Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 12, No. 02, June 2026.

ANALYSIS OF PCAP-DERIVED FLOW-BASED TRAFFIC REPRESENTATION FOR LIGHTWEIGHT INTRUSION DETECTION

Andrés Eduardo Villamarín Olmos and Edward Paul Guillen Pinto

(Received: 13-Mar.-2026, Revised: 30-May-2026, Accepted: 6-Jun.-2026)

ABSTRACT

The proliferation of interconnected network infrastructures and IoT devices has significantly expanded the cyber-attack surface, requiring efficient Machine Learning-based Intrusion Detection Systems (IDSs). Although reference datasets like UNSW-NB15 exist, their official features impose limitations regarding flexibility and class imbalance. This study evaluates the impact of a custom data representation by constructing a new dataset from the original UNSW-NB15 PCAP files. We implemented a workflow to label packets, group unidirectional flows and extract a reduced set of 21 features, comparing this representation with the official 49-feature UNSW-NB15 set using different ML architectures in binary and multi-class classification tasks. Results indicate that the custom dataset achieves competitive performance despite a significant reduction in file size and the number of features. Notably, the custom representation effectively balances detection accuracy with computational efficiency, offering a viable strategy for environments with strict operational constraints, such as edge nodes or IoT gateways.

KEYWORDS

Intrusion detection systems (IDSs), Network traffic classification, UNSW-NB15, Machine learning, Network security.

1. INTRODUCTION

In the current digital era, the proliferation of connected devices has reached unprecedented levels. Industry reports project that the number of Internet of Things (IoT) devices will exceed 30 billion by 2030, generating a massive volume of data at the network edge [1]. This hyper-connectivity, however, has significantly expanded the attack surface, with global cybercrime costs estimated to reach $10.5 trillion annually by 2025 [2]. In this scenario, traditional signature-based security mechanisms are insufficient to handle the volume and sophistication of modern threats, making Intrusion Detection Systems (IDSs) based on Machine Learning (ML) indispensable tools for automating the identification of anomalous traffic.

However, the effectiveness of these ML-based systems largely depends not only on the algorithms used, but fundamentally on the quality of the data and the design of the features representing network traffic. While deep-learning models have shown high detection rates, their deployment in limited-resource environments, such as IoT gateways, is often hindered by the high computational cost associated with processing high-dimensional data. Consequently, optimizing the data representation stage is critical to achieving a balance between detection accuracy and operational efficiency.

Among contemporary datasets, UNSW-NB15 for enterprise network traffic has become a reference for the evaluation of detection methods, since it combines real traffic with multiple families of up-to-date attacks and provides CSV files with 49 attributes derived from PCAP captures. Nevertheless, the direct use of these official CSV files imposes certain constraints: on the one hand, it restricts the flexibility to implement custom packet grouping strategies; on the other, it inherits class imbalance and pre-processing decisions that shape the behavior of the models. Consequently, it remains an open question to what degree it is possible to redesign the representation of traffic without sacrificing performance, while at the same time reducing data size and the computational cost of training.

In this context, this study investigates how different choices in the representation and structuring of traffic data affect IDS performance. Using UNSW-NB15 as a case study, we construct an alternative packet-grouping strategy, reduce feature sub-sets and compare them against the original configuration in order to characterize their impact on detection performance and their suitability under environments

A. E. V. Olmos (Corresponding Author) and E. P. G. Pinto are with Universidad Militar Nueva Granada, Bogotá, Colombia. Emails: edward.guillen@unimilitar.edu.co and est.andres.villama1@unimilitar.edu.co

with strict operational constraints, such as edge nodes or IoT gateways.

This paper is organized as follows: Section 2 reviews the state of the art regarding IDSs and feature-engineering techniques. Section 3 describes the UNSW-NB15 dataset and its statistical distribution. Section 4 details the methodology, including the pipeline for packet labeling, traffic grouping, feature extraction and the experimental evaluation framework. Section 5 discusses the results obtained from the comparative analysis between the official and the reconstructed dataset. Finally, Section 6 summarizes the main conclusions and outlines directions for future research.

2. RELATED WORK

Network exposure to attack continues to increase and evolve over time, resulting in the constant emergence of previously unseen variants and attack methods. Consequently, a significant proportion of the current literature evaluates its methodologies using the UNSW-NB15 dataset, which integrates real traffic with updated malicious traffic, provides 49 attributes and covers nine contemporary families of attacks, making it a more representative tool compared to historical datasets, such as KDD-99 [3]. On this basis, the works are grouped into three lines: (i) classical learning with feature selection, (ii) deep learning, often preceded by dimension reduction and (iii) deployment-oriented ensembles/hybrids. Table 1 summarizes the key methodologies and reported performance metrics across these research lines.

Classical learning with feature selection. First, across the following studies, researchers combine supervised algorithms with feature selection to reduce dimensionality, mitigate overfitting and decrease computational cost. Methodologically, these studies typically begin with exploratory and correlation analyses that assess attribute relevance and eliminate redundancies before modeling. More et al. [4] strengthen classical models on UNSW-NB15 by cleaning the dataset, running exploratory and correlation analyses, estimating attribute relevance (e.g., with XGBoost) and removing redundant variables. The authors then compare Logistic Regression, SVM, Decision Tree and Random Forest to select the best detector. The study finds that, after feature selection, Random Forest consistently outperforms the alternatives, underscoring the impact of variable filtering on performance.

Ahmad et al. [5] extend the attribute-prioritization approach to an IoT setting aligned with protocol specifics. The authors group features by domain (Flow/MQTT and TCP) and exclude features that induce overfitting, then evaluate RF, SVM and Neural Networks per cluster and in combination. The authors report that the cluster strategy maintains high effectiveness with Random Forest while reducing training time relative to general supervised pipelines, demonstrating the efficacy of protocol-sensitive filtering. Hussain et al. [6] analyze detection under a Zero Trust framework with continuous network operation. The authors prepare the data *via* imputation, encoding and normalization, apply Recursive Feature Elimination (RFE) to identify predictive variables and compare Logistic Regression, Random Forest and XGBoost. XGBoost achieves an AUC of 1.00, indicating strong capacity to capture non-linear relationships in network traffic.

Deep learning, with dimension reduction. Another significant line of research employs deep networks together with dimension reduction to accelerate inference and improve generalization, particularly in resource-constrained scenarios (e.g., IoT) where classical models face limitations in complex, high-volume settings. Jouhari et al. [7] design an efficient IoT IDS using a lightweight CNN-BiLSTM model. A Chi-square selection process reduces the input to the 20 most relevant features before training on UNSW-NB15. The design aims for high accuracy with low complexity; the reported performance is 97.90% (binary) and 97.09% (multi-class), with lower prediction latency attributable to feature reduction.

