Computer Modeling in Engineering & Sciences

Tech Science Press

https://doi.org/10.32604/cmes.2026.083958

ARTICLE

DDoS Defense Model on 5G Network Slices

Kun-Lin Tsai[1], **Shih-Ting Chiu**[2], **Chihhsiong Shih**[2] **and Fang-Yie Leu**[2,*]

[1]Electrical Engineering Department, Tunghai University, Taichung, Taiwan
[2]Computer Science Department, Tunghai University, Taichung, Taiwan

*Corresponding Author: Fang-Yie Leu. Email: leufy@thu.edu.tw

Received: 14 April 2026; Accepted: 17 June 2026; Published: 28 August 2026

ABSTRACT: With the quick development of 5G networks, network slicing and Open Radio Access Network (O-RAN) have become key technologies for improving network resource-allocation efficiency and flexibility. However, network slicing also faces intrusion-detection challenges, particularly for detecting DDoS attacks, which are difficult to detect due to traffic being silently transmitted across multiple sub-slices. To address this problem, this paper proposes a 5G network slicing intrusion detection mechanism, called the DDoS Defense Model on 5G Network Slices (2D5NS) which integrates machine learning and real-time traffic monitoring techniques to detect and mitigate DDoS attacks within an O-RAN. This security system consists of a Random Forest (RF) classification model, which is deployed within the Service Management and Orchestration (SMO) of O-RAN to classify packets transmitted from UE to the RAN into eMBB, mMTC and uRLLC slices, and a detection approach comprising the XGBoost mechanism which monitors the traffic within each slice in real time to detect DDoS attacks issued by User Equipment (UE). Once traffic is abnormal, it triggers an Entropy Algorithm to identify the sources of the DDoS attacks. The simulation results of our second experiment show that the classification accuracies of RF classification model in its 3-fold Cross Validation (CV) for eMBB and mMTC training achieve 99.98%. In our third experiment, the detection accuracy of 2D5NS/XGBoost model on uRLLC reaches at least 93.43%. Several state-of-the-art systems are evaluated. Here, the conclusion is that the 2D5NS outperforms each of them and the 2D5NS can effectively mitigate and block DDoS attacks for network slices.

KEYWORDS: 5G/6G network; intrusion detection; random forest; XGBoost; O-RAN; entropy analysis

1 Introduction

In recent years, people have request high-efficiency and high-speed 5G/6G (hereinafter referred to as 5G) networks, wishing to enrich and color their everyday lives. To achieve this, network equipment and facilities need to be constantly upgraded. Network Slice [1] and Open Radio Access Network (O-RAN) [2] have emerged in response to this. The former allows network resources to be used more efficiently. Due to its isolation characteristics, users can utilize different network resources at the same time in the same physical network system without interfering with each other. Network slices as denoted by S-NSSAI (Single Network Slice Selection Assistance Information) and slice types can be distinguished by Slice/Service type (SST) [1,3].

In the O-RAN architecture, the Non-Real-Time RAN Intelligent Controller (Non-RT RIC) equipped in the Service Management and Orchestration (SMO) function and the Near-Real-Time RAN Intelligent Controller (Near-RT RIC) enable the deployment of machine learning or artificial intelligence (ML/AI) models [2] to autonomously allocate bandwidth and QoS policies for improving packet delivery efficiency and enhancing traffic control capabilities. However, due to the quick advances of network technology, hackers' intrusion techniques and knowledge follow. That is why network attacks have never decreased,

and DDoS attacks are always around us [4,5]. In other words, the defense of network systems must be continuously improved. Generally, it is very important to detect attacks early, defend against them as soon as possible, and mitigate the power of network attacks immediately.

On the other hand, network slices are functionally isolated from each other. If a slice, e.g., slice i, is attacked and its traffic is heavy, the traffic of other slices, for example, slice $j, i \neq j$, may still be low. The IDS/IPS that monitors the entire network traffic to detect DoS/DDoS attacks on the physical network or components may be unable to discover it. Singh et al. [3] mentioned that 5G network slices must be prevented from DoS and DDoS attacks before they can serve users efficiently. Ref. [6] reported that AI-driven automation is transforming the way applications and APIs are accessed. They are now accounting for the majority of global web traffic. As agentic AI conducts new classes of automated traffic and most are not benign, meaning that the Intent attacks are harder to detect, distinguish, and control than before. Radware 2026 Global Threat Report [7] shows DDoS attacks jump 168% as cyber threats escalate across networks and applications.

