International Journal of Safety and Security Engineering
Vol. 16, No. 9, September, 2026, pp. 2153-2166
Journal homepage: http://iieta.org/journals/ijsse

Payload-Oriented Intrusion Detection Through Packet-Flow Association and Cryptography-Driven Statistical Features

Nadia K. Al-Hasani, Farah Al-Shareefi, Hadeel Q. Gheni*

Department of Computer Science, College of Science for Women, University of Babylon, Babil 51001, Iraq

Corresponding Author Email: wsci.hadeel.qasem@uobabylon.edu.iq

Copyright: ©2026 The authors. This article is published by IIETA and is licensed under the CC BY 4.0 license (http://creativecommons.org/licenses/by/4.0/).

https://doi.org/10.18280/ijsse.160918

Received: 3 July 2026
Revised: 13 September 2026
Accepted: 20 September 2026
Available online: 30 September 2026

Keywords:
intrusion detection system, statistical payload modeling, entropy analysis, chi-square analysis, bit-balance analysis, temporal flow indexing

ABSTRACT

Intrusion detection systems are essential for protecting network infrastructures against increasingly sophisticated cyberattacks. Although machine learning-based intrusion detection systems have been extensively investigated, most existing approaches rely primarily on conventional flow-level features, such as packet size and flow duration, which may inadequately characterize the statistical properties of packet payloads, particularly in encrypted network traffic. To address this limitation, a payload-oriented intrusion detection framework was proposed in which conventional flow-level features were integrated with cryptography-inspired statistical features extracted from packet payloads. First, individual packets were associated with their corresponding flows through a temporal hash-based indexing strategy and bidirectional flow keys, thereby reducing redundant comparisons during packet-to-flow association. Subsequently, three payload-level features, namely Shannon entropy, the chi-square statistic, and bit balance, were extracted. For each flow, the extracted features were aggregated using the mean, standard deviation, maximum, and minimum to construct a compact flow-level statistical representation. The resulting payload-derived features were then combined with conventional flow-level features to generate a unified feature representation for supervised intrusion detection. The proposed framework was evaluated under binary and three-class classification settings using five machine learning classifiers: logistic regression, extra trees, adaptive boosting, extreme gradient boosting, and a multilayer perceptron. Furthermore, Shapley Additive Explanations (SHAP) analysis was employed to interpret model predictions and assess the contributions of individual features to classification decisions. Experimental results indicated that the integration of payload-derived statistical features improved intrusion detection performance compared with conventional flow-level representations across the evaluated classification settings. The SHAP analysis further identified discriminative contributions from the augmented features, providing additional insight into their influence on model predictions. These findings demonstrate the potential of integrating packet-level payload statistics with flow-level traffic characteristics to enhance machine learning-based intrusion detection and provide an interpretable framework for network traffic analysis.

1. INTRODUCTION

The rapid proliferation of networked environments, combined with the increasing sophistication of cyberattacks, has made intrusion detection systems a vital component of modern cybersecurity infrastructures [1]. Over the past few years, a lot of focus has been directed towards the machine learning-based intrusion detection systems because of their ability to identify known and unknown cyberattacks [2-4]. Most of the existing studies are mainly based on features extracted from the flow level of datasets, in which the analysis is mainly limited to statistical characteristics such as packet count, flow duration, and inter-arrival times, among others [5, 6]. Although these features are very useful in many intrusion detection situations, based on existing research, the statistical properties of packets inherent to their payloads have not been extensively studied. To this end, emerging research has started to investigate the use of payload-level statistical features in anomaly detection; for example, it has been suggested that entropy-based measures have the ability to provide unique information about the normal and anomalous traffic patterns [7]. In other words, entropy is a good measure of payload randomness [8], and hence it can be used to detect encrypted, compressed, or obfuscated traffic.

Inspired by these findings, this study further extends payload-driven analysis by incorporating chi-square deviation [9] and bit balance [7, 9, 10] as integral cryptography-driven features. The chi-square statistic measures the deviations from the payload byte-frequency distributions; bit balance is used to measure the deviations from the structural balance of the payload binary values. These features can be used together to provide a more comprehensive representation of traffic than flow statistics alone.

The ability to extract such features, however, requires direct

access to raw packet capture traffic and not just relying solely on pre-computed comma-separated values (CSV) datasets. Indeed, there are a number of tools for accessing the contents and other information of flows [11, 12] but the proposed framework is not limited to packet extraction. Rather, it is the problem of building a scalable framework of associations between packets and flows, optimized for payload-based intrusion detection. Therefore, this study introduces a temporal hashing technique to reduce the amount of redundant comparison processes and to enable direct aggregation of statistical payload features within a machine learning-oriented intrusion detection system pipeline.

Consequently, the proposed framework makes contributions at multiple levels. First, a temporal hash-based indexing strategy is introduced for associating packets with flows. Second, it enriches conventional flow feature representation by directly extracting entropy, chi-square deviation, and bit balance from packet payloads. Third, a deterministic packet-to-flow assignment method ensures that each packet is linked to no more than one suitable flow, resolving ambiguities from temporally coinciding flows and prohibiting the multiple allocation of the same packet. Fourth, a session-aware data partitioning approach is introduced, in which related bidirectional flows are grouped into communication sessions prior to the establishment of training, validation, and test sets, thus eliminating the duplication of sessions across data partitions and lessening the likelihood of data leakage. Finally, it assesses the engineered features utilizing different supervised learning models and Shapley Additive Explanations (SHAP) [13] interpretable analysis. The integrated contributions allow a more discriminating and explainable intrusion detection system that is able to include statistical features which are usually neglected in conventional intrusion detection systems.

The remainder of this study is structured as follows. Section 2 critically reviews the relevant literature. Section 3 introduces the proposed framework and details the following stages: flow construction, packet-to-flow association, payload feature derivation, classification, and explainability evaluation. Experimental conditions are described in Section 4 and the results are presented in Section 5. Finally, Section 6 concludes the study and outlines directions for future research.

2. LITERATURE REVIEW

Driven by the rapid advancement of cyber threats, extensive research efforts have been dedicated to machine learning-based intrusion detection systems [1]. Recent studies have emphasized several key directions, including classifier selection, feature engineering, payload-centric analysis, encrypted traffic examination, interpretable artificial intelligence, and robust intrusion detection system design.

Ensemble or hybrid learning approaches employed classifiers such as random forest, extreme gradient boosting, k-nearest neighbors, or adaptive boosting to improve detection accuracy in both binary and multi-class classification schemes [14]. The results showed that classifier selection significantly affected the performance of the intrusion detection system. Similarly, an intrusion detection system based on feature selection and recurrent neural network architectures was proposed [15]. The results asserted the significance of informative feature selection prior to model training. Consistent with this trend, the effectiveness of a long short-term memory-based intrusion detection system was evaluated, and the importance of deep learning for intrusion detection was emphasized [16]. In spite of all these efforts, their main focus was on the effectiveness of the model rather than on the features of the payload used and their quality.

Much of the intrusion detection system research relies on already collected flow-level features on CSV files using automatic tools, e.g., CICFlowMeter [17, 18]. The datasets are widely utilized because of the traffic-structured representation [2]. Generally, the tabular flow-level features include flow lifetime, packet counts, packet timing intervals, protocol-related features, and header-based communication statistics [4, 5]. Although the detection results are encouraging, these representations remain limited in their ability to capture the intrinsic statistical characteristics of packet payloads. Consequently, several recent studies have focused on adding additional attributes to the traditional flow-centric view through the use of packet-level and payload-based traffic analysis methods. Raw packet-byte representations were employed to discover payload information other than the conventional flow-level features [19]. However, raw bytes can lead to higher computational costs and reduced interpretability. The proposed method, on the contrary, employs interpretable statistical payload descriptors. A multimodal intrusion detection system framework that integrates flow-level and payload-level information to enhance the detection capability was proposed [20]. However, the potential of statistical payload indicators like entropy, chi-square deviation and bit balance has not been studied. Entropy-aided detection in Industrial Internet of Things environments was further investigated [21]. This research focused on improving the accuracy of entropy estimation by analyzing how the aggregation method and traffic characteristics affect detection responsiveness. The present work, however, does not aim to optimize the computation of entropy. Rather, it identifies entropy as just one of a number of features derived from the payload. Similarly, an entropy-based analysis showed that payload entropy, as analyzed against a traffic baseline, is more informative than when used as an individual anomaly indicator [7]. This observation provides an incentive for its use in this work as an additional statistical characteristic.

Ciphered traffic analysis without decrypting payload data was scrutinized based on traffic-related discernible statistical and behavioral signatures [22]. In addition, sequential representations of encrypted traffic were considered to uncover hidden packet-level behaviors [23]. The findings of both works recommended that valuable security information can still be obtained from encrypted traffic via statistical inference. However, this was achieved at the expense of additional modeling complexity. Even though explainable artificial intelligence mechanisms, like SHAP, have been broadly adopted for explaining intrusion detection outcomes [24, 25], their implementation has substantially focused on conventional flow-level features, with little attention paid to hybrid payload-aware ones.

The literature review reveals that several limitations remain inadequately addressed in the intrusion detection system domain. First, a wide range of studies continue to depend mainly on flow-level features without extensively leveraging payload-based statistical behavior. Second, existing payload-centric methods either focus solely on entropy or rely on resource-intensive raw packet representations. Third, existing studies have failed to sufficiently consider the combination of

cryptographically motivated statistical payload features, flow-level features, and interpretable machine-learning models in a scalable packet-to-flow association framework. To date, no solution has been proposed addressing these three problems within a scalability-aware framework.