Sharma and Kumar [8] propose a CapsNet+BiLSTM hybrid that exploits spatial hierarchies and temporal dependencies. Capsules with dynamic routing replace classical convolutions to extract local features and preserve hierarchical information, thereby reducing dimensionality and enhancing feature representation. The architecture then adds BiLSTM for bidirectional sequence modeling to capture temporal dependencies. The authors evaluate the model on CIC-IDS2017, KDD CUP 99 and UNSW-NB15, reporting improvements over individual architectures and 97% accuracy on UNSW-NB15. Vibhute et al. [9] combine feature selection with deep learning by selecting 15 of the 49 UNSW-NB15 attributes *via* Random Forest and training a CNN on the reduced sub-set. The pipeline aims to simplify

the input space and improve generalization, achieving 99.00% test accuracy, 98.86% recall and 99.00% F1. By contrast, Farhan et al. [10] build a sequential DNN preceded by Extra Trees feature selection, reducing 43 features to 8. The reduction improves computational efficiency and inference speed while maintaining effective attack discrimination; on UNSW-NB15 (binary), the authors report 97.93% accuracy and 97% recall/F1.

Deployment-oriented ensembles/hybrids. Finally, transitioning towards production environments, several studies emphasize ensemble and hybrid systems that balance accuracy, latency and robustness, often with feature selection and distributed execution. Chkirbene et al. [11] exemplify this direction with a hybrid that selects significant features using Random Forest and classifies attacks with CART. CART scales well to large datasets and adapts tree structure to input variables. Compared with alternative trees and baselines, the hybrid improves accuracy while reducing complexity and training/prediction time. Kabir et al. [12] study stacking for better generalization using two configurations: (i) XGBoost and KNN as base models with Random Forest as meta-classifier and (ii) XGBoost, Neural Networks and KNN with the same meta-classifier. The pre-selection phase uses Extra Trees and Mutual Information Gain. On UNSW-NB15, the combination of Mutual Information Gain with the first stacking configuration (XGBoost+KNN) reaches 96.24% accuracy, surpassing individual models and highlighting the value of heterogeneous ensembles for complex attack patterns. With a deployment and scalability focus, Belouch et al. [13] implement an Apache Spark pipeline and evaluate SVM, Naïve Bayes, Decision Tree and Random Forest on UNSW-NB15. Random Forest outperforms the other classifiers, achieving 97.49% accuracy with 0.08 s inference time.

Furthermore, Mutambik [14] proposed IoT-FIDS (Flow-based Intrusion Detection System for IoT), with a focus on reducing computational complexity in resource-constrained networks. Unlike methods based on computationally expensive traditional ML algorithms, IoT-FIDS identifies anomalous behaviors by analyzing flow-based representations that capture communication patterns. This use of flow-level features is key to reducing dimensionality and data volume, thereby justifying the adoption of flow-based representations to achieve a balance between efficiency and accuracy in IoT gateways, as proposed in our study.

Table 1. Metrics reported in Related Work on UNSW-NB15.

Reference	Model / Approach	Reported Metrics
[7]	CNN-BiLSTM (Chi-square: 20 features)	ACC = 97.90%(binary); ACC = 97.09%(multi)
[8]	CapsNet + BiLSTM	ACC = 97.00% (UNSW-NB15)
[9]	Random Forest feature selection (15/49) + CNN	ACC = 99.00%; Recall = 98.86%; F1-score = 99.00%
[10]	Extra Trees (8 feats) + DNN	ACC = 97.93%(binary); Recall = 97.00%; F1-score = 97.00%
[12]	Stacking: XGBoost + KNN → Random Forest (with MI gain)	ACC = 96.24%
[13]	Random Forest (Apache Spark)	ACC = 97.49%; Latency = 0.08 s

Note: The work by Mutambik [14] is excluded from this comparison, since its performance metrics were evaluated on the BoT-IoT dataset, utilizing UNSW-NB15 exclusively for training.

3. DATASET UNSW-NB15

UNSW-NB15 is a widely recognized benchmark representing generic enterprise network traffic. It was generated at the UNSW Canberra laboratory with normal and malicious simulated traffic from contemporary attacks. The captures were carried out over two sessions: on 22-01-2015 with a duration of 16 hours and on 17-02-2015 with a duration of 15 hours, totaling 99.1 GB of PCAP files. From these PCAPs, the authors extracted features using Argus and Bro-IDS to produce the official CSVs with 49 features and 9 attack categories, the breakdown of which by class is presented in Table 2.

In this study, we process all 93,715,272 raw packets from the 15 -hour capture PCAP files to build our custom CSV. This choice is based on the better balance between the normal and attack classes compared to the 16 -hour capture session. According to the figures in Table 3, in the 16 -hour capture, the attacks account for only 2.04% (22,215 attacks out of 1,087,202 total records), whereas the 15-hour capture reaches 20.59% (299,068 attacks out of 1,452,842 total records). This indicates that, although using the

15 -hour capture significantly increases the representation of attack categories, a serious class imbalance persists throughout the dataset. For example, in Table 2, while the Normal category comprises 2,218,761 records (87.35%), minority classes such as Worms represent only 174 records (0.007%), posing a significant challenge for detection stability and model training.

Table 2. Distribution of the UNSW-NB15 dataset.

Category	**No. Records**
Normal	2,218,761
Fuzzers	24,246
Analysis	2,677
Backdoors	2,329
DoS	16,353
Exploits	44,525
Generic	215,481
Reconnaissance	13,987
Shellcode	1,511
Worms	174

Table 3. Statistics per day of capture at UNSW-NB15.

Feature	22-01-15 (16h)	17-02-15 (15h)
Src_bytes	4,860,168,866	5,940,523,728
Dst_bytes	44,743,560,943	44,303,195,509
Src_Pkts	41,168,425	41,129,810
Dst_pkts	53,402,915	52,585,462
Normal records	1,064,987	1,153,774
Attack records	22,215	299,068

The information presented in Tables 2 and 3 was obtained directly from the work [3] by Moustafa et al.

4. METHODOLOGY

We followed a four-stage pipeline: (i) label each packet with its category; (ii) group the packets into unidirectional flows; (iii) compute features per flow and consolidate a single CSV; and (iv) apply pre-processing, training and evaluation of machine-learning models, comparing the own CSV (from PCAP) against the official CSVs, as illustrated in Figure 1.

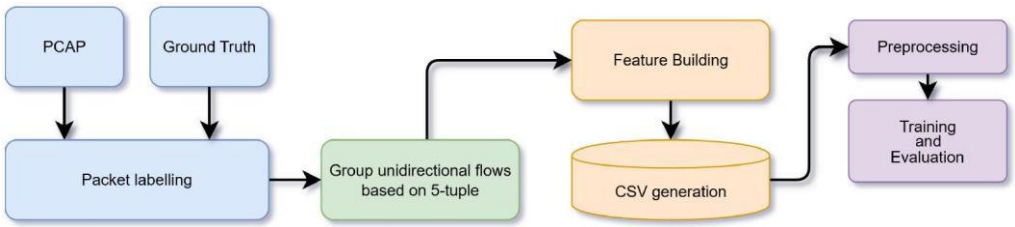


Figure 1. Methodology pipeline.

4.1 Packet Labeling

In addition to the raw PCAP files available in the UNSW-NB15 repository, the set includes a ground truth file that contains key information: the start and end time of each attack in Unix time, its category, the protocol and the source and destination IP addresses and ports.