Polese et al. [8] found that several limitations related to SMO can be found in the literature, e.g., the defense coordination and deployment of intelligent strategies in SMO have not been fully described, explained and implemented. As a result, when network slices face dynamic attacks, such as DDoS, the solution tends to be with static encryption, lacking intelligent real-time detection and response capabilities which need to be continuously enhanced to effectively protect network systems.

To solve these problems, in this paper, a security mechanism, named the DDoS Defense Model on 5G Network Slices (2D5NS), that detects DDoS attacks on O-RAN, was proposed. The 2D5NS deploys machine learning (ML) techniques, such as the Random Forest (RF) model in the SMO,to classify the slice to which a packet entering the base station belongs. Furthermore, it monitors the operational status of all slices in the network in real time by using the XGBoost (eXtreme Gradient Boosting) model. This allows the rapid identification of the attack sources when an attack occurs, and the attack is then blocked according to the QoS rules of each slice. Our previous research results were published in [9], in which entropy is calculated based on IP connection frequency for identifying hackers. This study inherits this calculation.

The contributions of this research are summarized as follows.

(1) When an incoming packet p arrives, it is proactively classified into the corresponding slices by using an ML technique since the 2D5NS detects DDoS attacks for network slices.
(2) When there is a suspected attack, the 2D5NS first identifies the source IPs, reduces each of their bandwidths to 80%, and analyzes the entropy of the corresponding QoS flows [9]. Once the attack is sure, the source IP (Src IP) addresses of the DDoS attack will be blocked.
(3) Our "the time to first detection" is defined as the time period from the time point when an attack starts to the time point when the attack is discovered, and is shorter than 3.68 ms. It means the proposed system can discover an attack in a very short time after the malicious behavior starts.

The research limitations are that hackers may employ many UEs to DDoS attack a BS. But after sending DDoS traffic, UEs disappear from the system, i.e., UEs are randomly chosen to connect to the BS and transmit insufficient traffic, or the inter-connection time for UEs is long, e.g., several hours or several days. In this case, those Src IPs cannot be identified since they do not provide enough information for the 2D5NS to calculate their entropies. This case is not within the scope of this study. Also, the DDoS that consumes computer resources is not our focus. Furthermore, new DDoS approaches may come out at any time. In this case, new training data is required.

The rest of this paper is structured as follows. Section 2 reviews the relevant literature and backgrounds of this research. Section 3 describes the main system architecture and attack detection methods of the 2D5NS.

Experimental results and functional verification are presented in Section 4. Section 5 concludes this study and addresses our future work.

2 Related Work and Background

This section will review some existing studies, and security mechanisms related to this study.

2.1 Literature Review

To address DDoS attacks in 5G networks, researchers have proposed various approaches ranging from deep learning to architectural optimizations. Majeed et al. [10] utilized a Convolutional Neural Network (CNN) to identify abnormal network traffic, enabling early detection and traffic reduction to mitigate attack impact. However, as 5G networks evolve, the focus has shifted towards the vulnerabilities inherent in Network Slicing. A recent comprehensive survey by De Alwis et al. [11] highlights that despite logical isolation, network slices remain vulnerable to resource exhaustion attacks. Furthermore, Allaw et al. [12] simulated slicing within an SDN architecture, using neural networks to pre-divide traffic into logical networks (eMBB, audio and web text) and utilizing OpenFlow for frequency band division. While these studies addressed classification, they do not fully tackle the real-time mitigation of attacks within the O-RAN architecture.

Regarding slice management, there is currently no consensus on standard classification methods based on existing protocols. Wu et al. [13] proposed a traffic-level classification, categorizing applications into lightweight (e.g., HTTP), mixed (e.g., Facebook), and heavyweight slices (e.g., Google Drive).