3. PROPOSED FRAMEWORK

This study proposes an improved traffic representation framework to improve intrusion detection performance within supervised learning settings. This framework consolidates flow-level and packet-level information via a well-organized mechanism for packet-to-flow matching, while incorporating payload-based statistical feature extraction. In contrast to traditional flow-oriented methods that depend entirely on aggregated statistical features, the proposed framework augments flow records by per-packet information, providing a richer representation of network traffic behavior, as illustrated schematically in Figure 1.

The proposed framework consists of five main stages:

Stage 1: Flow construction from original packet capture traffic.

Stage 2: Effective packet-to-flow association based on temporal ordering indexing.

Stage 3: Feature augmentation utilizing payload-inferred statistics in parallel with the traditional ones.

Stage 4: Machine learning-based classification for anomaly detection.

Stage 5: Evaluation of model interpretability using SHAP.

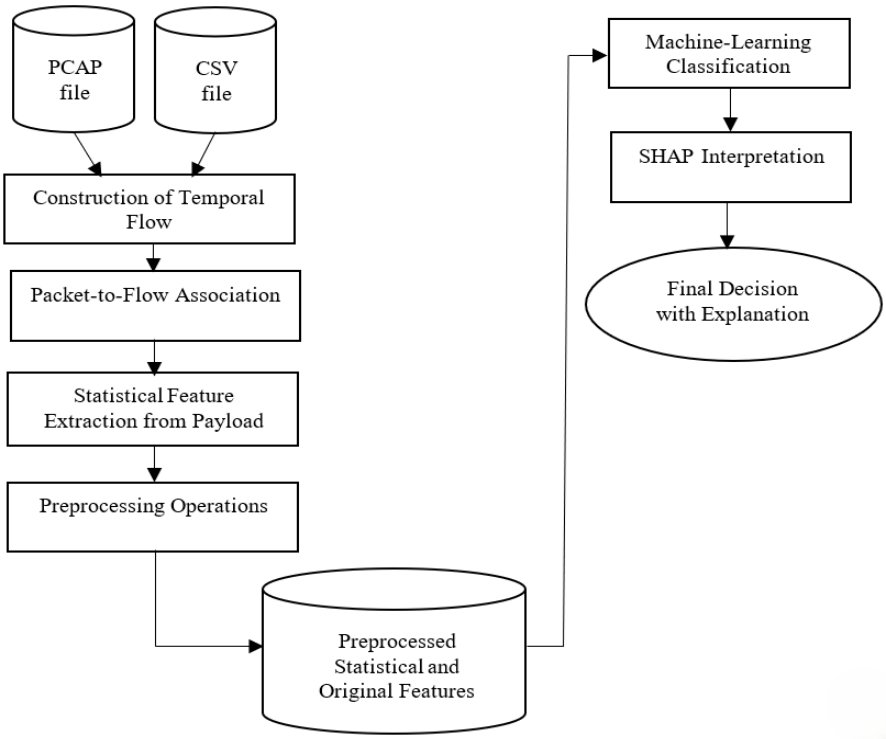


Figure 1. Pipeline of the proposed payload-based intrusion detection framework

The designed framework enables scalable processing of large-scale network traffic captures while ensuring temporal consistency between packets and flows. Although tools, such as Wireshark [11] and TShark [12], enable extensive packet inspection and protocol decoding functionalities, this framework goes beyond packet representation and field derivation. Instead, it deals with effective packet-to-flow association and payload-derived statistical aggregation for forming consolidated flow-based feature representations. The following subsections describe each component in detail.

3.1 Flow construction stage

This stage reconstructs network flows straight from unprocessed packet capture traces. This stage is essential, as all successive stages depend on a rigorous and temporally coherent representation of network flows.

A naïve approach to performing this stage would involve comparing every incoming packet with all present flows. However, if the number of packets is P and the number of flows is N, then a brute-force solution would incur a considerable time complexity represented by $O(P \times N)$. Such computational complexity is impractical for addressing large-scale network traces, particularly when processing millions of packets. To cope with this limitation, this study utilizes a hash-indexed strategy whereby flows are grouped and indexed in accordance with their network keys and session interval. As a result, the search operation is narrowed from scanning the entire flow table to only a limited subset of candidate flows sharing the same key and falling within the same time window. The following formal definitions and algorithms describe the proposed solution in detail.

Definition 1. Let the set of network flows be denoted by $\mathcal{F} = \{\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_N\}$, where each flow $\mathcal{F}_i$ with index i is identified by a 3-tuple, $\left(k_i, t_i^{start}, t_i^{end}\right)$. Here, $k_i = (srcIp_i, dstIp_i, srcPort_i, dstPort_i, proto_i)$ represents the standard 5-tuple key, which consists of the following session components pertinent to F_i: source and destination of Internet Protocol (IP) addresses, source and destination port numbers, and the protocol identifier. Furthermore, t_i^{start} and t_i^{end} represent the start and end times of flow $\mathcal{F}_i$, respectively.

Definition 2. To efficiently organize the flows, the temporal

flow index is specified as a hash structure $\mathcal{H}$. This structure is formulated as a mapping function $\mathcal{H}$: K $\rightarrow \mathcal{L}$, such that K is the set of all viable network keys, and $\mathcal{L} = \bigcup_{k \in K} L_k$, such that L_k is the list of 3-tuple of the form $\left(t_i^{start}, t_i^{end}, i\right)$. Hence, L_k contains all flows associated with the key $k_i \in$ K. In order to enable effective temporal matching, this list is arranged in non-decreasing order in regard to the flow start times, i.e., $t_i^{start} \leq t_j^{start}$ for any two members with indices $i < j$, it holds that $\forall\, i, j \in \{1, \ldots, |L_k|\}, i < j \Longrightarrow t_i^{start} \leq t_j^{start}$.

Definition 3. For each key $k_i \in$ K, a temporal indicator is defined as $\mathfrak{D}$ which denotes the current scanning position within the flow list L_k. The purpose is to eliminate unnecessary rescanning of inactive flows and enabling effective packet-to-flow association under large-scale traffic streams.

Definition 4. Assume that the packet stream is expressed as: $p = \{p_1, p_2, \ldots, p_P\}$, and every packet p_j with index j can be formulated as a 2-tuple: (kp_j, tp_j), where kp_j and tp_j are the key and the timestamp of packet p_j, respectively, and $kp_j = (srcIp_j, dstIp_j, srcPort_j, dstPort_j, proto_j)$. Consequently, we can say that a packet p_j is associated with the flow $\mathcal{F}_i$ if and only if: $k_i = kp_j \; and \; t_i^{start} \leq tp_j \leq t_i^{end}$.

Algorithm 1: Temporal flow index construction

Input:
$\mathcal{F}$ ▷ set of flows
Output:
$\mathcal{H}_{fwd}$, $\mathcal{H}_{bwd}$ ▷ temporal flow hash indices
$\mathfrak{D}^{fwd}$ ▷ set of per − key indicators
Begin
$\mathcal{H}_{fwd} \leftarrow \emptyset$
$\mathcal{H}_{bwd} \leftarrow \emptyset$
$\mathfrak{D} \leftarrow \emptyset$
for each flow $\mathcal{F}_i \in \mathcal{F}$ **do**
$k_{fwd} \leftarrow (srcIp_i, dstIp_i, srcPort_i, dstPort_i, proto_i)$
$k_{bwd} \leftarrow (dstIp_i, srcIp_i, dstPort_i, srcPort_i, proto_i)$
$append(t_i^{start}, t_i^{end}, i)\ to\ \mathcal{H}_{fwd}[k_{fwd}]$
$append(t_i^{start}, t_i^{end}, i)\ to\ \mathcal{H}_{bwd}[k_{bwd}]$
end for
for each key ∈ $\mathcal{H}$ **do**
$L_k^{fwd} \leftarrow \mathcal{H}_{fwd}[k_{fwd}]$
$L_k^{bwd} \leftarrow \mathcal{H}_{bwd}[k_{bwd}]$
sort each L_k by start time in ascending order
$ptr_k^{fwd} \leftarrow 0, ptr_k^{bwd} \leftarrow 0$
$\mathfrak{D}^{fwd}[k] \leftarrow ptr_k^{fwd}, \mathfrak{D}^{bwd}[k] \leftarrow ptr_k^{bwd},$
end for
End.

The procedure for constructing $\mathcal{H}$ is detailed in Algorithm 1. Although the basic definition of this structure is expressed as a single mapping $\mathcal{H}$(k), the implementation promotes it into forward and backward indices to establish reliable processing of bidirectional network traffic. More specifically, each flow $\mathcal{F}_i$ is associated with dual directional keys: $k_{fwd_i} = (srcIp_i, dstIp_i, srcPort_i, dstPort_i, proto_i)$ for forward direction and $k_{bwd_i} = (dstIp_i,\ srcIp_i, srcPort_i, dstPort_i, proto_i)$ with respect to backward one. Therefore, the temporal index $\mathcal{H}$ is reformulated to include two mapping functions: $\mathcal{H}\left[k_{fwd}\right]$ and $\mathcal{H}[k_{bwd}]$. Accordingly, Algorithm 1 initially generates for each flow a forward 5-tuple key and a backward 5-tuple key. Next, it inserts the relevant flow record $(t_i^{start}, t_i^{end}, i)$ into both indices: $\mathcal{H}\left[k_{fwd}\right]$ and $\mathcal{H}[k_{bwd}]$. Later, it sorts each flow list L_k with respect to flow start time. Finally, it initializes a temporal indicator $\mathfrak{D}$ for each key, to enable straightforward access to filtered flow candidates associated with a packet.

Lemma 1. Let each flow $\mathcal{F}_i$ be assigned bidirectional indexing keys: a forward key k_i^{fwd} and a backward key k_i^{bwd} and stored into corresponding hash structure $\mathcal{H}_{fwd}$ and $\mathcal{H}_{bwd}$, respectively. Then, for any packet p_j pertaining to flow $\mathcal{F}_i$, disregarding its direction, there is a direct retrieval path, such that $\mathcal{F}_i$ can be efficiently retrieved from either $\mathcal{H}_{fwd}$ or $\mathcal{H}_{bwd}$ utilizing the packet key.