Based on this file, each packet in the PCAPs was labeled as normal or as one of the nine attack families. To do this, from each packet its timestamp (Unix time) and IP addresses were extracted and then contrasted with the ⟨Source IP, Destination IP, Start time, Last time⟩ records of the ground truth. If

the packet's timestamp fell within the interval [Start time, Last time] associated with that pair of IPs, the corresponding Attack category from the ground truth was assigned; otherwise, it was labeled as normal, since the ground truth only records malicious traffic events. The procedure is summarized in Algorithm 1.

Algorithm 1: Packet Labeling using Ground Truth

```
Data: PCAP packets; GroundTruth with (SrcIP, DstIP, StartTime, LastTime,
      AttackCategory)
Result: Label per packet (normal or attack category)
foreach packet p in PCAP do
    PacketTimestamp ← timestamp(p);
    (PacketSrcIP, PacketDstIP) ← (IP.src(p), IP.dst(p));
    Label ← normal;
    foreach record r in GroundTruth with r.SrcIP = PacketSrcIP and r.DstIP =
     PacketDstIP do
        if r.StartTime ≤ PacketTimestamp ≤ r.LastTime then
            Label ← r.AttackCategory; break;
    save Label for p;
```

4.2 Flow Grouping

With the packets labeled, we proceed to the creation of unidirectional flows. To this end, a grouping strategy based on the 5-tuple method is proposed, together with three flow-closing strategies: category change, a time threshold without meeting the condition and, finally, exceeding a maximum lifetime for an active flow.

To identify the packets belonging to the same unidirectional flow, the 5 -tuple method is used, defined as [source IP, destination IP, source port, destination port, protocol]. The process consists of iterating through the packets one by one in temporal order and grouping those that maintain the same 5 -tuple values, which indicates that they belong to the same unidirectional flow.

The flow must be closed when any of the following three conditions is met: (i) a new packet does not belong to the same category as the other packets in the flow, even if it matches the 5 -tuple; (ii) the flow does not receive new packets that satisfy the 5 -tuple for 15 seconds; or (iii) the flow reaches a maximum active lifetime of 60 seconds from its first packet (even if it remains active by adding more packets).

The selection of these timeout parameters is aligned with established network industry standards and operational monitoring guidelines. The 15 -second idle timeout is a widely adopted configuration in Cisco NetFlow environments and conforms to the IPFIX protocol specifications defined in RFC 5101. Regarding the active timeout, while standard enterprise implementations may employ longer intervals of 30 minutes to minimize processing overhead, RFC 5101 specifies that long-lived flows should be periodically exported to ensure continuous visibility. In the context of intrusion detection, reducing this threshold to 60 seconds is essential for enabling segmented traffic analysis and facilitating near real-time threat response. This design choice is consistent with prominent flow-based extraction frameworks in the literature, such as CICFlowMeter, which utilizes a 60 -second limit to prevent delayed detection of persistent anomalous behaviors. The procedure for creating and closing flows is summarized in Algorithm 2.

It is important to highlight that both the packet-labeling mechanism (Algorithm 1) and the subsequent flow-grouping stage (Algorithm 2) operate under the assumption of static IP allocations and the absence of overlapping, NATed (Network Address Translation) flows. In real-world environments where multiple internal devices share a single public IP address, concurrent sessions may overlap temporally or collide within the same 5-tuple structure, representing an inherent limitation of this dataset construction framework.

4.3 Feature Calculation

To justify the feature design selection, a two-stage workflow was implemented, combining statistical

filtering and explaining ability validation. Initially, a broad set of candidate features was extracted from the raw unidirectional flows. To reduce dimensionality and eliminate uninformative variables, a selection phase based on Mutual Information (MI) was performed. As shown in Figure 2a, features with an importance score above 1% (MI>0.01) were retained, discarding attributes with low predictive power.

Algorithm 2: Unidirectional Flow Construction (5-tuple and closure rules)

```
Data: LabeledPackets (time-ordered), each with {Timestamp, SrcIP, DstIP, SrcPort,
      DstPort, Protocol, Category}
Result: Set of unidirectional flows grouped by 5-tuple and Category
IDLE_TIMEOUT ← 15 s; MAX_LIFETIME ← 60 s;
ActiveFlows ← empty map from 5-tuple to flow state;
foreach packet p in LabeledPackets do
    key ← (p.SrcIP, p.DstIP, p.SrcPort, p.DstPort, p.Protocol);
    if key ∉ ActiveFlows then
        open new flow with Category ← p.Category, StartTime = LastTime =
          p.Timestamp;
        ActiveFlows[key] ← flow;
        continue;
    flow ← ActiveFlows[key];
    if p.Category ≠ flow.Category or (p.Timestamp − flow.LastTime) ≥
      IDLE_TIMEOUT or (p.Timestamp − flow.StartTime) ≥ MAX_LIFETIME
      then
        emit flow;;
        remove ActiveFlows[key];
        open new flow with Category ← p.Category, StartTime = LastTime =
          p.Timestamp;
        ActiveFlows[key] ← flow;
    else
        flow.LastTime ← p.Timestamp;
foreach remaining flow in ActiveFlows do
    emit flow
```

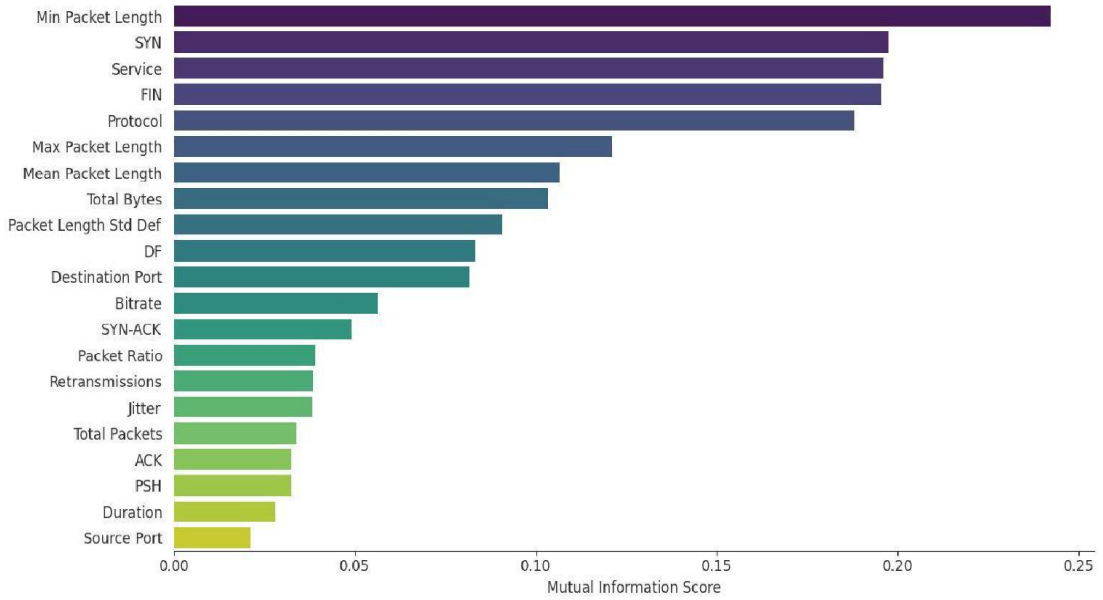
Subsequently, to validate this selection from the model's perspective, a SHAP (SHapley Additive exPlanations) analysis was conducted using a XGBoost model for both binary and multi-class classification tasks (Figure 2b and Figure 2c). This analysis confirmed the individual contribution of each variable. For instance, it was found that attributes statistically significant in Mutual Information, such as the SYN flag, were also important for the binary XGBoost model, but showed less relevance in multi-class classification. This demonstrates how attribute importance varies depending on the predictive goal: while some features serve as general anomaly indicators (attack *vs.* normal traffic), others are required to distinguish between specific attack signatures. The result is a refined set of 21 features (detailed in Table 4) that reduce training complexity and dataset size without compromising detection accuracy.

