IPS, enabling effective real-time deployment without sacrificing detection reliability.

Figure 8

Accuracy of Proposed Versus Existing Models across Various Datasets

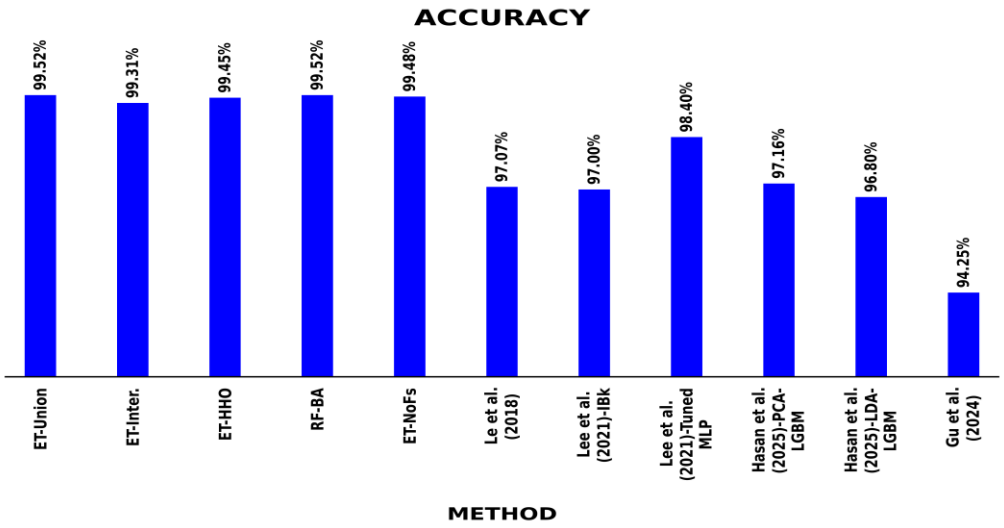


Table 9

Training Time

Classifier	NoFs	HHO ∪ BA	HHO ∩ BA	BA	HHO
RF	2.139	1.795	0.687	1.168	1.500
KNN	0.011	0.007	0.023	0.006	0.003
DT	0.424	0.179	0.030	0.098	0.100
ET	0.913	0.906	0.503	0.664	0.812

Table 10

Inference Time

Classifier	NoFs	HHO ∪ BA	HHO ∩ BA	BA	HHO
RF	0.010	0.010	0.007	0.009	0.009
KNN	0.004	0.003	0.002	0.003	0.003
DT	0.002	0.001	0.001	0.001	0.001
ET	0.010	0.010	0.009	0.009	0.010

Table 11

Summary of Latency Results by Classifier (Best Method, Time, and Reduction vs. NoFs)

Classifier	Best training time	Reduction vs NoFs	Best inference time	Reduction vs. NoFs	Deployment note
RF	0.687 s with HHO ∩ BA	−67.9%	0.007 s with HHO ∩ BA	−30.0%	Intersection is fastest; BA is close for inference and is the RF accuracy leader (IDS).
KNN	0.003 s with HHO	−72.7%	0.002 s with HHO ∩ BA	−50.0%	Ultra-fast; intersection gives the lowest latency.
DT	0.030 s with HHO ∩ BA	−92.9%	0.001 s with HHO ∩ BA / BA / HHO / HHO ∪ BA	−50.0%	1 ms inference floor; training greatly accelerated by intersection.
ET	0.503 s with HHO ∩ BA	−44.9%	0.009 s with HHO ∩ BA / BA	−10.0%	Compact subsets lower latency; accuracy tends to peak with union for ET.

DISCUSSIONS

The proposed framework achieves near-optimal classification performance with respect to accuracy, precision, recall, and F1-score. The highest accuracies are achieved by RF with BA (99.52%) and ET with the union subset HHO ∪ BA (99.52%). KNN reaches 98.59% with BA and 98.55% with HHO ∪ BA, while DT reaches 99.00% with BA, tied with NoFS at 99.00%. Generally, the intersection subset is the smallest but remains very close to the leading configurations, e.g., 99.31% for ET and 99.24% for RF. The same trend continues for precision, recall, and F1-score, indicating that the mean separability between malware and benign URLs remains stable.