Proof. As a flow $\mathcal{F}_i$ denotes bidirectional communication between a pair of endpoints, then a packet p_j associated with that flow if and only if the key of p_j is equal to either:

•the forward direction key k_i^{fwd} of $\mathcal{F}_i$, thus this flow is identified through $\mathcal{H}_{fwd}$,

•or the backward direction key k_i^{bwd} of $\mathcal{F}_i$, hence this flow is determined through $\mathcal{H}_{bwd}$.

As a consequence, the correct flow in both cases can be directly accessed without needing any key conversion or reverse lookup.

3.2 Packet-to-flow association

This stage is based on a packet-to-flow association algorithm (Algorithm 2), which is designed to process packets in a chronological order. Typically, it utilizes the hash index constructed in Algorithm 1 to identify the set of flows that remain active upon packet arrival. A key challenge is eliminating redundant traversal of expired or inactive flows. This is addressed by employing a monotonically advancing pointer, also produced by Algorithm 1, to selectively inspect flows whose temporal intervals may intersect the packet timestamp. The following formal definitions reflect that behavior.

Algorithm 2: Packet to active flow association

Input:
$\mathcal{H}^d$ ▷ flow hash index produced by Algorithm 1
$\mathfrak{D}^d$ ▷ indicators set produced by Algorithm 1
k_p ▷ packet key
t_p ▷ packet timestamp
Output:
ptr_{k_p} ▷ updated pointer for packet key k_p
A_f ▷ set of associated active flows
Begin
$L_{k_p} \leftarrow \mathcal{H}[k_p]$
$ptr_{k_p}^d \leftarrow \mathfrak{D}^d[k_p]$
while $ptr_{k_p}^d < \left|L_{k_p}^d\right|$ and $L_{k_p}\left[ptr_{k_p}^d\right]$.endtime $< t_p$ **do**
$ptr_{k_p}^d \leftarrow ptr_{k_p}^d + 1$
end while
$A_f \leftarrow \emptyset$
$j \leftarrow ptr_{k_p}^d$
while $j < \left|L_{k_p}^d\right|$ and $L_{k_p}^d[j]$.starttime $\leq t_p$ **do**
if $t_p \leq L_{k_p}^d[j]$.endtime **then**
$A_f \leftarrow A_f \cup \{L_{k_p}[j].flow_{id}\}$
end if
$j \leftarrow j + 1$
end while
$\mathfrak{D}^d[k_p] \leftarrow ptr_{k_p}^d$
End.

Definition 5. Let $L_{k_p}^{d}$=$\left(t_i^{start}, t_i^{end}, i\right)$ denotes the ordered flows list associated with the packet key p_k under d direction, where $d \in \{fwd, bwd\}$. Consider t_p is the packet timestamp and $\mathfrak{D}_{k_p}^{d}$ is the corresponding directional indicator. Accordingly, the indicator increment operation is formulated as:

$\mathfrak{D}_{k_p}^{d} = \min\{j : j < \left| L_{k_p}^{d} \right| \wedge t_j^{end} \geq t_p\}$. That is, the indicator is increased to the first flow whose end time is strictly greater than or equal to the packet timestamp, to effectively ignoring all expired flows.

Definition 6. The candidate active flow set associated with packet p can be defined as follows: $A_p = \{L_{k_p}^{d}[j]: j \geq \mathfrak{D}_{k_p}^{d}, t_j^{start} \leq t_p \leq t_j^{end}\}$. Therefore, A_p comprises non-expired flows whose duration intervals incorporate the timestamp of packet p.

Specifically, given a packet p in Algorithm 2, the associated flow list $L_{k_p}^{d}$ is efficiently retrieved via constant time hash indexing. The indicator $\mathfrak{D}_{k_p}^{d}$ is then advanced to bypass expired flows whose end times exceed the packet timestamp. The remaining candidate active flows are sequentially examined until a temporal ordering condition is not satisfied, i.e., the flow start time precedes the packet timestamp.

Illustrative Example: This example elucidates how the hash structure $\mathcal{H}$ gathers flows sharing the same network key into sorted lists, thereby minimizing the packet-to-flow association problem from an exhaustive search across the entire set of flows to a more focused search within a restricted subset. Assume that three flows have been extracted from PCAP file, as illustrated in Table 1.

Table 1. Sample flows extracted from a packet capture file

Flow ID (F_i)	Source IP Address ($srcIp_i$)	Destination IP Address ($dstIp_i$)	Source Port ($srcPort_i$)	Destination Port ($dstPort_i$)	Protocol ($proto_i$)	Start Time (t_i^{start})	End Time (t_i^{end})
F_1	A	B	2000	60	TCP	1	12
F_2	A	B	2000	60	TCP	6	18
F_3	C	D	1000	500	TCP	3	15

Note: IP = Internet Protocol; TCP = Transmission Control Protocol.

Based upon this table, an associated key will be established for every flow, i.e, $k_{fwd_1} = (A, B, 2000, 60, TCP)$, $k_{bwd_1} = (B, A, 60, 2000, TCP)$ which are relevant to $\mathcal{F}_1$ and $\mathcal{F}_2$, and $k_{fwd_2} = (C, D, 1000, 500, TCP)$ and $k_{bwd_2} = (D, C, 500, 1000, TCP)$ which are associated with $\mathcal{F}_3$. Now, the flows can be organized within the hash structure to be as follows:

$$\mathcal{H}_{fwd}(k_{fwd_1}) = [(1,12,\mathcal{F}_1), (6,18,\mathcal{F}_2)]$$

$$\mathcal{H}_{bwd}(k_{bwd_1}) = [(1,12,\mathcal{F}_1), (6,18,\mathcal{F}_2)]$$

$$\mathcal{H}_{fwd}(k_{fwd_2}) = [(3,15,\mathcal{F}_3)]$$

$$\mathcal{H}_{bwd}(k_{bwd_2}) = [(3,15,\mathcal{F}_3)]$$

The forward and backward hash structures may contain identical flow record entries in order to serve distinct access paths and to enable constant-time lookup for packets regardless of their direction, without requiring key transformation at runtime.

In addition bidirectional lists are formed, $L_{k_1}^{fwd} = [(1,12,\mathcal{F}_1), (6,18,\mathcal{F}_2)]$, $L_{k_1}^{bwd} = [(1,12,\mathcal{F}_1), (6,18,\mathcal{F}_2)]$ and $L_{k_2}^{fwd} = [(3,15,\mathcal{F}_3)], L_{k_2}^{bwd} = [(3,15,\mathcal{F}_3)]$. Both lists contain the same flow intervals but are indexed using different directions.

To apply Algorithm 2, assume that a packet p has a timestamp $t_p = 8$, and key $k_p = (A, B, 2000, 60, TCP)$. . According to this key and after checking the temporal conditions, it can be deduced that this packet matches both $\mathcal{F}_1$ and $\mathcal{F}_2$, since the packet timestamp is within the intervals of both flows. However, when the same packet has a timestamp equal to 14, then the indicator bypasses $\mathcal{F}_1$ as its end time precedes 14, remaining only $\mathcal{F}_2$ as an active flow. This confirms the effectiveness of the indicator mechanism in omitting redundant traversals over inactive flows while preserving sound temporal matching.

Since multiple flows might have the same direction and may be simultaneously ready for the same packet, a deterministic selection algorithm (Algorithm 3) is introduced. Initially, this algorithm discards candidate flows for which the directional packet capacity cap_f^d is depleted.cap_f^d refers to the number of packets that remain assignable to a flow in a specific direction, based on its reference forward or backward packet count. When only one eligible candidate remains, it is chosen directly. In cases where multiple candidates remain, the algorithm prefers an already-initiated flow whose timing falls within the bounds of the reference duration, and selects the most recently active flow. In the absence of such a flow, the candidate whose reference time window is closest to the packet timestamp is chosen instead. When several candidate flows have an equal temporal distance from the packet,dis_f, priority is granted to the flow with the earlier reference time window. If the candidates still have the same referential windows, the flow with the smallest reference index is used as an ultimate deterministic tiebreaker. As a result, each packet maps to at most one flow, guaranteeing duplication-free and deterministic packet-to-flow association.

Algorithm 3: Deterministic selection among overlapping flows candidates

Input:
A_f ▷ set of active candidate flows resulting from Algorithm 2
t_p ▷ timestamp of packet p
cap_f^d ▷ remaining packet capacity of flow f in direction d
$t_{1st}(f)$ ▷ timestamp of the initial packet associated with flow f
$t_{recent}(f)$ ▷ timestamp of the recent packet associated with flow f
Output:
Uf ▷ determinsticlly selected flow
A_f ▷ set of associated active flows
Begin

$Q \leftarrow \{f \in A_f : cap_f^d > 0\}$
if $Q = \emptyset$ **then**
return $\emptyset$
else if $|Q| = 1$ **then**
return only the unique flow in Q
else
$S \leftarrow \{f \in Q: f$ has previously been assigned packets and t_p falls within the temporal bounds identified by the reference flow duration$\}$
if $S \neq \emptyset$ **then**
$Uf \leftarrow \arg\max_{f \in S} t_{recent}(f)$
if multiple flows have the same t_{recent} **then**
$Uf \leftarrow$ the flow with the smallest reference index
end if
end if
for each flow $\in Q$ ***do***
$dis_f \leftarrow |t_p - center(f)|$
end for
$Uf \leftarrow arg\min_{f \in Q} dis_f$
if multiple flows have the same dis_f **then**
$Uf \leftarrow$ **Select** the flow with the earlier reference window
if multiple flows still have equal priority **then**
$Uf \leftarrow$ **Select** the flow with the smallest reference window
end if
end if
end if
End.