The 21 features were computed over the packets of each flow and stored in a single CSV file. The generated CSV file contains a total of 2,064,494 records corresponding to the features of each flow, resulting in a final size of 220 MB, which is 2.5 times smaller compared to the total of 560 MB of the official CSV files.

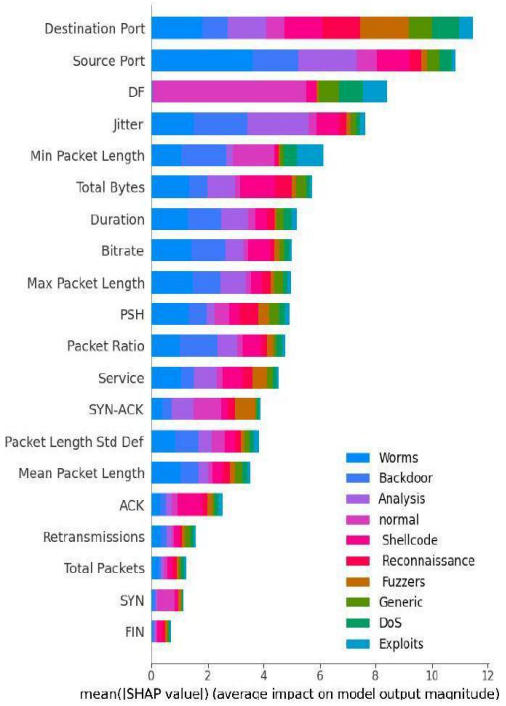
4.4 Pre-processing and Evaluation

To ensure a fair comparison, the full CSV version of the UNSW-NB15 dataset was employed. However, a feature filtering process was implemented to align this set with the attributes present in the official train/test partitions. This decision was made to maintain methodological consistency and, crucially, to

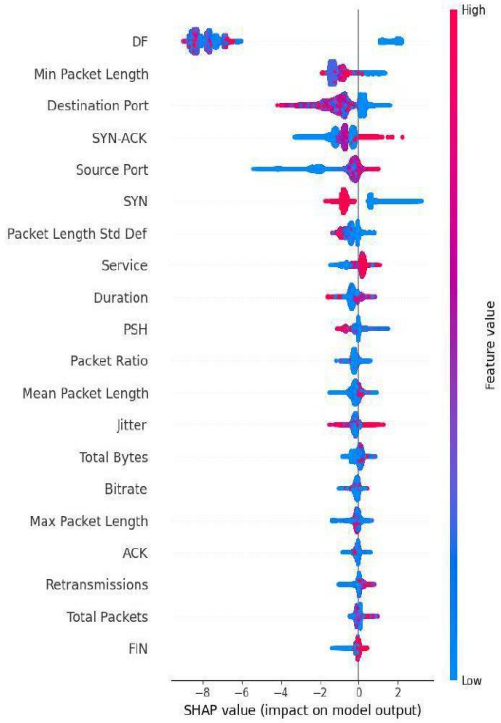
prevent data leakage. Beyond this feature alignment, an identical pre-processing pipeline was applied to both the official UNSW-NB15 CSV and our custom representation derived from the raw PCAP files.



(a) Mutual Information scores.



(b) SHAP importance (Multi-class).



(c) SHAP importance (Binary).

Figure 2. Feature analysis: (a) Mutual Information scores for the selected features set; (b) and (c) SHAP global importance for model explainability.

Pre-processing consists of the following:

- Removal of duplicate records and/or records with missing values (NaN).
- Numerical coding of labels.
- Random mix of records.
- Stratified 5-fold Cross-validation protocol.

Table 4. Unidirectional 21 flow features: definition and unit.

Feature	Definition	Unit
Total Packets	Total number of packets in the flow	packets
Source Port	Source transport-layer port number	integer
Destination Port	Destination transport-layer port number	integer
Protocol	IP protocol identifier	integer
Duration	Elapsed time between first and last packet	s
Min. Packet Length	Smallest packet size observed in the flow	bytes
Max. Packet Length	Largest packet size observed in the flow	bytes
Mean Packet Length	Average packet size in the flow	bytes
Packet Length Std. Dev.	Standard deviation of packet sizes	bytes
Total Bytes	Sum of all packet lengths in the flow	Bytes
DF	'Don't Fragment' bit in the IP header	binary
FIN	TCP 'Finish' flag	binary
SYN	TCP 'Synchronize' flag	binary
PSH	TCP 'Push' flag	binary
ACK	TCP 'Acknowledgment' flag	binary
Retransmissions	Count of retransmitted segments within the flow	integer
Service	Service or application name associated with the observed port	string
Bitrate	Average number of bits transmitted per second over the flow	bps
Jitter	Average variability of inter-packet arrival times	s
SYN-ACK	Delay from the initial SYN to the corresponding SYN-ACK	s
Packet Ratio	Number of packets transmitted in a second	integer

Since our methodology calculates feature flows derived from the raw packets using grouping, calculation and structural rules that differ from the official dataset, it is impossible to directly employ the official test split for a fair evaluation. By using stratified 5-fold cross-validation, we ensure an identical and unbiased evaluation environment for both datasets.

The label was handled under two tasks: binary (normal *vs.* attack) and multi-class (normal +9 attack families) across both data sources. We selected four models per task (Decision Tree, Random Forest, MLP and XGBoost), for a total of 16 training experiments (8 per CSV). Each model and its configuration are summarized in Table 5. For the MLP, only the output layer is adjusted according to the task (sigmoid for binary; softmax for multi-class).

Table 5. Training configuration for each model.

Model	Configuration for binary and multi-class
Decision Tree	Minimum samples split: 2; minimum samples per leaf: 1; criterion: Gini; class weight: balanced.
Random Forest	Trees: 100; minimum samples split: 2; minimum samples per leaf: 1; criterion: Gini; class weight: balanced.
MLP	Layers: Dense(512, ReLU) → Dropout(0.3) → Dense(256, ReLU) → Dense(n classes, Sigmoid/ Softmax). Optimizer: Adam (Initial LR = 0.001, exponential decay to 0.0001 from epoch 50); epochs: 150; batch size: 8192 ; class weight: balanced.
XGBoost	Trees: 1000; max depth: 8; LR: 0.1 (exponential decay to 0.01 from round 100); subsample/ colsample: 0.6; regularization: gamma = 0.1, alpha = 0.5, lambda = 1.5; sample weight: balanced.