In terms of latency, using a smaller number of features is beneficial. The mean training time and mean per-sample inference time for HHO ∩ BA are reduced by approximately 64% and 27%, respectively, on average across the classifiers compared to no feature selection. Notable improvements include RF training time decreasing from 2.139 s to 0.687 s and inference time from 0.010 s to 0.007 s, DT training time decreasing from 0.424 s to 0.030 s and inference time from 0.002 s to 0.001 s, and KNN inference time decreasing from 0.004 s to 0.002 s. The second-best latency profile is achieved with BA, which maintains the best accuracy for RF; HHO ∪ BA provides the best ET accuracy, with only a slight increase in time. These findings correspond directly to the integrated IDS and IPS architecture:

- IDS configuration (accuracy first, out of band): Favor RF with BA or ET with HHO∪BA. These configurations consistently achieve the highest or joint-highest accuracy and F1 score while maintaining an appropriate inference time for mirrored or offline analysis. Absolute accuracy improvements of 0.2 to 0.5 percentage points translate to roughly 2,000 to 5,000 additional correct decisions per one million URLs, reducing false negatives and false positives and alleviating the subsequent triage workload.

- IPS configuration (latency first, inline): Prefer HHO ∩ BA with DT or KNN when very low per sample latency is required; RF with HHO ∩ BA or ET with BA or HHO ∩ BA is appropriate when a slight increase in latency is permissible for enhanced ensemble robustness. The observed latency reductions of about 30 to 50 percent compared with the absence of feature selection (for example, KNN from 4 ms to 2 ms, DT from 2 ms to 1 ms, and RF from 10 ms to 7 ms) increase per core throughput by 40–100 percent and facilitate adherence to strict inline decision time budgets.

CONCLUSION

This study presented a cohesive ML framework for malware-laden URL detection that jointly addresses intrusion detection (accuracy first, out of band) and intrusion prevention (latency first, inline) via bio-inspired feature selection. The framework achieved near-optimal performance by utilizing HHO and BA, both individually and via union and intersection subsets across RF, KNN, DT, and ET, achieving maximum accuracies of 99.52% for RF with BA and 99.52% for ET with the union subset. Compact consensus subsets significantly enhanced efficiency, reducing average training time by 64% and per-sample inference time by about 27% compared with the absence of feature selection, with notable latency reductions from 10 ms to 7 ms for RF, 4 ms to 2 ms for KNN, and 2 ms to 1 ms for DT. These improvements result in significantly fewer errors at scale and high throughput within fixed computational constraints. The results yield clear deployment recommendations within a single architecture: RF with BA or ET utilizing the union subset for IDS when accuracy is paramount, and DT or KNN employing the intersection subset for IPS under strict latency requirements, with BA or intersection-based ensembles serving as robust alternatives when moderate latency is acceptable. Overall, aligning feature selection with module objectives facilitates an effective equilibrium between accuracy and speed in integrated intrusion detection and prevention.

ACKNOWLEDGMENT

The authors would like to acknowledge Al-Ahliyya Amman University (AAU) for its support and facilitation of this research. This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

REFERENCES

Abu Al-Haija, Q., & Al-Fayoumi, M. (2023). An intelligent identification and classification system for malicious uniform resource locators (URLs). *Neural Computing and Applications, 35*(23), 16995–17011. https://doi.org/10.1007/s00521-023-08592-z

Abualhaj, M. M., & Al-Khatib, S. N. (2024). Using decision tree classifier to detect Trojan Horse based on memory data. *TELKOMNIKA (Telecommunication Computing Electronics and Control), 22*(2), 393–400. https://doi.org/10.12928/telkomnika.v22i2.25753

Abualhaj, M. M., Abu-Shareha, A. A., Alkhatib, S. N., Shambour, Q. Y., & Alsaaidah, A. M. (2025). Detecting spam using Harris Hawks optimizer as a feature selection algorithm. *Bulletin of Electrical Engineering and Informatics, 14*(3), 2361–2369. https://doi.org/10.11591/eei.v14i3.9198

Abualhaj, M. M., Abu-Shareha, A. A., Shambour, Q. Y., Al-Khatib, S. N., & Hiari, M. O. (2024). Tuning the K value in K-nearest neighbors for malware detection. *IAES International Journal of Artificial Intelligence (IJ-AI), 13*(2), 2275–2282. https://doi.org/10.11591/ijai.v13.i2.pp2275-2282

Abualhaj, M. M., Al-Khatib, S. N., Al Zyoud, M., Qaddara, I., & Anbar, M. (2025). Enhancing intrusion detection system performance using a hybrid of Harris Hawks and Whale Optimization algorithms. *Engineering, Technology & Applied Science Research, 15*(4), 24354–24361. https://doi.org/10.48084/etasr.10919