To extend defense mechanisms for network edges, scholars have deployed ML models on the RAN Intelligent Controller (RIC) of the Service Management and Orchestration (SMO). The adoption of O-RAN introduces new security paradigms; Soleymani et al. [14] demonstrated that deploying DDoS detection logic as an xApp on the Near-RT RIC allows for mitigation at the network edge, preventing malicious traffic from saturating the backhaul. Complementing this, El-Hajj [15] proposed a Zero-Trust and Federated Learning framework, emphasizing that the open nature of O-RAN requires robust, embedded security optimization for 6G networks.

Several studies have implemented specific models within this architecture. Abou El Houda et al. [16] proposed a Federated Deep Reinforcement Learning technology for O-RAN to mitigate jamming attacks, leveraging the local nature of federated learning to ensure privacy while improving detection accuracy across multiple nodes. Tsourdinis et al. [17] presented an ML/AI-driven framework with an Anomaly Traffic Detector (ATD) xApp, which dynamically adjusts resources under attack, reducing CPU usage by 15%. Addressing the Open Fronthaul (OFH) vulnerability, Chang et al. [18] proposed an IDS for the CUS-Plane using a packet continuity-based method, finding that a CNN-LSTM model excels in binary classification tasks even with reduced feature sets.

For advanced threat landscapes in 5G-Advanced IoT, Baidar et al. [19] developed “Precision AI”, a hybrid framework combining 1D-CNN and BiLSTM with Federated Learning, achieving high separability (AUC $\approx$ 0.996, will describe later) suitable for uRLLC requirements. To support these data-driven approaches, Zadeh et al. [20] introduced the NetsLab-5GORAN-IDD dataset, capturing both packet-level and radio telemetry data from a live O-RAN testbed. Additionally, Liao et al. [21] proposed the RANGAN framework, utilizing Generative Adversarial Networks (GANs) and transformers to capture temporal dependencies, achieving an F1-score of 83% in identifying network contention.

Despite these advancements, most existing studies focus on either general O-RAN security or static slice classification [5]. There is a lack of integrated research that specifically addresses intrusion detection within network slices using the real-time capabilities on O-RAN.

2.2 XGBoost

XGBoost [22] is a powerful tree-based ensemble learning algorithm built upon the gradient boosting framework. Its primary strategy is to improve predictive performance by sequentially combining multiple "weak learners", which are typically decision trees. The core of XGBoost is an additive model designed to minimize a specific objective function that includes a regularization term. This design creates a balance between model accuracy and generalization. The objective function is expressed as Eq. (1):

$$Obj(\theta) = L(\theta) + \Omega(\theta) \tag{1}$$

where $L(\theta)$ represents the loss function, and $\Omega(\theta)$ is the regularization term which penalizes the model's complexity, prevents overfitting and enhances the model's robustness on unseen data.

For optimizing its objective function, XGBoost employs a second-order Taylor expansion, a technique similar to Newton's method. This allows the algorithm to use both first-order (gradient) and second-order (Hessian) statistics when determining the best splits in a tree.

2.3 Entropy

Shannon expressed the degrees of information uncertainty by using entropy in physics, which is called information entropy [9,23]. Assuming that there is a discrete random variable $x = \{x_1, x_2, x_3, \ldots, x_n\}$, with a total of n different values. Let $p(x_i)$ be the probability of the random variable x_i, then the entropy $H(x)$ [9,23] of x is (see Eq. (2)):

$$H(x) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i) \tag{2}$$

This shows the relationship between probability and information content with low (high) probability events carrying more (less) information.

2.4 Network Slicing

Network slicing is an innovative technology for 5G wireless networks that allows operators to create multiple virtualized and independently running network instances (slices) on a shared physical infrastructure/component to meet the needs of different application requirements. Zhang [24] found that current developed network slicing can be customized for specific services, such as enhanced mobile broadband (eMBB), massive machine-type communications (mMTC), and ultra-reliable low-latency communications (uRLLC).