3.3 Feature augmentation using payload-based statistics

During this stage, each flow undergoes a feature augmentation process by incorporating of discriminative statistical features extracted from its relevant packet payload contents. In contrast to conventional flow-level features, which mainly depend on packet header information, the proposed framework consolidates statistical measures calculated directly from raw payload data to attain fine-grained traffic behavior. In essence, this stage aims to capture intrinsic patterns within the data distribution, thereby enhancing the distinction between benign traffic and anomalous traffic, particularly in scenarios involving encrypted or obfuscated communications.

Typically, this stage consists of four sequential steps, as described below. In the first step, the payload for every packet belonging to a particular flow is represented as a sequence of bytes. In the second step, three statistical metrics are calculated for this sequence.

•**Shannon Entropy**: In the packet context, entropy denotes the degree of uncertainty in the payload byte distribution. It can be considered an indicator of payload randomness, since encrypted, compressed, or obscured content may exhibit higher entropy values than structured and regular traffic. Given a payload with byte probability distribution $Prob(b)$, the entropy can then be computed as:

$$E(p_j) = -\sum_b Prob(b) \log_2 Prob(b) \quad (1)$$

•**Chi-Square Deviation**: It can be defined in this study as a measurement that determines whether the actual byte-frequency distribution considerably deviates from the estimated equal-probability distribution according to the following formula:

$$CSD(p_j) = \sum_b \frac{(A_b - E_b)^2}{E_b} \quad (2)$$

where, A_b represents the actual distribution of byte value b contained in the payload of packet p_j, and E_b indicates the estimated (predicted) distribution of byte value b under an even distribution. Therefore, the chi-square deviation can quantify the extent of deviation from uniformity, with lower values indicating a more uniform byte distribution and higher values indicating greater deviations from uniformity.

•**Bit Balance**: It is used to describe the distributional characteristics of the binary value distribution within a payload as it is defined as:

$$BB(p_j) = \frac{N_1}{N_{bits}} \quad (3)$$

where, N_1 denotes the total count of occurrences of bits equal to 1, and N_{bits} is the aggregate number of ones and zeros in payload p_j. Payloads with bit balance values approaching 0.5 primarily manifest arbitrary-like behavior, and biased distribution, though, may refer to structured or deterministic content.

In the third step, the calculated payload features are aggregated for each flow to construct a concise representation, due to the multiple packets contained within every single flow. For each feature $\Upsilon \in \{E, CSD, BB\}$, and for each packet $p_j \in F_i$, four statistical descriptors are calculated, including mean μ, standard deviation σ, maximum max, and minimum min values.

In the final step, for each flow F_i, a final feature vector is formed as:

$$\gamma(F_i) = \{\mu_E, \mu_{CSD}, \mu_{BB}, \sigma_E, \sigma_{CSD}, \sigma_{BB}, max_E, max_{CSD}, max_{BB}, min_E, min_{CSD}, min_{BB}\}$$

This feature vector is finally concatenated with the original preexisting flow-based vector to produce the comprehensive input to the classification model.

3.4 Machine learning-based classification stage

After producing the augmented feature space, supervised learning models are employed to assess the effectiveness of the proposed model. In essence, five classification models representing different learning paradigms are chosen: logistic regression [26], extra trees [27], adaptive boosting [28], extreme gradient boosting [29], and a multilayer perceptron [30]. These classifiers are chosen to cover all four types of learning paradigms: linear, ensemble, boosting and neural-network.

3.5 Shapley Additive Explanations-based interpretability stage

In order to improve the interpretability of the selected classifier, SHAP is utilized. In principle, SHAP characterizes the contribution of both traditional flow-level features and the introduced payload-derived features to the final detection decision, and evaluates the effectiveness of these features.

4. EXPERIMENTAL CONFIGURATIONS

The proposed framework was empirically evaluated using two publicly accessible intrusion detection datasets: CICIDS2017 [17, 31] and CICIoMT2024 [32, 33], both of which support the payload-aware feature extraction process. Specifically, one traffic capture from the CICIDS2017 dataset was employed, namely the Tuesday traffic capture. The selected capture compromises both benign and suspicious traffic, covering attacks against Hypertext Transfer Protocol (HTTP) and File Transfer Protocol (FTP) services. This dataset was adopted because it is broadly considered a benchmark in intrusion detection system research and presents de facto enterprise network traffic with standardized flow-related representations. A subset of CICIoMT2024 was also chosen to further evaluate the generalization capability related to the proposed framework, and to assess modern attack behaviors. The selected subset compromises both normal traffic and three representative attack classes: Message Queuing Telemetry Transport (MQTT), reconnaissance, and spoofing attack activities. The experiments were carried out under both binary (benign and malicious traffic) and multi-class (the mentioned multiple attack classes) classification settings. For every dataset, the traditional flow-level features were obtained from the available CSV files, whereas the proposed payload-aware features were extracted from the corresponding packet capture files using the proposed algorithms. Before partitioning the datasets, the distribution of traffic flows among more than one subset must be avoided. As a result, the below subsection illustrates that.

4.1 Data partitioning with session awareness

This subsection is dedicated to mitigate the risk of training-test data dependency arising from the allocation of closely associated traffic flow records belonging to the same communication session across different data partitions. Instead of depending solely on record-level stratified sampling, the suggested algorithm (Algorithm 4) groups flow data into bidirectional sessions prior to partitioning.

Algorithm 4: Traffic grouping for independent data partitioning with bidirectional awareness of session

Input:
Uf ▷ determinsticlly selected flow resulting from Algorithm 3
α ▷ inactivity threshold for temporally separating distinct sessions
Output:
$\mathbb{S}_i$ ▷ session identifier for each flow f_i
Begin
for each flow $f_i \in Uf$ **do**
$e_s \leftarrow (srcIp_i, srcPort_i)$ {constructing source endpoint}
$e_d \leftarrow (dstIp_i, dstPort_i)$ {constructing destination endpoint}
if $e_s \leq e_d$ **then**
$E_{low} \leftarrow e_s$
$E_{high} \leftarrow e_d$
else
$E_{low} \leftarrow e_d$
$E_{high} \leftarrow e_s$
end if
$B_i = (E_{low}, E_{high}, proto_i)$ {defining the bidirectional session base}
end for

Group the flows that have the same base B_i and then sort them within each group by $timestamp_i$ in ascending order, such that $(r_1^{B_i}, r_2^{B_i}, \dots, r_m^{B_i})$ be the time ordered flows belonging to B_i
for each session base B_i **do**
Initialize session counter $\mathbb{S}c \leftarrow 1$
Assign the first flow $r_1^{B_i}$ in B_i to session $\mathbb{S}_1^{B_i} = (B_i, \mathbb{S}c)$
for $j \leftarrow 2$ to m **do**
$\Delta t \leftarrow timestamp\,(r_j^{B_i} - r_{j-1}^{B_i})$
if $\Delta t > \alpha$ **then**
$\mathbb{S}c \leftarrow \mathbb{S}c + 1$
end if
Assign the flow $r_j^{B_i}$ to session $\mathbb{S}_j^{B_i} = (B_i, \mathbb{S}c)$
end for
end for
End.

Typically, Algorithm 4 initially establishes a session base that is not contingent upon direction, derived from the communicating IP-port endpoints and the protocol. Flow records belonging to the same session base are arranged in chronological order, and a new session commences once the inactivity period between two successive records exceeds the threshold α. The produced session identifier is subsequently employed as the categorization variable for dataset partitioning, guaranteeing that all flow records associated with the same specified session are retained within a singular training, validation, or test subset. The overlap of sessions is then examined among the training, validation, and test sets to ensure the independence of partitions. Furthermore, all preprocessing procedures capable of extracting insights from the data, such as feature scaling and Synthetic Minority Over-Sampling Technique (SMOTE)-based oversampling [34], are executed solely on the training subset, thus ensuring that information from the validation or test datasets does not affect model training. Notably, SMOTE is implemented on the training data in order to mitigate class imbalance and minimize potential bias toward the majority class.

5. RESULTS AND DISCUSSION

This section introduces and discusses the experimental results generated from applying the proposed framework. Essentially, the results are discussed from the following three perspectives:

5.1 Packet-to-flow association performance

The packet-to-flow association algorithms are evaluated using the CICIDS2017 dataset, which comprises large-scale and heterogeneous daily traffic captures wherein both packet captures and flow records co-occur within the same network trace. This dataset requires explicit packet-to-flow association processing prior to integrating the payload-oriented features with the flow-level ones. Conversely, the adopted CICIoMT2024 subsets are structured into separate packet capture and CSV files for each attack category, without explicit flow identifiers, enabling payload-extracted features to be combined via protocol-aware aggregation rather than direct packet-to-flow associating.

In order to evaluate the correctness of the packet-to-flow-association algorithm, Table 2 summarizes a set of

complementary validation metrics encompassing both packet and flow levels. For packet-level analysis, the time-eligible packets metric is used to identify which packets have timestamps that lie within the temporal range mentioned by the reference CSV records. Eligible association coverage quantifies the percentage of eligible packets successfully associated with flows and is calculated using the following equation:

$$Eligible\ Assocattion\ Coverage = \frac{N_{assigned}}{N_{eligible}} \quad (4)$$

where, $N_{assigned}$ is the number of assigned eligible packets, and $N_{eligible}$ is the total number of time eligible packets. In addition, flow-level packet-count consistency was assessed by defining the absolute packet-count difference for each flow F_i:

$$D_i = \left|N_{assigned_i} - N_{CSV_i}\right| \quad (5)$$

where, $N_{assigned_i}$ denotes the number of packets assigned by the proposed association algorithm to flow F_i , and N_{CSV_i} represents the original packet count related to flow F_i and reported in the CSV file. Depending on D_i, exact match rates and agreement rates within tolerance margins of 2 and 5 packets, i.e., $\left|N_{assigned} - N_{CSV}\right| \leq 2$, and $\left|N_{assigned} - N_{CSV}\right| \leq 5$, respectively, were then computed, together with the mean and median of D_i. Precise packet-count agreement was defined as the proportion of flows for which the number of packets assigned by the proposed algorithm exactly matched the corresponding packet count recorded in the reference CSV file. Depending on D_i, agreement rates within tolerance margins of 2 and 5 packets, i.e., $\left|N_{assigned} - N_{CSV}\right| \leq 2, and\ \left|N_{assigned} - N_{CSV}\right| \leq 5$, respectively, were then computed, together with the mean and median of D_i.