Abualhaj, M. M., Al-Shamayleh, A. S., Munther, A., Alkhatib, S. N., Hiari, M. O., & Anbar, M. (2024). Enhancing spyware detection by utilizing decision trees with hyperparameter optimization. *Bulletin of Electrical Engineering and Informatics, 13*(5), 3653–3662. https://doi.org/10.11591/eei.v13i5.7939

Abualhaj, M. M., Hiari, M. O., Alsaaidah, A., & Al-Zyoud, M. M. (2025). Comparative analysis of whale and Harris Hawks optimization for feature selection in intrusion detection. *Indonesian Journal of Electrical Engineering and Computer Science, 37*(1), 179–185. https://doi.org/10.11591/ijeecs.v37.i1.pp179-185

Abualhaj, M. M., Shambour, Q. Y., Abu-Shareha, A. A., Al-Khatib, S. N., & Amer, A. (2025). Enhancing malware detection through self-union feature selection using gray wolf optimizer. *Indonesian Journal of Electrical Engineering and Computer Science, 37*(1), 197–205. https://doi.org/10.11591/ijeecs.v37.i1.pp197-205

Abu-Shareha, A. A. (2024). A Framework for Diabetes Detection Using Machine Learning and Data Preprocessing. *Journal of Applied Data Sciences*, *5*(4), 1654–1667. https://doi.org/10.47738/jads.v5i4.363

Afshar, F., Seyedabrishami, S., & Moridpour, S. (2022). Application of extremely randomised trees for exploring influential factors on variant crash severity data. *Scientific Reports, 12*, Article 11476. https://doi.org/10.1038/s41598-022-15693-7

Al Saaidah, A., Abualhaj, M. M., Shambour, Q. Y., Abu-Shareha, A. A., Abualigah, L., Al-Khatib, S. N., & Alraba'nah, Y. H. (2024). Enhancing malware detection performance: Leveraging K-nearest neighbors with Firefly Optimization Algorithm. *Multimedia Tools and Applications, 84*, 10071–10094. https://doi.org/10.1007/s11042-024-18914-5

Al-Dabbas, L., & Abu-Shareha, A. A. (2024). Early detection of female type-2 diabetes using machine learning and oversampling techniques. *Journal of Applied Data Sciences, 5*(3), 1237–1245. https://doi.org/10.47738/jads.v5i3.298

Almomani, A., Akour, I., Manasrah, A., Almomani, O., Alauthman, M., Abdullah, E., Shwait, A., & Sharaa, R. (2023). Ensemble-Based Approach for Efficient Intrusion Detection in Network Traffic. *Intelligent Automation & Soft Computing*, *37*(2), 2499–2517. https://doi.org/10.32604/iasc.2023.039687

Almomani, O., Alsaaidah, A., Abu-Shareha, A. A., Alzaqebah, A., Almaiah, M. A., & Shambour, Q. (2025). Enhance URL defacement attack detection using particle swarm optimization and machine learning. *Journal of Computational and Cognitive Engineering, 4*(3), 296–308. https://doi.org/10.47852/bonviewJCCE52024668

Alraba'nah, Y., Al-Sharaeh, S., & Al Hindi, G. (2025). Enhancing Intrusion Detection Using Hybrid Long Short-Term Memory and XGBoost. *Journal of Soft Computing and Data Mining*, *6*(1), 247–261.

Alraba'nah, Y., & Toghuj, W. (2024). A deep learning based architecture for malaria parasite detection. *Bulletin of Electrical Engineering and Informatics*, *13*(1), 292–299. https://doi.org/10.11591/eei.v13i1.5485

AlTalhi, R., Saqib, M. N., Saeed, U., & Alghamdi, A. S. (2021). *Malicious URL detection using streaming feature selection.* In Proceedings of the 5th International Conference on Future Networks & Distributed Systems (pp. 100–104). ACM. https://doi.org/10.1145/3508072.3508088

Anne, W. R., & CarolinJeeva, S. (2021). *Performance analysis of boosting techniques for classification and detection of malicious websites*. Proceedings of the International Conference on Combinatorial and Optimization (ICCAP), 405–415.