In a 5G core network, the Network Slice Selection Function (NSSF) selects an appropriate slice and the corresponding AMF based on the UE's subscription information in the Unified Data Management (UDM). At the RAN, the appropriate slice is selected to handle the user's radio resources according to relevant information provided by the UE or the core network, such as S-NSSAI.

2.5 Random Forest

Random Forest (RF) [25,26] is an ensemble learning method based on decision trees that can be used to classify data and perform regression analysis. Its main mechanisms include bootstrap sampling and

random feature selection. The former samples training data from D = $\{(x_1, y_1), (x_2, y_2), \ldots, (x_i, y_i)\}$ by using bootstrap sampling method, and a total of k samples are selected to train a decision tree where x_j is sample j and y_j the label of x_j, $k \leq i, 1 \leq j \leq i$. The latter, the random feature selection, randomly selects m features before training each decision tree, $m < j$, where j is the total number of system features. Then, the optimal splitting feature for node N is chosen using Gini Impurity or Information Gain [25], so that the data can be split into N's child nodes $N_{s,child} = \{N_1, N_2, \ldots, N_t\}$.

For classification problems, a majority decision determines the predicted result (see Eq. (3)).

$$\hat{y} = \arg\max_{k} \sum_{j=1}^{T} I(h_j(x) = k) \tag{3}$$

where T is the number of decision trees, $h_j(x)$ is the predicted result of the jth decision tree, $1 \leq j \leq T$, and $I(\cdot)$ is a function, indicating whether a certain condition is met. For regression, RF adopts the average prediction values of all decision trees as the result for rising its generalization ability and robustness to noise.

3 2D5NS System Architecture

Fig. 1 shows the 2D5NS architecture. Upon system startup, the SMO delivers our AI/ML model, i.e., RF-based packet classification model M (RF classification model or just RF model) to the Near-RT RIC and then sends relevant parameters, like slice-specific QoS profiles, traffic steering policies, and admission control rules, to Central Unit (CU). When a packet p from UE is transmitted to the base station, it is passed from Radio Unit (RU) to Distributed Unit (DU) where analog signals are converted to digital. The DU then passes p to the CU which then mirrors p into the RF model M via xAPP [27]. M in the CU classifies p to the slice it belongs based on its Differentiated Service Code Point (DSCP) value. The IDS/IPS then counts and accumulates traffic. This method individually calculates traffic for all QoS flows (flows for short) of all PDU sessions passing through a slice in the CU to detect attacks.

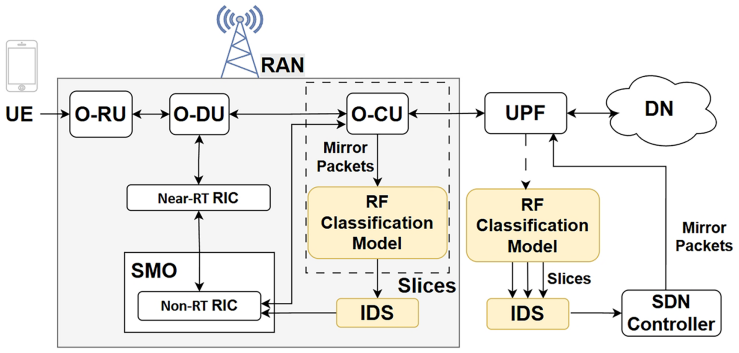

Figure 1: 2D5NS system architecture.

3.1 Slice Classification

Currently, network standards lack clear definitions on the fact that which slice packet should map to which specific network protocol (e.g., IP or port #). In this study, a custom-coded DSCP field for traffic classification is used to map existing application protocols to the 5G QCI/5QI (see Table 1), thereby redefining the DSCP values to the corresponding application protocols.

Table 1: The mapping between NSSAI_Type and DSCP definition.

NSSAI_Type	DSCP's First 3 Bits	Description
eMBB	100	Representing eMBB
mMTC	010	Representing mMTC
uRLLC	001	Representing uRLLC
eMBB + uRLLC	101	For example, transmitting images and life signs of patients lying on the bed for telesurgery
mMTC + uRLLC	011	For example, an emergency shutdown failure in a smart factory triggers immediate control instructions.