The temporal analysis results revealed that, out of the 11,497,247 Transmission Control Protocol (TCP)/User Datagram Protocol (UDP) packets inspected, 3,082,997 packets (26.82%) fell outside the temporal range covered by the reference CSV records. As a result, these packets were excluded from the association evaluation instead of being considered as failed associations. The remaining 8,414,250 packets constituted the temporally eligible group, of which 8,370,404 were successfully associated with their respective flows. Consequently, the eligible association coverage attained 99.48%, with only 0.52% of the eligible packets failing to be associated. Precise packet-count agreement was recorded for 50.33% of the evaluated flows, indicating the absence of any difference between the reconstructed and reference packet counts. Considering minor deviations, 68.06% of the flows deviated by at most 2 packets, while 86.40% deviated by at most 5 packets. These tolerance-based metrics permit a graded assessment of discrepancy magnitude, instead of treating all nonzero differences as equivalent. The median packet-count difference or the central value of the observed differences was found to be zero packets. Conversely, the mean absolute packet-count difference amounted to 6.94 packets, implying that a smaller subset of flows with notably larger discrepancies skewed the overall average upward. It is important to note that the 99.48% value represents the extent of association coverage instead of the accuracy of individual packet-level associations; it should not be considered as association correctness. In the same vein, packet-count agreement captures numerical consistency, but does not guarantee, by itself, the correctness of individual packet-to-flow mappings. These additional validation findings support the effectiveness of the deterministic disambiguation selection strategy presented in Algorithm 3. Typically, out of 3,373,803 packets exhibiting multiple initial flow candidates, 1,099,747 were reduced to a single candidate via the directional-capacity constraint, whereas the remaining 2,253,348 ambiguous cases were disambiguated by conducting the deterministic temporal rule. Notably, neither duplicate packet assignments nor directional-capacity overruns were detected, confirming that each packet was associated with at most one flow.

Table 2. Experimental validation of the proposed packet-to-flow association algorithm

Validation Metric	Value	Percentage
Total Transmission Control Protocol (TCP)/User Datagram Protocol (UDP) packets	11,497,247	100%
Packets exceeding the temporal scope of comma-separated values (CSV)	3,082,997	26.82%
Time-eligible packets	8,414,250	73.18%
Eligible packets successfully associated	8,370,404/ 8,414,250	99.48%
Unassociated eligible packets	43,846/ 8,414,250	0.52%
Precise packet-count agreement	224,422 / 445,909	50.33%
Flows with absolute packet-count difference ≤ 2	303,477 / 445,909	68.06%
Flows with absolute packet-count difference ≤ 5	385,246 / 445,909	86.40%
Median absolute packet-count difference	0	-
Mean absolute packet-count difference	6.94	-
Packets initially matched to multiple candidate flows	3,373,803	-
Multiple-candidate cases resolved to one via directional capacity	1,099,747	-
Cases resolved to one by a deterministic temporal rule	2,253,348	-
Unassigned packets due to exhausted candidate capacities	21,620	-
Duplicate packet assignments	0	0%
Maximum flow assignments per packet	1	-
Forward directional packet-capacity overruns	0	0%
Backward directional packet-capacity overruns	0	0%

As a complement to the automated validation, a manual audit was conducted on a sample of 20 packet-to-flow associations [35] randomly selected from the TCP/UDP packets within the Tuesday-WorkingHours packet capture. A stratified sampling strategy was applied to capture both association outcomes relevant to the proposed algorithm, consisting of 10 DIRECT-UNIQUE cases featuring a single eligible flow candidate and 10 DISAMBIGUATED cases

involving multiple overlapping candidates. This strategy ensures that the audit addressed both straightforward associations and the more difficult cases that called for deterministic resolution. Each sampled packet was manually traced back to its original packet capture entry and compared with the corresponding reference-flow record, checking the connection identifiers, direction, and the timing for consistency. An association was treated as correct only when the reference information clearly and uniquely underpinned the selected flow; it was deemed as incorrect if the reference evidence unambiguously indicated a different flow; otherwise, it was labeled as indeterminate to avoid overstating accuracy. The findings were dissimilar between the two audit strata (Table 3). Each of the 10 DIRECT-UNIQUE cases was judged separately as correct, with no cases deemed incorrect or indeterminate. Conversely, all 10 DISAMBIGUATED cases stayed indeterminate due to the inadequacy of the accessible reference information to independently identify a single candidate flow. Therefore, the ambiguity-resolution decisions of Algorithm 3 were not independently verified by the manual audit that was considered a conservative audit.

The outcomes of the manual audit were not the same as the 99.48% association coverage reported earlier. The coverage value represented the percentage of temporally eligible packets that were successfully associated with flows, whereas the manual audit assessed whether the selected associations could be independently confirmed. Thus, the 99.48% coverage must not be regarded as an indication of the association's correctness.

Table 3. Manual audit findings organized by association stratum

Audit Stratum	Correct	Incorrect	Indeterminate
DIRECT-UNIQUE (n=10)	10 (100%)	0 (0%)	0 (0%)
DISAMBIGUATED (n=10)	0 (0%)	0 (0%)	10 (100%)
Overall (n=20)	10 (50%)	0 (0%)	10 (50%)

5.2 Session-based partitioning performance

To test the impact of the inactivity threshold on session formation and data partitioning, four threshold values (1, 5, 10, and 15 minutes) were analyzed. Table 4 presents a summary of the session features and the stability of the partitioning.

Table 4. Sensitivity of data partitioning-based session awareness to the inactivity threshold

Inactivity Threshold (min)	Total Sessions	Mean Flows/Session	Train Ratio	Validation Ratio	Test Ratio	Session Overlap
1	335332	1.329754	60.00103	20.00004	19.99892	0
5	290378	1.535616	59.99924	20.00027	20.00049	0
10	275962	1.615835	60.00148	19.99937	19.99915	0
15	265804	1.677586	60.00327	19.99691	19.99982	0

In general, the sensitivity study indicates that the session-based partitioning exhibited stability across all assessed inactivity threshold values. No overlap in sessions was seen among the training, validation, and test subsets for any evaluated value of α, while the resultant partition sizes closely adhered to the anticipated 60/20/20 distribution. As the threshold value rose, the number of sessions decreased while the average flow count per session grew, indicating the anticipated consolidation of temporally nearby flow records.

Following this sensitivity analysis, a 5-minute inactivity threshold was used for the final evaluation experiments as a moderate parameter setting between the extremes of excessive session fragmentation and the stronger consolidation noted with longer thresholds. It further preserved the desired 60/20/20 partition ratios and was therefore applied for the classification findings reported in Tables 5–10.

An independent record-level audit was conducted to supplement the session-level examination. Prior to the construction of sessions and the partitioning of data, each flow record was allocated a unique original-row identification, which was preserved exclusively for auditing purposes and omitted from the predictive feature set. Subsequent to partitioning, the identifier sets were evaluated in pairs across the training, validation, and test subsets. No overlapping original-record identifiers were detected among training and validation, training and test, or validation and test, as the intersection sizes yielded zero for these three comparisons. This record-level evaluation was documented independently from session overlap, as the absence of session overlap was an anticipated characteristic of group-based partitioning and did not inherently confirm independence at the capture, day, or source level.

5.3 Classification performance

Regarding the CICIDS2017 dataset, the assessed classifiers exhibited commendable detection efficacy in both binary and multi-class scenarios under the session-based partitioning algorithm. Tables 5 and 6, along with Figures 2 and 3, compare the suggested payload-oriented attributes with the conventional flow-level depiction. To address class imbalance, macro-averaged precision, recall, and F1-score were utilized, ensuring each class contributed equally irrespective of its prevalence. In the context of binary classification, extreme gradient boosting attained the superior overall performance, recording an accuracy of 0.99991, macro-precision of 0.99891, macro-recall of 0.99917, macro-F1 score of 0.99904, and area under the receiver operating characteristic curve (ROC-AUC) of 1.00000. Adaptive boosting also yielded commendable outcomes, especially regarding macro-F1 (0.99726) and ROC-AUC (1.00000). In the multi-class classification case, extreme gradient boosting had the most robust performance, attaining an accuracy of 0.99994, macro-precision of 0.99998, macro-F1 score of 0.99869, and ROC-AUC of 1.00000. Adaptive boosting produced results that were nearly equivalent and achieved a marginally superior macro-recall (0.99792) compared to extreme gradient boosting (0.99741). The other classifiers exhibited diminished macro-F1 scores, especially in the multi-class context, even if they often had elevated accuracy and ROC-AUC metrics. In summary, these findings suggest that extreme gradient boosting and adaptive boosting exhibit the best reliable performance across the assessed metrics under the utilized session-based partitioning framework.