Fan, C., Chen, M., Wang, X., Wang, J., & Huang, B. (2021). A review on data preprocessing techniques toward efficient and reliable knowledge discovery from building operational data. *Frontiers in Energy Research*, *9*, Article 652801. https://doi.org/10.3389/fenrg.2021.652801

Gopal, S. B., Poongodi, C., Nanthiya, D., Kirubakaran, T., Logeshwar, D., & Saravanan, B. K. (2022). *Autoencoder based architecture for mitigating phishing URL attack in the internet of things (IoT) using deep neural networks*. Proceedings of the International Conference on Devices, Circuits and Systems (ICDCS), 427–431. IEEE.10.1109/ICDCS54290.2022.9780673

Gu, R., Fei, J., Wang, D., Zhu, Y., Yu, H. & Guo, F. (2024). *A malicious encrypted traffic detection method based on hybrid feature selection.* Proceedings of Fifth International Conference on Computer Communication and Network Security (CCNS 2024), 13228, 517-526. https://doi.org/10.1117/12.3038339

Hamdan Mohammad, A., Alwada'n, T., Almomani, O., Smadi, S., & ElOmari, N. (2022). Bio-inspired Hybrid Feature Selection Model for Intrusion Detection. *Computers, Materials & Continua*, *73*(1), 133–150. https://doi.org/10.32604/cmc.2022.027475

Hasan, R., Biswas, B., Samiun, M., Saleh, M. A., Prabha, M., Akter, J., Joya, F. H., & Abdullah, M. (2025). Enhancing malware detection with feature selection and scaling techniques using machine learning models. *Scientific Reports*, *15*(1), 9122. https://doi.org/10.1038/s41598-025-93447-x

Heidar, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris hawks optimization: Algorithm and applications. *Future Generation Computer Systems*, *97*, 849–872. https://doi.org/10.1016/j.future.2019.02.028

Hersyaputra, M. S., Jaya, M. T. T., & Anggraini, R. N. E. (2026). A hybrid machine learning model for streaming frequency-based anomaly detection in banking transactions. *Journal of Information and Communication Technology*, *25*(1), 64-78. https://doi.org/10.32890/jict2026.25.1.4

Hoang, X. D., Nguyen, B. C., & Thu, T. (2023). *Detecting malware based on statistics and machine learning using opcode N-Grams.* 2023 RIVF International Conference on Computing and Communication Technologyies (RIVF). https://doi.org/10.1109/rivf60135.2023.10471824

Hussain, A. S., Pati, K. D., Atiyah, A. K., & Tashtoush, M. A. (2025). Rate of occurrence estimation in geometric processes with Maxwell distribution: A comparative study between artificial intelligence and classical methods. *International Journal of Advances in Soft Computing and Its Applications*, *17*(1), 1–15. https://doi.org/10.15849/ijasca.250330.01

Kocev, D., Geurts, P., Wehenkel, L., & Džeroski, J. (2020). Ensembles of extremely randomized predictive clustering trees. *Machine Learning*, *101*(1-3), 139–172. https://doi.org/10.1007/s10994-020-05894-4

Kumi, S., Lim, C., & Lee, S.-G. (2021). Malicious URL detection based on associative classification. *Entropy*, *23*(2), 182. https://doi.org/10.3390/e23020182

Le, H., Pham, Q., Sahoo, D., & Hoi, S. C. H. (2018). *URLNet: Learning a URL representation with deep learning for malicious URL detection*. arXiv. https://doi.org/10.48550/arXiv.1802.03162

Lee, J., Jang, H., Ha, S., & Yoon, Y. (2021). Android malware detection using machine learning with feature selection based on the genetic algorithm. *Mathematics*, *9*(21), 2813. https://doi.org/10.3390/math9212813

Liu, Z., Chang, B., & Cheng, F. (2021). An interactive filter-wrapper multi-objective evolutionary algorithm for feature selection. *Swarm and Evolutionary Computation*, *65*, 100925. https://doi.org/10.1016/j.swevo.2021.100925

Mamun, M. S. I., Rathore, M. A., Lashkari, A. H., Stakhanova, N., & Ghorbani, A. A. (2016). *Detecting malicious URLs using lexical analysis*. In J. Chen, V. Piuri, C. Su, & M. Yung (Eds.), Network and system security (Lecture Notes in Computer Science, Vol. 9955). Springer. https://doi.org/10.1007/978-3-319-46298-1_30

Okunnuga, I. D. (2023). Prediction and detection of malicious URL using machine learning. *International Journal of Advance Research, Ideas and Innovations in Technology, 10*(1), 195-219.