The DSCP field has 6 bits defined as follows. The first 3 bits are the NSSAI tag, i.e., bit 0 = 1 for eMBB, bit 1 = 1 for mMTC, and bit 2 = 1 for uRLLC (Note that if V2X and HMTC (High performance machine-type communication) need to be considered, the coding approach ought to be enhanced. Currently, only the first three slices are dealt with in this study). The last 3 bits represent the priority of the slice; a larger value is given a higher priority.

Table 1 shows the mapping between NSSAI_Type and DSCP definitions. However, due to the large number of protocols and applications in the world, it is hard to completely list all of them here and map them to any of the three types of slices. Also, some applications require more than two types of slices, such as industrial automation, intelligent transportation systems, and augmented reality. Thus, Table 1 also defines hybrid slices, e.g., 101 represents eMBB + uRLLC, 011 shows mMTC + uRLLC, and so on.

Table 2 shows the QCI/5QI values for each application category involved in this study, and lists the DSCP values defined. The priority levels (the last three bits of DSCP) are specified based on the application category's default priority, packet loss rate, and latency. mMTC communications do not have a specific 5QI type. Then, the 5QI values, including 11–13, are expanded, corresponding to MQTT, CoAP, and LwM2M, respectively, since they are not defined in 3GPP TS 23.501 [28]. A 5QI value of 9, NSSAI_Type = Others, categorizes traffic as the Best Effort with the DSCP value 000000_2. Table 2 is only an example; users may modify it following this encoding principle and/or define specific DSCP values for more applications.

Table 2: Correspondence between 5QI value, application category and DSCP value.

NSSAI_Type	5QI	DSCP Value	GBR Type	Application Category
uRLLC	1	001 010	GBR	Voice conversation
uRLLC	3	001 100	GBR	Real-time games, V2X
uRLLC	65	001 011	GBR	Critical message transmission for multimedia
uRLLC	67	001 101	GBR	Mission-critical video
mMTC	11	010 011	Non-GBR	MQTT (message queuing telemetry transport protocol)–undefined in 3GPP TS 23.501 [28]
mMTC	12	010 010	Non-GBR	CoAP (conventional application protocol)–undefined in [28]

(Continued)

Table 2 (continued)

NSSAI_Type	5QI	DSCP Value	GBR Type	Application Category
mMTC	13	010 000	Non-GBR	LwM2M (lightweight device communication)–undefined in [28]
eMBB	2	100 101	GBR	Live video
eMBB	4	100 011	GBR	Non-real-time video streaming
eMBB	6	100 000	Non-GBR	Web browsing, background downloading
Others	9	000 000	Non-GBR	Applied to best effort
eMBB + uRLLC	82	101 001	Critical GBR	Industrial automation + augmented reality (AR)
eMBB + uRLLC	83	101 010	Critical GBR	Discrete automation (V2X)
eMBB + uRLLC	84	101 011	Critical GBR	Intelligent transportation system

3.2 RF Classification Model Training

In 5G network slicing, misclassifying eMBB traffic as uRLLC (False Positive) wastes valuable high-priority resources, while missing uRLLC (False Negative) violates SLA.

To solve this problem, in this study, a confidence-based classification inference logic was implemented by involving the Precision-Recall curve (PR Curve) [29]. Let $Class_{pred}$ be the initial category predicted by the concerned model, and let $Prob_{uRLLC}$ be the probability of the uRLLC class. The final network slice determination, denoted as $NSSAI_{final}$, is defined according to the following rule.

$$NSSAI_{final} = \begin{cases} uRLLC, & if\ Class_{pred} = uRLLC\ and\ Prob_{uRLLC} \geq 0.9 \\ mMTC, & if\ Class_{pred} = mMTC \\ eMBB, & if\ Class_{pred} = eMBB \\ Others, & otherwise \end{cases}$$

This strict threshold of uRLLC ensures that a slice is allocated to uRLLC only when the model exhibits extremely high confidence. RF model is deployed in SMO with the following functions.