Table 5. Classification performance of the proposed payload-oriented feature representation on the CICIDS2017 dataset

Classification Task	Classifier	Accuracy	Precision (Macro)	Recall (Macro)	F1 (Macro)	ROC-AUC (Macro-OvR)
Binary	Extreme Gradient Boosting	**0.99991**	**0.99891**	**0.99917**	**0.99904**	**1.00000**
	Adaptive Boosting	0.99975	0.99570	0.99882	0.99726	**1.00000**
	Multilayer Perceptron	0.99457	0.90753	0.99118	0.94524	0.99958
	Extra Trees	0.99918	0.99426	0.98751	0.99086	0.99940
	Logistic Regression	0.99521	0.92059	0.98548	0.95058	0.99877
Multi-Class	Extreme Gradient Boosting	**0.99994**	**0.99998**	**0.99741**	**0.99869**	**1.00000**
	Adaptive Boosting	0.99993	0.99919	0.99792	0.99855	**1.00000**
	Multilayer Perceptron	0.98963	0.80988	0.99022	0.86502	0.99959
	Extra Trees	0.98915	0.80419	**0.99035**	0.86086	0.99953
	Logistic Regression	0.96027	0.70548	0.98204	0.74658	0.99897

Table 6. Classification performance of the flow-oriented feature representation on the CICIDS2017 dataset

Classification Task	Classifier	Accuracy	Precision (Macro)	Recall (Macro)	F1 (Macro)	ROC-AUC (Macro-OvR)
Binary	Extreme Gradient Boosting	**0.99997**	**0.99972**	**0.99972**	**0.99972**	**1.00000**
	Adaptive Boosting	0.99977	0.99567	0.99935	0.99751	**1.00000**
	Multilayer Perceptron	0.99312	0.89095	0.98773	0.93369	0.99960
	Extra Trees	0.99021	0.85818	0.97882	0.90923	0.99917
	Logistic Regression	0.99278	0.88926	0.98093	0.92997	0.99864
Multi-Class	Extreme Gradient Boosting	**0.99990**	**0.99807**	0.99713	**0.99760**	**1.00000**
	Adaptive Boosting	0.99915	0.97236	**0.99841**	0.98496	0.9999
	Multilayer Perceptron	0.98457	0.78069	0.98548	0.83357	0.99954
	Extra Trees	0.99077	0.87905	0.97123	0.91139	0.99929
	Logistic Regression	0.96155	0.71597	0.98223	0.75502	0.99907

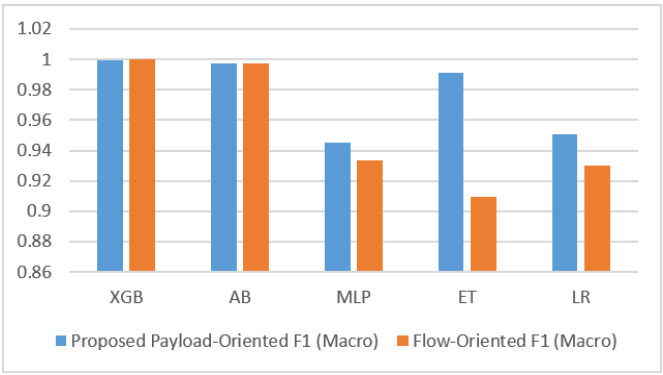


Figure 2. Macro-F1 score comparison on CICIDS2017 based on binary classification

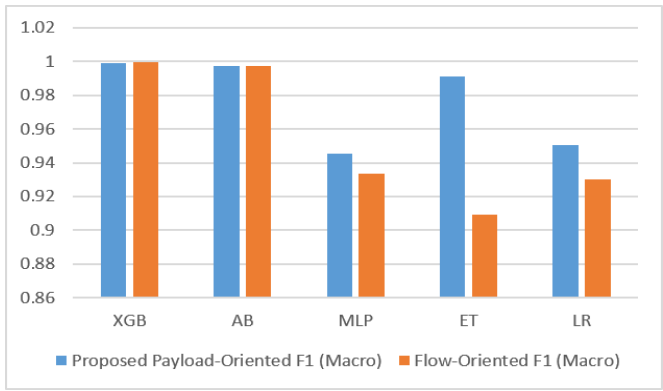


Figure 3. Macro-F1 score comparison on CICIDS2017 based on multi-class classification

The introduced payload-driven feature representation attained impressive results on the CICIoMT2024 dataset in terms of binary and multi-class classification tasks, as presented in Table 7. For binary classification, extreme gradient boosting demonstrated superior performance, with an accuracy of 0.98997 and a macro-F1 of 0.97061, whereas adaptive boosting had a macro-F1 of 0.89549. The remaining classifiers recorded macro-F1 scores of 0.88856, 0.86227, and 0.85364 for the multilayer perceptron, extra trees, and logistic regression, respectively. Within the multi-class case, extreme gradient boosting again attained superior performance, with an accuracy of 0.99036 and a macro-F1 score of 0.93017. Reduced macro-F1 scores were noted for the alternative classifiers, especially extra trees and logistic regression, signifying variances in their ability to differentiate among various traffic categories when equal weight was given to each category. In general, extreme gradient boosting had the most reliable performance in both classification tests.

In contrast, Table 8 displays the findings achieved by the original flow-based feature representation. Extreme gradient boosting recorded macro-F1 scores of 0.95412 and 0.88623 for binary and multi-class classification, respectively. The addition of the recommended payload-focused attributes elevated the macro-F1 score across all five classifiers in both cases. The macro-F1 metric in binary classification improved from 0.95412 to 0.97061 for extreme gradient boosting, from 0.84279 to 0.89549 for adaptive boosting, from 0.80857 to 0.88856 for multilayer perceptron, from 0.85875 to 0.86227 for extra trees, and from 0.77656 to 0.85364 for logistic regression. The enhancements were predominantly more significant in the multi-class task: macro-F1 rose from 0.88623 to 0.93017 for extreme gradient boosting, from 0.64202 to 0.76567 for adaptive boosting, from 0.63048 to 0.70317 for multilayer perceptron, from 0.59455 to 0.62109 for extra trees, and from 0.53299 to 0.60082 for logistic regression. These findings suggest that the payload-centric descriptors offer additional distinguishing information beyond the initial flow characteristics, with the impact of the payload-focused descriptors remaining apparent in the more intricate multi-class classification scenario.

To provide additional insight into the effectiveness of the features extracted in this study, Figures 4 and 5 present a performance comparison of the flow-based and payload-based feature representations under binary and multi-class classification settings, respectively. The comparison is based on the macro-F1 score metric because effective intrusion detection requires high detection capability while minimizing false alarms, which are concurrently captured by this metric via the joint use of precision and recall.

Table 7. Classification performance of the proposed payload-oriented feature representation on the CICIoMT2024 dataset

Classification Task	Classifier	Accuracy	Precision (Macro)	Recall (Macro)	F1 (Macro)	ROC-AUC
Binary	Extreme Gradient Boosting	**0.98997**	**0.97441**	**0.96688**	**0.97061**	**0.99797**
	Adaptive Boosting	0.95917	0.85573	0.95028	0.89549	0.99319
	Multilayer Perceptron	0.95893	0.86511	0.91650	0.88856	0.98042
	Extra Trees	0.95181	0.85637	0.86842	0.86227	0.95405
	Logistic Regression	0.95113	0.86489	0.84325	0.85364	0.94275
Multi-Class	Extreme Gradient Boosting	**0.99036**	**0.92409**	**0.93885**	**0.93017**	**0.99866**
	Adaptive Boosting	0.96481	0.70496	0.90244	0.76567	0.98152
	Multilayer Perceptron	0.94409	0.64305	0.83316	0.70317	0.97784
	Extra Trees	0.93174	0.62251	0.73423	0.62109	0.95572
	Logistic Regression	0.92593	0.56774	0.70675	0.60082	0.94300

Table 8. Classification performance of the flow-oriented feature representation on the CICIoMT2024 dataset

Classification Task	Classifier	Accuracy	Precision (Macro)	Recall (Macro)	F1 (Macro)	ROC-AUC
Binary	Extreme Gradient Boosting	**0.98394**	**0.94647**	**0.96211**	**0.95412**	**0.99717**
	Adaptive Boosting	0.93087	0.78913	0.94359	0.84279	0.99203
	Multilayer Perceptron	0.91222	0.75663	0.91919	0.80857	0.98046
	Extra Trees	0.94846	0.83992	0.88067	0.85875	0.96079
	Logistic Regression	0.90068	0.73299	0.86333	0.77656	0.95402
Multi-Class	Extreme Gradient Boosting	**0.98413**	**0.85671**	**0.92506**	**0.88623**	**0.99775**
	Adaptive Boosting	0.92320	0.59874	0.88002	0.64202	0.97959
	Multilayer Perceptron	0.91142	0.58194	0.86122	0.63048	0.97954
	Extra Trees	0.92468	0.85443	0.71736	0.59455	0.95722
	Logistic Regression	0.84036	0.54019	0.73729	0.53299	0.94227

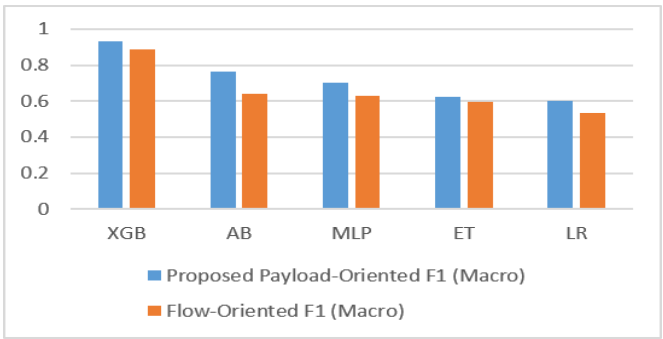


Figure 4. Macro-F1 score comparison on CICIoMT2024 based on binary classification