The models were evaluated using a Stratified 5-fold Cross-validation scheme to ensure robustness and

account for class imbalance. For each fold, a comprehensive set of performance and computational efficiency metrics was recorded; the results across all folds are presented and discussed in Section 5.

4.5 Statistical Significance

To validate the performance evaluations and determine whether the observed differences between models and datasets are statistically significant, two non-parametric tests were employed. This statistical validation is essential in network-intrusion detection to ensure that performance claims are robust, especially when dealing with the inherent class imbalance of traffic datasets.

4.5.1 McNemar's Test

McNemar's test was employed to perform pairwise comparisons between the classification models (Decision Tree, Random Forest, MLP and XGBoost) on the PCAP-derived dataset. This test is particularly suited for IDS evaluation as it focuses on discordant prediction cases where one model classifies an attack correctly while the other fails. By analyzing these prediction shifts, the test evaluates the null hypothesis that both classifiers have an equal proportion of errors, providing a more rigorous basis for asserting model superiority in imbalanced scenarios.

4.5.2 Wilcoxon Signed-rank Test

While McNemar's test evaluates model performance on a single dataset, the Wilcoxon Signed-rank test was implemented to compare the proposed PCAP-derived representation against the original UNSW-NB15 dataset. Using XGBoost as the benchmark architecture, this test evaluates whether the performance shift between datasets is consistent across 10 folds of Stratified K-fold cross-validation. This 10 -fold configuration was specifically chosen for the statistical comparison to provide a more robust sample size, distinct from the 5 -fold setup used during the initial training phase. Unlike parametric tests, Wilcoxon does not assume a normal distribution of the results, making it ideal for comparing metrics, such as latency, training time and F1-score across different data representations.

5. RESULTS AND DISCUSSION

Table 6 summarizes the performance of each model using a Stratified 5-fold Cross-validation evaluation scheme. The mean values for each metric are presented alongside their respective standard deviations for both datasets in binary and multi-class classification tasks. As expected, the task of discriminating between normal traffic and attacks presents an intrinsically lower complexity than identifying specific attack categories; consequently, performance metrics in binary classification are significantly higher across all evaluated models. It is evident that the proposed PCAP-derived approach, by reducing the dimensionality of the input data, optimizes inference time and enables the processing of a higher volume of samples per second-as measured on an NVIDIA A100-SXM4-40GB GPU-in exchange for a minor reduction in detection capability. For instance, the XGBoost model, which demonstrated statistical superiority over the other models, experienced a 6% decrease in its F1-score for multi-class classification, offset by a 48% increase in samples per second. In the binary scenario, a 6% reduction in the F1-score was also observed, against a substantial 68.8% increase in processing performance.

Notably, tree-based models, such as Decision Tree, Random Forest and XGBoost, exhibit an increase in the number of parameters despite being trained on a reduced feature set. This phenomenon suggests that these models require greater depth and node density to compensate for the reduced dimensional information and maintain classification accuracy. However, this structural complexity does not correlate directly with inference speed; it is observed that models with a higher parameter count can achieve superior processing speeds. In contrast to the MLP model, the results confirm that the relationship between parameter load and computational efficiency is not universal, but depends strictly on the nature of the algorithm.

The results of the McNemar test for all pairwise model comparisons are presented in Table 7. Across both binary and multi-class tasks, all comparisons yielded p -values under 0.001, confirming that the performance differences between the models are statistically significant.

The highest test statistics are consistently associated with the MLP model, particularly when compared against Random Forest and XGBoost. This high divergence indicates that the MLP produces a

substantially different error distribution, likely due to its neural network architecture failing on different types of traffic compared to the tree-based ensembles. In contrast, the comparison between Decision Tree and XGBoost shows the lowest statistical divergence. This suggests a closer logical alignment between these two models; however, the test still confirms that the two classifiers produce distinct results, validating that the performance gap observed between the baseline Decision Tree and the optimized XGBoost is statistically significant.

Table 6. Evaluation results for each model.

Model	Metric	Multi-class		Binary	
		PCAP-derived CSV	UNSW-NB15 CSV	PCAP-derived CSV	UNSW-NB15 CSV
Decision Tree	ACC	$0.977 \pm 2e-4$	$0.98 \pm 3e-4$	$0.984 \pm 2e-4$	$0.99 \pm 1e-4$
	AUC	0.727 ± 0.008	0.776 ± 0.003	0.86 ± 0.002	$0.943 \pm 7e-4$
	Precision*	0.477 ± 0.017	0.562 ± 0.007	0.872 ± 0.002	$0.95 \pm 8e-4$
	Recall*	0.481 ± 0.016	0.542 ± 0.005	0.86 ± 0.002	$0.943 \pm 7e-4$
	F1-score*	0.478 ± 0.014	0.548 ± 0.02	0.866 ± 0.002	$0.946 \pm 7e-4$
	Param.	$96.3\ K \pm 271$	$55.5\ K \pm 641$	$73.5\ K \pm 1.41\ K$	$35.8\ K \pm 850$
	Sample/s	$2.51M \pm 69\ K$	$1.99M \pm 58.4\ K$	$2.9M \pm 157\ K$	$2.15M \pm 94.3\ K$
Random Forest	ACC	$0.982 \pm 1e-4$	$0.983 \pm 1e-4$	$0.986 \pm 1e-4$	$0.993 \pm 1e-4$
	AUC	0.946 ± 0.007	0.936 ± 0.002	$0.994 \pm 3e-4$	$0.999 \pm 1e-4$
	Precision*	0.608 ± 0.017	0.619 ± 0.01	$0.893 \pm 7e-4$	$0.966 \pm 4e-4$
	Recall*	0.502 ± 0.006	0.557 ± 0.007	0.883 ± 0.002	$0.955 \pm 9e-4$
	F1-score*	0.537 ± 0.008	0.579 ± 0.007	0.888 ± 0.001	$0.961 \pm 6e-4$
	Param.	$5.24M \pm 15.1\ K$	$3.78M \pm 11\ K$	$3.69M \pm 8.91\ K$	$2.15M \pm 8.79\ K$
	Sample/s	$280\ K \pm 9.8\ K$	$240\ K \pm 5.46\ K$	$537\ K \pm 24\ K$	$569\ K \pm 21.5\ K$
MLP	ACC	0.956 ± 0.002	$0.972 \pm 5e-4$	$0.977 \pm 6e-4$	$0.986 \pm 4e-4$
	AUC	0.991 ± 0.001	$0.997 \pm 1e-4$	$0.996 \pm 2e-4$	$0.999 \pm 1e-4$
	Precision*	0.316 ± 0.005	0.487 ± 0.006	0.789 ± 0.003	0.888 ± 0.002
	Recall*	0.695 ± 0.008	0.706 ± 0.009	$0.986 \pm 4e-4$	$0.992 \pm 2e-4$
	F1-score*	0.355 ± 0.008	0.516 ± 0.008	0.86 ± 0.003	0.933 ± 0.002
	Param.	145.2 K	155.4 K	142.8 K	153.1 K
	Sample/s	$19.8\ K \pm 2.07\ K$	$19.8\ K \pm 1.68\ K$	$21.4\ K \pm 270$	$21\ K \pm 152$
XGBoost	ACC	$0.975 \pm 2e-4$	$0.977 \pm 3e-4$	$0.982 \pm 1e-4$	$0.989 \pm 2e-4$
	AUC	$0.996 \pm 47e-4$	$0.997 \pm 1e-4$	$0.997 \pm < 1e-4$	$0.999 \pm < 1e-4$
	Precision*	0.481 ± 0.014	0.584 ± 0.008	$0.823 \pm 8e-4$	0.908 ± 0.001
	Recall*	0.674 ± 0.009	0.669 ± 0.007	$0.984 \pm 3e-4$	$0.99 \pm 4e-4$
	F1-score*	0.547 ± 0.009	0.607 ± 0.007	$0.885 \pm 7e-4$	$0.945 \pm 8e-4$
	Param.	$1.93M \pm 7.48\ K$	$1.68M \pm 17.9\ K$	$263\ K \pm 2.41\ K$	$243\ K \pm 2.74\ K$
	Sample/s	$364\ K \pm 9.7\ K$	$246\ K \pm 4\ K$	$1.43M \pm 79.4\ K$	$847K \pm 42.1\ K$