(1) With Table 2, a two-tiered mapping mechanism was established.
   (1) The first tier, Protocol Name to Application Category, categorizes the packet's protocol nadme (Protocol Name) into a higher-level Application Category. For example, instant messaging applications, such as SKYPE and TEAMSPEAK, are categorized as Voice Conversation (see the first row of Table 2 with 5QI = 1).
   (2) The second tier, Application Category to Slice and QoS, defines the corresponding NSSAI_Type and DSCP for each Application Category. For example, in the first row, the Voice Conversation category is mapped to the slice with NSSAI_Type = uRLLC and DSCP = 001010_2. All uncategorized applications are assigned to the NSSAI_Type of Others. The NSSAI_Type is the RF model's predicted label. Note that DSCP is safe since it is generated by the 2D5NS, rather than by UE. UE cannot forge its DSCP value, implicitly showing that UEs are untrustworthy and other components/sub-systems of the 2D5NS are trustable.

(2) To address the significant class imbalance (e.g., $|eMBB| \gg |uRLLC|$), a hybrid strategy was employed. First, SMOTE [30] was applied to augment each of the uRLLC and mMTC slices to 100,000 samples. Second, the class_weight = 'balanced' parameter was integrated into the RF model to dynamically adjust penalty weights during training.

Due to the large size of the original dataset Y with $|Y|$ samples, for shortening training time, a subset of samples, denoted by X_s, was extracted for a specific NSSAI_Type S from Y to form a representative subset of S by using Bootstrap Aggregation technique where $|X_s| < |S| < |Y|$ (see Eq. (4)).

$$|X_s| = \frac{|S|}{|Y|} \times |X| \tag{4}$$

where $|X|$ is the total number of samples extracted from Y for training, aiming to eliminate the bias that may be introduced by random sampling and ensure that $\frac{|X_s|}{|X|} = \frac{|S|}{|Y|}$.

$$|X| = \sum_{s=1}^{m} |X_s|, \cdots 1 \le S \le m \tag{5}$$

In Eq. (5), m is the number of the concerned network slices.

(3) To classify the behavior patterns among different application categories more effectively, three ratio features that quantify the bidirectional interaction of traffic and the overhead of data transmission were established.

(1) fwd_bwd_packet_ratio (FBPR, traffic directionality index): defined as the ratio of the total number of forward packets delivered from the sender to the receiver to the total number of backward packets in the same QoS flow S (see Eq. (6)).

$$FBPR = \frac{\text{Total no. of all forward packets sent}}{\text{Total no. of all backward packets sent}} \tag{6}$$

(2) fwd_bwd_bytes_ratio (FBBR, traffic load directionality index): defined as the ratio of the total number of bytes of all forward packets sent from the sender to the receiver over the total number of bytes of all backward packets in S (see Eq. (7)).

$$FBBR = \frac{\text{Total no. of bytes of all forward packets}}{\text{Total no. of bytes of all backward packets}} \tag{7}$$

(3) fwd_header_payload_ratio (FHPR, packet payload efficiency index): defined as the ratio of the header length of all forward packets to the total packet length in S (see Eq. (8)).

$$FHPR = \frac{\sum_{i=1}^{n} \text{Header length of packet } P \text{ delievered}}{\sum_{i=1}^{n} |P|} \tag{8}$$

where n is the number of S's packets flowing through the UPF or CU.

(4) The RF model M is trained on the subset X and Gini Impurity [25] is employed as the criterion for evaluating split quality. To solve the problem of uneven number of samples of different slices (e.g., |eMBB traffic| >> |uRLLC traffic|), which may cause the case that the RF model tends to favor the eMBB. This study adopts penalty weight 1/|uRLLC traffic| as the cost of misclassifying an uRLLC packet which is larger than 1/|eMBB traffic|, i.e., the penalty of misclassifying an eMBB packet.

The RF Network Slicing and Application_Category Training algorithm is summarized in Algorithm 1.

Algorithm 1: RF network slicing and application_category training

Input: Network traffic dataset (CSV format), Application Category to NSSAI_Type/DSCP mapping rules
Output: Trained RF model M (ONNX format)

Phase 1: Network Slicing Label Mapping

1. Adding Application Category and NSSAI_Type/DSCP fields based on their mapping rules.
2. Preprocessing data on those fields with infinite (inf) and missing values (NaN).