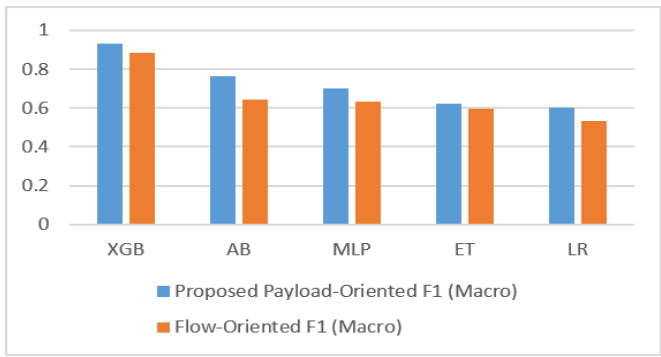


Figure 5. Macro-F1 score comparison on CICIoMT2024 based on multi-class classification

Table 9. Feature-level ablation results for binary classification with test macro-F1 scores (mean ± standard deviation)

Configuration of Features	Extreme Gradient Boosting	Adaptive Boosting	Multilayer Perceptron	Extra Trees	Logistic Regression
Original Flow Features	0.99959 ± 0.00026	0.99956 ± 0.00015	0.85294 ± 0.01725	0.92185 ± 0.00521	0.92529 ± 0.00886
Shannon Entropy + Original	0.99956 ± 0.00040	0.99898 ± 0.00074	0.89364 ± 0.02493	0.94844 ± 0.00577	0.93860 ± 0.01130
Chi-Square Deviation + Original	0.99956 ± 0.00030	0.99959 ± 0.00014	0.88456 ± 0.02488	0.92568 ± 0.00497	0.93409 ± 0.01156
Bit Balance + Original	0.99956 ± 0.00040	0.99879 ± 0.00048	0.88190 ± 0.02896	0.93662 ± 0.00333	0.93880 ± 0.00914
Shannon Entropy +Chi-Square Deviation + Original	0.99953 ± 0.00038	0.99904 ± 0.00047	0.89414 ± 0.02815	0.95791 ± 0.00877	0.94172 ± 0.01001
Shannon Entropy + Bit Balance + Original	0.99959 ± 0.00035	0.99907 ± 0.00052	0.90660 ± 0.01007	0.96585 ± 0.01153	0.93669 ± 0.01004
Chi-Square Deviation + Bit Balance + Original	0.99956 ± 0.00033	0.99879 ± 0.00030	0.89884 ± 0.01769	0.93773 ± 0.00325	0.94057 ± 0.00853
Shannon Entropy + Chi-Square Deviation + Bit Balance + Original	0.99953 ± 0.00015	0.99953 ± 0.00015	0.90318 ± 0.01227	0.98637 ± 0.00679	0.93986 ± 0.00869

Table 10. Feature-level ablation results for multi-class classification with test macro-F1 scores (mean ± standard deviation)

Configuration of Features	Extreme Gradient Boosting	Adaptive Boosting	Multilayer Perceptron	Extra Trees	Logistic Regression
Original Flow Features	0.99931 ± 0.00045	0.97011± 004303	0.80413 ± 0.02411	0.91365 ± 0.00683	0.74855 ± 0.01163
Shannon Entropy + Original	0.99969 ± 0.00019	0.99965 ± 0.00033	0.85317 ± 0.02104	0.89962 ± 0.07740	0.75173 ± 0.01043
Chi-Square Deviation + Original	0.99931 ± 0.00054	0.99915 ± 0.00079	0.85216 ± 0.01162	0.84075 ± 0.06460	0.75249 ± 0.01658
Bit Balance + Original	0.99934 ± 0.00057	0.99926 ± 0.00086	0.85081 ± 0.01892	0.86464 ± 0.06667	0.75246 ± 0.01428
Shannon Entropy +Chi-Square Deviation + Original	0.99931 ± 0.00060	0.99913 ± 0.00080	0.86197 ± 0.00871	0.92486 ± 0.05494	0.76035 ± 0.02890
Shannon Entropy + Bit Balance + Original	0.99926 ± 0.00061	0.99910 ± 0.00083	0.86032 ± 0.01154	0.90992 ± 0.08195	0.75817 ± 0.02788
Chi-Square Deviation + Bit Balance + Original	0.99941 ± 0.00052	0.99916 ± 0.00072	0.85884 ± 0.00590	0.90854 ± 0.06827	0.75696 ± 0.01767
Shannon Entropy + Chi-Square Deviation + Bit Balance + Original	0.99944 ± 0.00052	0.99910 ± 0.00070	0.85314 ± 0.01564	0.89222 ± 0.06780	0.76115 ± 0.02716

The feature-level ablation assessment conducted over five random seeds, including 42, 52, 62, 72, and 82, indicates that the impact of the payload-derived descriptors differs among classifiers and classification tasks (Tables 9 and 10). Essentially, in binary classification, extreme gradient boosting exhibited consistently superior performance across all configurations, achieving macro-F1 values near 0.9995 with minimal fluctuations across seeds. A more substantial effect was recorded for extra trees, as its macro-F1 escalated from 0.92185 ± 0.00521 with the original flow characteristics to 0.98637 ± 0.00679 after integrating all three payload descriptors. Within the multi-class scenario, extreme gradient boosting maintained impressive stability, recording 0.99931 ± 0.00045 with the basic features and 0.99944 ± 0.00052 with the whole feature set. The influence of the payload descriptors was notably obvious for adaptive boosting, where the macro-F1 advanced from 0.97011 ± 0.00403 to 0.99910 ± 0.00070 when all three descriptors were utilized. Alternatively, extra trees showed enhanced variability in relation to feature configurations and random seeds, signifying a stronger sensitivity to feature composition and partitioning of data. The repeated-seed outcomes reveal that Shannon entropy, chi-square deviation, and bit balance deliver supplementary data, whose influence is reliant on the classifier instead of achieving a standardized improvement across all models.

5.4 Shapley Additive Explanations-based explainability analysis

The SHAP analysis provides insights into the contributions of both traditional flow-level features and payload-derived features. Essentially, the SHAP analysis depicted in Figure 6 for CICIDS2017 demonstrates that the extreme gradient boosting model is mostly driven by standard flow attributes, notably Bwd Header Length, Packet Length Mean, and Init_Win_bytes_forward. Of the recommended payload-derived descriptors, EntropyFwd_Count ranks fourth in global feature importance, suggesting that payload entropy contributes additional information alongside the prevailing flow-based attributes. In the context of CICIoMT2024, Figure 7 illustrates that traditional traffic characteristics, specifically IAT, rst_count, and Rate, possess the greatest global SHAP significance. Nevertheless, other suggested payload-derived descriptors are also listed among the highest-ranked attributes, such as EntropyBwd_Count, ChiSquareFwd_Min, EntropyFwd_Count, ChiSquareFwd_Max, and EntropyBwd_Min. This suggests that entropy- and chi-square-derived payload descriptors offer supplementary distinguishing insights compared to traditional traffic characteristics.

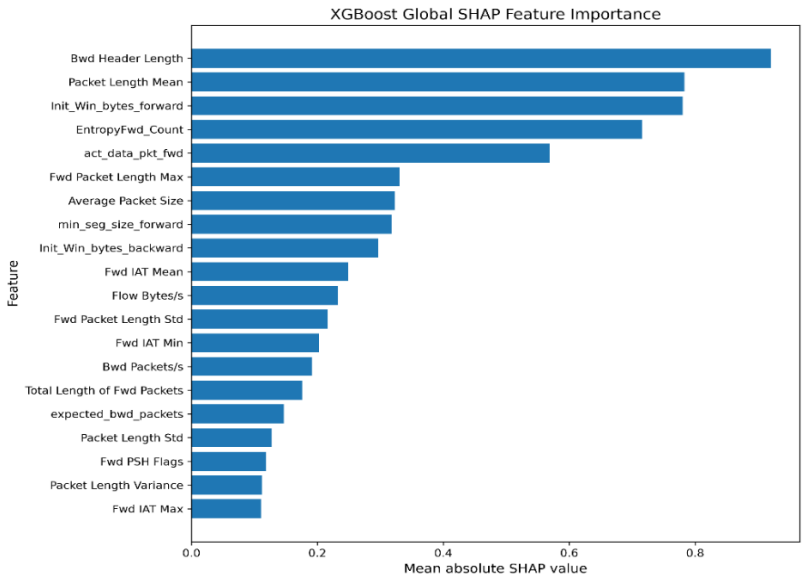


Figure 6. Shapley Additive Explanations (SHAP) analysis of the proposed payload-oriented features on the CICIDS2017 dataset

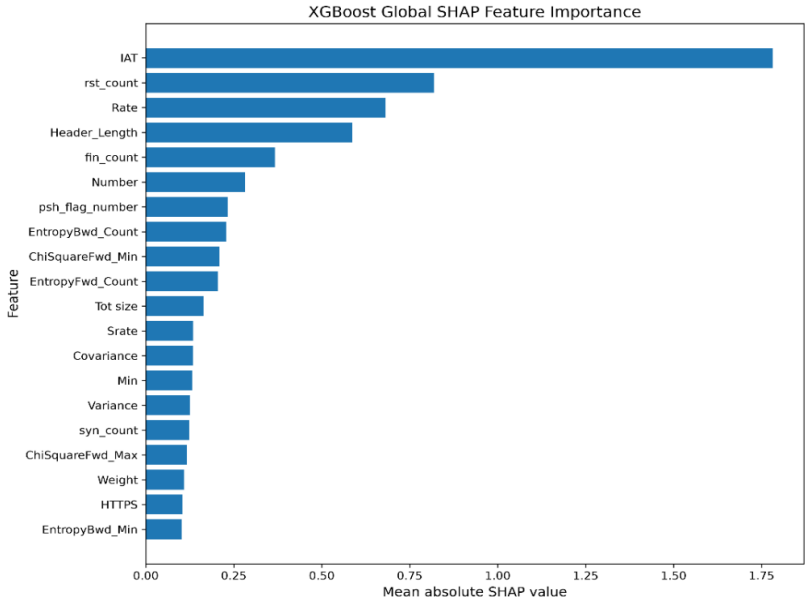


Figure 7. Shapley Additive Explanations (SHAP) analysis of the proposed payload-oriented features on the CICIoMT2024 dataset

6. CONCLUSIONS

This study introduced a payload-aware intrusion detection framework that consolidates packet-based statistical features together with standard flow-based network features. Particularly, the framework incorporates three payload-derived features, including entropy, chi-square, and bit balance, which are fed into machine learning-based intrusion detection classifiers. To enable payload-feature consolidation in large-scale network traffic traces, a tailored packet-to-flow association algorithm based on temporal indexing was designed and validated on the CICIDS2017 dataset. Experimental analysis on CICIDS2017 and CICIoMT2024 also confirmed that the proposed feature representation consistently enhanced or preserved classification performance under several classifiers across both binary and multi-class settings. In addition, the obtained results suggest that the augmented features contribute complementary discriminative information beyond traditional flow-level features, resulting in improved detection capability. Furthermore, SHAP explainability analysis revealed that a number of entropy- and bit balance-based features were identified among the most influential features across both datasets. This implies that the payload-derived features can enrich the flow-level features with more meaningful statistical features, thereby contributing substantially to the decision-making process.