*The Precision, Recall and F1-score metrics are calculated as macro averages, executed on A100 GPU.

Table 7. McNemar test results for model-performance comparison.

Comparison	Multi-class		Binary	
	Statistic	p-value	Statistic	p-value
Decision Tree *vs.* Random Forest	3,158.27	< 0.001	1,439.14	< 0.001
Decision Tree *vs.* MLP	26,816.13	< 0.001	4,090.86	< 0.001
Decision Tree *vs.* XGBoost	863.23	< 0.001	218.52	< 0.001
Random Forest *vs.* MLP	36,677.67	< 0.001	7,897.61	< 0.001
Random Forest *vs.* XGBoost	5,241.43	< 0.001	1,895.01	< 0.001
MLP *vs.* XGBoost	34,287.12	< 0.001	9,397.44	< 0.001

The statistical comparison between the proposed PCAP-derived dataset and the original UNSW-NB15 representation, conducted using XGBoost across 10 folds of Stratified K-fold cross-validation, is summarized in Table 8. The Wilcoxon Signed-rank test confirms that the performance variations between both datasets are statistically significant ($p = 0.0019$) across all evaluated metrics, validating that these differences are consistent and not a result of stochastic variance.

Table 8. Wilcoxon Signed-rank test results comparing PCAP-derived and UNSW-NB15 datasets' performance across metrics.

Metric		Multi-class			Binary		
		PCAP-derived	UNSW-NB15	p-value	PCAP-derived	UNSW-NB15	p-value
Accuracy		0.9746	0.9766	0.0019	0.9822	0.9887	0.0019
AUC		0.9962	0.9973	0.0019	0.9966	0.9993	0.0019
F1 Macro		0.5474	0.6073	0.0019	0.8852	0.9442	0.0019
T4	Train Time (s)	180.0457	236.9737	0.0019	20.8764	28.3762	0.0019
	Latency (ms)	0.0142	0.0115	0.0019	0.0014	0.0024	0.0019
A100	Train Time (s)	62.7990	77.8722	0.0019	11.1072	13.6984	0.0019
	Latency (ms)	0.0036	0.0052	0.0019	0.0008	0.0013	0.0019

* Hardware platforms are denoted as T4 (NVIDIA Tesla T4 GPU) and A100 (NVIDIA A100-SXM4-40GB GPU).

As previously noted, the original UNSW-NB15 dataset maintains a slight advantage in detection metrics. Specifically, Accuracy and AUC differences remain within approximately 0.2% and 0.1% for multi-class tasks and 0.6% and 0.3% for binary classification, respectively. While a more noticeable trade-off of approximately 6% is observed in the F1-score, the PCAP-derived representation offers substantial gains in computational efficiency. On the T4 GPU architecture, training time was reduced by 24.0% for multi-class and 26.4% for binary classification, a trend that persists on the A100 architecture with reductions of 19.3% and 18.9%, respectively.

A notable exception occurs in the multi-class inference latency on the T4 platform, where the PCAP-derived dataset exhibited higher latency (0.0142 ms) compared to the original set (0.0115 ms). This behavior is attributed to the increased structural complexity of the model: to compensate for the reduced feature space, the XGBoost algorithm generated deeper trees to capture underlying traffic patterns. Specifically, the PCAP-derived model required 2,008,476 total nodes, a 7.3% increase over the 1,871,454 nodes produced by the official dataset. This higher node density leads to increased memory pressure and cache misses on the T4's memory hierarchy, which has more limited resources. However, this effect is mitigated on the A100 platform; its significantly larger L2 cache and superior memory bandwidth allow for more efficient traversal of complex tree structures, ultimately achieving latency reductions of up to 38% in binary tasks.

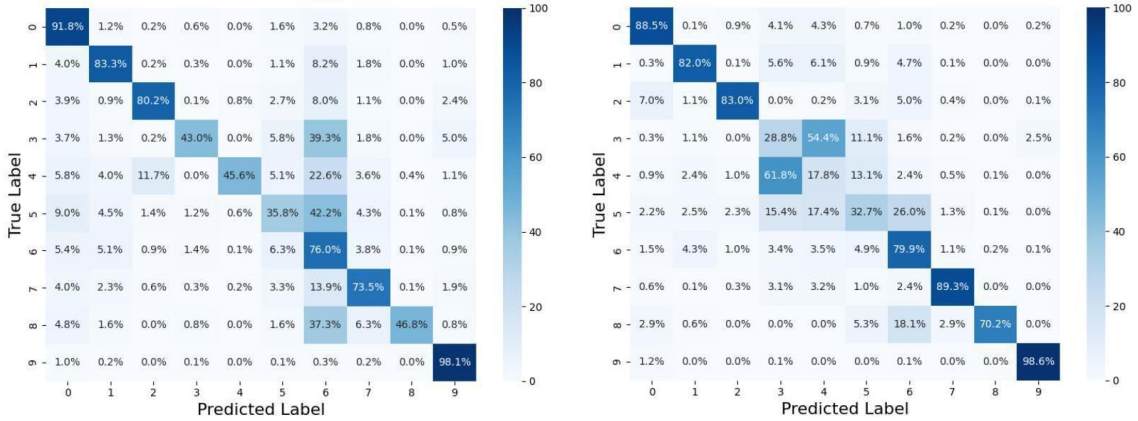
Compared with the studies in Related Work in Table 1, our approach is competitive at a lower computational cost. This is explained by the smaller number of classes and the reduced size of PCAP-derived CSV relative to the official datasets, without significantly sacrificing performance.