Pethe, Y. S., Gourisaria, M. K., Singh, P. K., & Das, H. (2024). FSBOA: Feature selection using bat optimization algorithm for software fault detection. *Discover Internet of Things*, *4*(1). https://doi.org/10.1007/s43926-024-00059-4

Potharlanka, J. L., & M, N. B. (2024). Feature importance feedback with Deep Q process in ensemble-based metaheuristic feature selection algorithms. *Scientific Reports*, *14*(1), 2923. https://doi.org/10.1038/s41598-024-53141-w

Qaddara, I., & Alraba'nah, Y. (2025). Enhancing requirements classification using machine learning techniques. *SN Computer Science*, *6*(6). https://doi.org/10.1007/s42979-025-04158-z

Rafrastara, F. A., Ghozi, W., Sani, R. R., Handoko, L. B., Abdussalam, Pramudya, E. R., & Abdollah, F. M. (2025). Integrating information gain and chi-square for enhanced malware detection performance. *Journal of Information and Communication Technology*, *24*(1), 79-101. https://doi.org/10.32890/jict 2025.24.1.4

Sagagi, I. T., Adamu, A., Abdulrahman, S. M., & Zainon, W. M. N. (2024). *Unsupervised feature selection technique with enhanced binary bat algorithm.* In 2024 IEEE 5th International Conference on Electro-Computing Technologies for Humanity (NIGERCON). IEEE. https://doi.org/10.1109/NIGERCON62786.2024.10926980

Saheed, Y. K., Kehinde, T. O., Ayobami Raji, M., & Baba, U. A. (2023). Feature selection in intrusion detection systems: A new hybrid fusion of bat algorithm and Residue Number System. *Journal of Information and Telecommunication, 8*(2), 189-207. https://doi.org/10.1080/24751839.2023.2272484

Scarfone, K., & Mell, P. (2007). *Guide to Intrusion Detection and Prevention Systems (IDPS)* (NIST Special Publication 800-94). National Institute of Standards and Technology https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=901146

Shambour, Q., Qandeel, N., Alrabanah, Y., Abumariam, A., & Shambour, M. K. (2024). Artificial intelligence techniques for early autism detection in toddlers: A comparative analysis. *Journal of Applied Data Sciences*, *5*(4), 1754-1764.

Sheikh, A. M., Islam, M. R., Habaebi, M. H., Zabidi, S. A., Rahman, A., & Kabbani, A. (2025). A survey on edge computing (EC) security challenges: Classification, threats, and mitigation strategies. *Future Internet*, *17*(4), 175–175. https://doi.org/10.3390/fi17040175

Spooner, A., Mohammadi, G., Sachdev, P. S., Brodaty, H., & Arcot Sowmya. (2023). Ensemble feature selection with data-driven thresholding for Alzheimer's disease biomarker discovery. *BMC Bioinformatics*, *24*(1). https://doi.org/10.1186/s12859-022-05132-9

Swetha, T., Seshaiah, M., Hemalatha, K. L., Murthy, S. V. N., & Manjunatha Kumar, B. H. (2024). Hybrid machine learning approach for real-time malicious URL detection using SOM-RMO and RBFN with tabu search. *International Journal of Advanced Computer Science and Applications, 15*(8), 450–458. https://doi.org/10.14569/ijacsa.2024.0150844

Tripathy, B. K., Maddikunta, P. K. R., Pham, Q.-V., Gadekallu, T. R., Dev, K., Pandya, S., & ElHalawany, B. M. (2022). Harris Hawk optimization: A survey on variants and applications. *Computational Intelligence and Neuroscience, 2022*, Article 2218594. https://doi.org/10.1155/2022/2218594

Tung, S. P., Wong, K. Y., Kuzminykh, I., Bakhshi, T., & Ghita, B. (2022). Using a machine learning model for malicious URL type detection. *Lecture Notes in Computer Science*, 493–505. https://doi.org/10.1007/978-3-030-97777-1_41

Yab, L. Y., Wahid, N., & Hamid, R. A. (2024). Impact of balanced exploration and exploitation on high-dimensional feature selection with hierarchical whale optimisation algorithm. *Journal of Information and Communication Technology*, *23*(4), 593-626. https://doi.org/10.32890/jict2024.23.4.2

Yeop, N. S., Zakaria, N. H., & Suendri (2025). Enhancing the effectiveness of machine learning-based phishing email detection via an improved pre-processing technique for data security. *Journal of Information and Communication Technology*, *24*(4), 87-110. https://doi.org/10.32890/jict2025.24.4.4

Zhang, X., & Jonassen, I. (2019). *An ensemble feature selection framework integrating stability*. 2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM). https://doi.org/10.1109/bibm47256.2019.8983310