In spite of the definitive assessment of cross-partition duplication and session-level differentiation, future validation through capture-level, day-level, or source-level partitioning could strengthen the evidence of generalizability across independently isolated traffic groups. In addition, the proposed framework can be subsequently developed by exploring additional payload-oriented statistical descriptors and by applying the approach in more heterogeneous traffic environments, incorporating a specific comparison between encrypted and unencrypted traffic to evaluate the contribution of the proposed method under encryption conditions. Further research may also investigate the application of the proposed framework with deep learning architectures.

REFERENCES

[1] Khraisat, A., Gondal, I., Vamplew, P., Kamruzzaman, J. (2019). Survey of intrusion detection systems: Techniques, datasets and challenges. Cybersecurity, 2: 20. https://doi.org/10.1186/s42400-019-0038-7

[2] Kikissagbe, B.R., Adda, M. (2024). Machine-learning-based intrusion detection methods in IoT systems: A comprehensive review. Electronics, 13(18): 3601. https://doi.org/10.3390/electronics13183601

[3] Uddin, M.A., Aryal, S., Bouadjenek, M.R., Al-Hawawreh, M., Talukder, M.A. (2024). usfAD based effective unknown attack detection focused IDS framework. Scientific Reports, 14: 29103. https://doi.org/10.1038/s41598-024-80021-0

[4] Boukela, L., Zhang, G.X., Yacoub, M., Bouzefrane, S. (2021). A near-autonomous and incremental intrusion detection system through active learning of known and unknown attacks. In 2021 International Conference on Security, Pattern Analysis, and Cybernetics（SPAC), Chengdu, China, pp. 374-379. https://doi.org/10.1109/SPAC53836.2021.9539947

[5] Kurniabudi, Stiawan, D., Darmawijoyo, Bin Idris, M.Y., Bamhdi, A.M., Budiarto, R. (2020). CICIDS-2017 dataset feature analysis with information gain for anomaly detection. IEEE Access, 8: 132911-132921. https://doi.org/10.1109/ACCESS.2020.3009843

[6] Villafranca, A., Tasic, I., Cano, M.D. (2026). TRUSTLab dataset: A real-world CICFlowMeter dataset for IoT/edge intrusion detection. Frontiers in Computer Science, 8: 1803271. https://doi.org/10.3389/fcomp.2026.1803271

[7] Kenyon, A., Deka, L., Elizondo, D. (2024). Characterising payload entropy in packet flows—Baseline entropy analysis for network anomaly detection. Future Internet, 16(12): 470. https://doi.org/10.3390/fi16120470

[8] Shannon, C.E. (1948). A mathematical theory of communication. Bell System Technical Journal, 27(3): 379-423. https://doi.org/10.1002/j.1538-

7305.1948.tb01338.x
[9] Pearson, K. (1900). On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 50(302): 157-175. https://doi.org/10.1080/14786440009463897
[10] Rukhin, A., Soto, J., Nechvatal, J., et al. (2010). A statistical test suite for random and pseudorandom number generators for cryptographic applications. NIST Special Publication 800-22. https://doi.org/10.6028/NIST.SP.800-22r1a
[11] Wireshark Foundation. wireshark(1) Manual Page. https://www.wireshark.org/docs/man-pages/wireshark.html.
[12] Wireshark Foundation. tshark(1) Manual Page. https://www.wireshark.org/docs/man-pages/tshark.html.
[13] Lundberg, S.M., Lee, S.I. (2017). A unified approach to interpreting model predictions. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, California, USA, pp. 4768-4777. https://dl.acm.org/doi/10.5555/3295222.3295230.
[14] Akif, M.A., Butun, I., Williams, A., Mahgoub, I. (2025). Hybrid machine learning models for intrusion detection in IoT: Leveraging a real-world IoT dataset. arXiv preprint arXiv:2502.12382. https://doi.org/10.48550/arXiv.2502.12382
[15] Kasongo, S.M. (2023). A deep learning technique for intrusion detection system using a Recurrent Neural Networks based framework. Computer Communications, 199: 113-125. https://doi.org/10.1016/j.comcom.2022.12.010
[16] Farhan, B.I., Jasim, A.D. (2022). Performance analysis of intrusion detection for deep learning model based on CSE-CIC-IDS2018 dataset. Indonesian Journal of Electrical Engineering and Computer Science, 26(2): 1165-1172. https://doi.org/10.11591/ijeecs.v26.i2.pp1165-1172
[17] Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. In Proceedings of the 4th International Conference on Information Systems Security and Privacy - ICISSP, Funchal, Portugal, pp. 108-116. https://doi.org/10.5220/0006639801080116
[18] Lashkari, A.H. (2021). CICFlowMeter. Canadian Institute for Cybersecurity. https://github.com/ahlashkari/CICFlowMeter.
[19] Yu, L., Dong, J.T., Chen, L.H., et al. (2021). PBCNN: Packet bytes-based convolutional neural network for network intrusion detection. Computer Networks, 194: 108117. https://doi.org/10.1016/j.comnet.2021.108117
[20] Kiflay, A., Tsokanos, A., Fazlali, M., Kirner, R. (2024). Network intrusion detection leveraging multimodal features. Array, 22: 100349. https://doi.org/10.1016/j.array.2024.100349
[21] Yu, H.R., Yang, W.C., Cui, B.J., Sui, R.Q., Wu, X.D. (2024). Renyi entropy-driven network traffic anomaly detection with dynamic threshold. Cybersecurity, 7: 64. https://doi.org/10.1186/s42400-024-00249-1
[22] Sattar, S., Khan, S., Khan, M.I., et al. (2025). Anomaly detection in encrypted network traffic using self-supervised learning. Scientific Reports, 15: 26585. https://doi.org/10.1038/s41598-025-08568-0
[23] Zhou, Y.Y., Zhang, W., Feng, W.D. (2025). RD-KNN: Detecting malicious encrypted traffic as processing natural language. In Proceedings of the 9th Asia-Pacific Workshop on Networking, Shanghai, China, pp. 258-260. https://doi.org/10.1145/3735358.3737755
[24] Younisse, R., Ahmad, A., Abu Al-Haija, Q. (2022). Explaining intrusion detection-based convolutional neural networks using Shapley additive explanations (SHAP). Big Data and Cognitive Computing, 6(4): 126. https://doi.org/10.3390/bdcc6040126
[25] Wang, M.N., Zheng, K.F., Yang, Y.Q., Wang, X.J. (2020). An explainable machine-learning framework for intrusion detection systems. IEEE Access, 8: 73127-73141. https://doi.org/10.1109/ACCESS.2020.2988359
[26] Hosmer, D.W., Lemeshow, S., Sturdivant, R.X. (2013). Applied Logistic Regression, 3rd ed. John Wiley & Sons. https://doi.org/10.1002/9781118548387
[27] Geurts, P., Ernst, D., Wehenkel, L. (2006). Extremely randomized trees. Machine Learning, 63: 3-42. https://doi.org/10.1007/s10994-006-6226-1
[28] Freund, Y., Schapire, R.E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences, 55: 119-139. https://doi.org/10.1006/jcss.1997.1504
[29] Chen, T.Q., Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, California, USA, pp. 785-794. https://doi.org/10.1145/2939672.2939785
[30] Rumelhart, D.E., Hinton, G.E., Williams, R.J. (1986). Learning representations by back-propagating errors. Nature, 323: 533-536. https://doi.org/10.1038/323533a0
[31] Canadian Institute for Cybersecurity (CIC). (2017). Intrusion detection evaluation dataset (CIC-IDS2017). https://www.unb.ca/cic/datasets/ids-2017.html.
[32] Dadkhah, S., Neto, E.C.P., Ferreira, R., Molokwu, R.C., Sadeghi, S., Ghorbani, A.A. (2024). CICIoMT2024: A benchmark dataset for multi-protocol security assessment in IoMT. Internet of Things, 28: 101351. https://doi.org/10.1016/j.iot.2024.101351
[33] Canadian Institute for Cybersecurity (CIC). (2024). CIC IoMT dataset 2024: Attack vectors in healthcare devices—A multi-protocol dataset for assessing IoMT device security. https://www.unb.ca/cic/datasets/iomt-dataset-2024.html.
[34] Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P. (2002). SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, 16: 321-357. https://doi.org/10.1613/jair.953
[35] Farahmandi, F. (2026). Manual validation [Dataset]. Zenodo. https://doi.org/10.5281/zenodo.22007715