Additionally, the confusion matrix for the XGBoost model evaluated on the PCAP-derived dataset (Figure 3a) exhibits a clearer diagonal trend across several categories, indicating a redistribution of classification performance compared to the original representation. However, a detailed analysis of class 6 (Exploits) reveals a performance trade-off: while the model demonstrates a moderate recall of 76% in identifying this specific class, its precision is compromised due to misclassifications originating from other categories, primarily class 3 (Analysis), 5 (DoS) and 8 (Worms).

In contrast, the official UNSW-NB15 dataset (Figure 3b) displays significant overlap across several classes and substantially higher false positive rates. For instance, class 4 (Backdoor) is mislabeled as class 3 at a rate of 61.8% and conversely, class 3 is also mislabeled as class 4 in 54.4% of cases. Overall, classes 3, 4 and 5 (Analysis, Backdoor and DoS, respectively) exhibit the most critical false-positive rates, as the model largely fails to discriminate between them.

In Figure 3a, class 6, representing malicious Exploit traffic, acts as the primary source of ambiguity for the multi-class classifier. This category induces significant noise, elevating the overall false-positive

rate and diminishing the model's discriminatory power across the dataset. Given this pattern, it may be advisable to evaluate the exclusion of this class from the general multi-class scheme or its isolation into an independent detector to strictly control false-positive rates. Although such a modular architecture increases computational overhead and operational complexity compared to a single multi-class classifier, this trade-off is justifiable when specific class overlaps significantly degrade the system's reliability.



(a) Confusion matrix (PCAP-derived). (b) Confusion matrix (UNSW-NB15).

Figure 3. Confusion matrix for the XGBoost multi-class classifier: (a) Evaluated on PCAP-derived CSV; (b) Evaluated on UNSW-NB15 CSV. Class indices: 0: Fuzzers, 1: Reconnaissance, 2: Shellcode, 3: Analysis, 4: Backdoor, 5: Dos, 6: Exploits, 7: Generic, 8: Worms, 9: Normal.

The per-class F1-score, illustrated in Figure 4, provides a more comprehensive assessment of the model's ability to balance precision and recall across the unbalanced categories of the UNSW-NB15 dataset. The results reveal a complex trade-off: the PCAP-derived representation significantly improves performance in the most critical minority classes where the official dataset shows its worst results. Notably, the F1-score for Backdoor increased from a negligible 0.09 to 0.45 and Analysis rose from 0.12 to 0.20 , suggesting that the reduced feature set captures more effective patterns for these specific threats.

Conversely, the reduction in features leads to a performance decline in other categories. The most substantial drop is observed in the Generic class, where the F1-score decreased from 0.93 to 0.52, alongside moderate reductions in Reconnaissance (0.85 to 0.68), Exploits (0.82 to 0.68) and Worms (0.66 to 0.45). This indicates that while the custom PCAP-derived with 21 feature set mitigates the total failure of the model to detect certain stealthy attacks, it also sacrifices discriminative power in classes that were well-supported by the original UNSW-NB15 features.

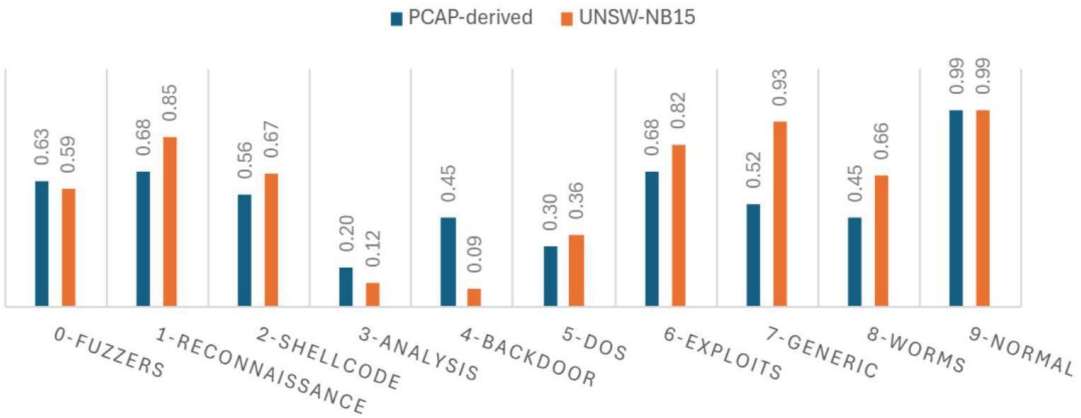


Figure 4. Bar chart showing per-class $f1$-score values for XGBoost model.

The performance of the proposed method compared to other recent works using the UNSW-NB15 dataset is presented in Table 9. This comparison focuses on accuracy metrics for both binary and multi-class tasks, where the proposed approach reports values of 98.60% and 98.20%, respectively. It is

important to clarify that these specific metrics correspond to the results achieved by the Random Forest model, which demonstrated the highest overall accuracy among all evaluated architectures in this study; however, this model is not necessarily the most suitable choice for scenarios requiring a balanced detection of minority classes. As previously discussed, although the Random Forest leads in raw accuracy, models like XGBoost provide a more robust and equitable performance across the entire attack spectrum.

Table 9. Performance comparison of the proposed approach with recent literature on the UNSW-NB15 dataset.

Reference	Accuracy (ACC)	
	Binary	Multi-class
[7]	97.90%	97.09%
[8]	97.00%	-
[9]	99.00%	-
[10]	97.93%	-
[12]	96.24%	-
[13]	97.49%	-
Proposed Work	98.60%	98.20%

The metrics obtained by our approach are highly competitive with recent literature. However, it is important to note that direct numerical comparisons with these state-of-the-art works serve as a contextual reference rather than an absolute benchmark, due to differences in evaluation protocols. While the referenced studies evaluate their models using the static, down-sampled official UNSW-NB15 test partition, our methodology necessitated a stratified 5-fold cross-validation on the full PCAP-derived capture to ensure an unbiased evaluation of our custom-flow construction. Although the main focus of this research was not to exhaustively maximize classification performance, but rather to evaluate the effectiveness of a reduced feature set within a lightweight and computationally efficient framework, the outcomes are highly satisfactory. The custom representation proves that it is possible to achieve state-of-the-art intrusion-detection accuracy while maintaining a reduced profile suitable for resource-constrained networks.

6. CONCLUSIONS

This study quantitatively evaluated a lightweight, network flow-based data representation constructed from the raw PCAP files of the UNSW-NB15 dataset, reducing the feature space from 49 to 21 attributes and achieving a file size 2.5 times smaller than the original. Experimental results demonstrated that this dimensional reduction offers an optimal trade-off for lightweight IDS deployments; the statistically superior XGBoost model experienced a minor 6% macro F_1-score reduction across both tasks, which was heavily compensated for by reduced training times and substantial inference speed increases of 68.8% in binary and 48% in multi-class classification.

Furthermore, this investigation revealed that the detection rate for specific attack classes varies significantly depending on how the dataset features are computed. Experimental results proved that certain features are highly effective for detecting specific attack categories, but less favorable for others. Quantitatively, this performance disparity is particularly evident in minority classes, such as Analysis and Backdoor, where the proposed methodology successfully increased the recall rates from 29% to 43% and from 18% to 46%, respectively. However, this localized improvement was accompanied by a performance trade-off, resulting in a reduction in detection accuracy for other categories, such as Generic and Worms.

These findings highlight that, in the design of robust IDSs, it is highly recommended to combine multi-class classification models with specialized binary classifiers dedicated to those specific classes where the multi-class model faces severe identification challenges. Such a hybrid approach maintains an optimal balance between classification performance and computational overhead, which is critical for

environments with strict operational constraints, such as edge nodes or IoT gateways. This structural recommendation aligns with the multi-stage architecture proposed by [15], which effectively mitigates class imbalance and enhances the overall robustness of the detection system.

Consequently, future research will explore class-specific feature-selection mechanisms to determine the optimal sub-set of attributes for each attack family. Specifically, subsequent analyses will focus on how variations in individual features impact detection accuracy across different threat categories. The objective is to isolate the most discriminative attributes required for precise classification while strictly limiting the size of the feature set to mitigate the noise and redundancy inherent to high-dimensional data.

REFERENCES

[1] S. Sinha, "State of IoT 2025: Number of Connected IoT Devices Growing 14% to 21.1 Billion Globally," IoT Analytics, [Online], Available: https://iot-analytics.com/number-connected-iot-devices/, 2025.

[2] T. Fox, "Cybercrime to Cost the World $12.2 Trillion Annually by 2031," Cybersecurity Ventures, [Online], Available: https://cybersecurityventures.com/official-cybercrime-report-2025/, 2025.

[3] N. Moustafa and J. Slay, "UNSW-NB15: A Comprehensive Dataset for Network Intrusion Detection Systems (UNSW-NB15 Network Dataset)," Proc. of the IEEE 2015 Military Communications and Information Systems Conf. (MilCIS), pp. 1-6, Canberra, ACT, Australia 2015.

[4] S. More, M. Idrissi, H. Mahmoud and A. T. Asyhari, "Enhanced Intrusion Detection Systems Performance with UNSW-NB15 Data Analysis," Algorithms, vol. 17, no. 2, p. 64, 2024.

[5] M. Ahmad et al., "Intrusion Detection in Internet of Things Using Supervised Machine Learning Based on Application and Transport Layer Features Using UNSW-NB15 Data-set," EURASIP Journal on Wireless Communications and Networking, vol. 2021, no. 1, p. 10, 2021.

[6] M. Z. Hussain, A. Iftikhar, T. N. Usmani and M. Z. Hasan, "Leveraging Zero Trust Architecture for Network Intrusion Detection: A Comprehensive Evaluation Using the UNSW-NB15 Dataset," Spectrum of Engineering Sciences, vol. 3, no. 3, pp. 669-676, 2025.

[7] M. Jouhari, H. Benaddi and K. Ibrahimi, "Efficient Intrusion Detection: Combining $\chi 2$ Feature Selection with CNN-BiLSTM on the UNSW-NB15 Dataset," arXiv preprint, arXiv: 2407.14945, 2024.

[8] V. Sharma and M. Kumar, "Improving Intrusion Detection with Hybrid Deep Learning Models: A Study on CIC-IDS2017, UNSW-NB15 and KDD CUP 99," Journal of Information Systems Engineering and Management, vol. 10, no. 11, DOI: 10.52783/jisem.v10i11s.1665, 2025.

[9] A. D. Vibhute, M. Khan, C. H. Patil, S. V. Gaikwad, A. V. Mane and K. K. Patel, "Network Anomaly Detection and Performance Evaluation of Convolutional Neural Networks on UNSW-NB15 Dataset," Procedia Computer Science, vol. 235, pp. 2227-2236, 2024.

[10] M. Farhan et al., "Network-based Intrusion Detection Using Deep Learning Technique," Scientific Reports, vol. 15, no. 1, p. 25550, 2025.

[11] Z. Chkirbene, S. Eltanbouly, M. Bashendy, N. AlNaimi and A. Erbad, "Hybrid Machine Learning for Network Anomaly Intrusion Detection," Proc. of 2020 IEEE Int. Conf. on Informatics, IoT and Enabling Technologies (ICIoT), pp. 163-170, Doha, Qatar, 2020.

[12] M. H. Kabir et al., "Network Intrusion Detection Using UNSW-NB15 Dataset: Stacking Machine Learning Based Approach," Proc. of the 2022 IEEE Int. Conf. on Advancement in Electrical and Electronic Engineering (ICAEEE), pp. 1-6, Gazipur, Bangladesh, 2022.

[13] M. Belouch, S. El Hadaj and M. Idhammad, "Performance Evaluation of Intrusion Detection Based on Machine Learning Using Apache Spark," Procedia Computer Science, vol. 127, pp. 1-6, 2018.

[14] I. Mutambik, "An Efficient Flow-based Anomaly Detection System for Enhanced Security in IoT Networks," Sensors, vol. 24, no. 22, p. 7408, 2024.

[15] M. M. Mahmoud, Y. O. Youssef and A. A. Abdel-Hamid, "XI2s-IDS: An Explainable Intelligent 2-stage Intrusion Detection System," Future Internet, vol. 17, no. 1, p. 25, 2025.

ملخص البحث:

أدى انتشار البنى التحتية للشّبكات المترابطة وأجهزة إنترنت الأشياء إلى توسيع نطاق الهجمات الإلكترونية بشكلٍ كبير، مما يستدعي وجود أنظمة فعّالة للكشف عن الاختراقات تعتمد على التّعلُّم الآلي. وعلى الرغم من وجود مجموعات بيانات مرجعية، فإنّ خصائصها الرّسمية تفرض قيوداً فيما يتعلق بالمرونة وعدم توازن البيانات.

تعمل هذه الدّراسة على تقييم أثر تمثيل بيانات مخصَّص من خلال إنشاء مجموعة بيانات جديدة من ملفّات PCAP. وقد طبّقنا آلية عملٍ لتصنيف الحزم وتجميع التّدفقات أحادية الاتّجاه، واستخراج مجموعةٍ مصغّرة من 21 ميزة، وقارنا هذا التّمثيل مع مجموعة البيانات الرّسمية الّتي تضمّ 49 ميزة باستخدام بنى تعلّم آلي مختلفة في مهامّ التّصنيف الثّنائي ومتعدّد الفئات.

وتشير النّتائج إلى أنّ مجموعة البيانات المخصّصة تحقّق أداءً تنافسياً على الرّغم من الانخفاض الكبير في حجم الملفّ وعدد الميزات. والجدير بالذّكر أنّ التّمثيل المخصّص يوازن بفاعلية بين دقّة الكشف والكفاءة الحسابية، ممّا يوفّر استراتيجية فعّالة للبيئات ذات القيود التّشغيلية الصّارمة، مثل عُقَد الحافة او بوابات إنترنت الأشياء.