Phase 2: Feature Engineering

3. Merging similar applications to generate MergedProtocolName, like, Google and Gmail → Google_services.
4. Creating new proportional features $FBPR$, $FBBR$, $andFHPR$ (Eqs. (6)–(8)).
5. Log Transformation: Apply $y = \log(1 + x)$ to numerical features with high variance (e.g., Flow.Duration and Total.Length) where x represents the original raw feature value, thus compressing the vast range of data values, preventing extreme outliers from dominating the model training and ensuring better numerical stability.

Phase 3: Data Cleaning and Stratified Sampling

6. Removing categories with number of samples less than r, in this study, $r = 2$.
7. Defining feature matrix X and target label Y (i.e., NSSAI_Type).
8. Extracting 500,000 records via stratified sampling.
9. Constructing an Imbalanced-learn Pipeline integrating SMOTE and RF.

Phase 4: Training RF model Performing hyperparameter search

10. Executing RandomizedSearchCV with three-fold cross-validation to optimize hyperparameters (n_estimators = 100, max_depth = 20).
11. Analyze Precision-Recall curves to determine the optimal decision threshold τ_{opt} (set to 0.9) for maximizing uRLLC precision.
12. Training the RF model M using the optimal parameters and exporting it to ONNX format.

3.3 Data Identification and Anomaly Detection Procedure

The 2D5NS individually detects malicious behaviors on the QoS flows of all PDU sessions within a slice (see Fig. 2) by using XGBoost.

A packet p transmitted from the Data Network (DN) to the UPF is also duplicated to IDS through mirror port of an UPF to identify the QoS flow to which p belongs. The system employs a cross-layer tracking mechanism based on a quadruple Quad (see Eq. (9)).

$$\text{Quad} = (\text{Src/Dst IP},\ \text{PortNum},\ \text{QFI},\ \text{TEID}) \tag{9}$$

which integrates key identifiers from the application layer to the tunnel layer, and in which

(1) Src/Dst IP and PortNum: At the CU-UP layer, they are used to accurately identify the source and destination IP addresses of an application's QoS flows.
(2) QFI (QoS Flow Identifier): Directly mapping to p's QoS attributes, such as S-NSSAI and 5QI, to identify its Application_Category (the 5th column of Table 2) and priority (the last three bits of DSCP in the 3rd column).
(3) TEID (Tunnel Endpoint Identifier): In the GTP-U protocol, it is used to associate p with its corresponding Data Radio Bearer (DRB).

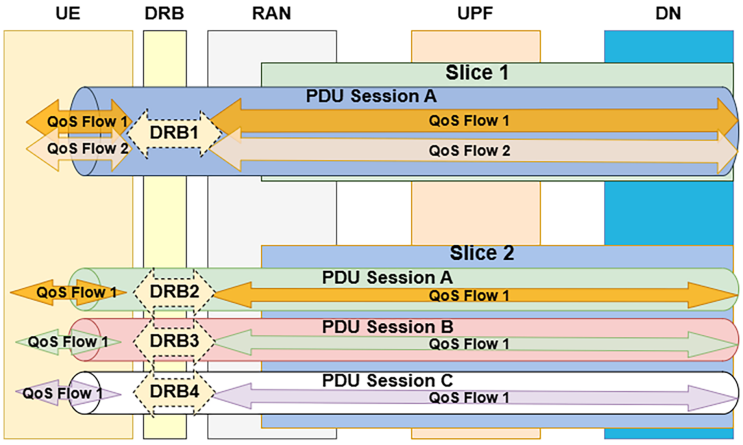

Figure 2: The relationship among network slices, PDU sessions and QoS flows.

3.3.1 Feature Selection for XGBoost

To maximize detection accuracy of XGBoost, the training data is pre-processed. However, high-dimensional network traffic often introduces the curse of dimensionality, leading to increased latency and model overfitting. To address this, a rigorous two-stage feature selection process was then implemented, systematically reducing the approx. 78 features of initial dataset to 16 critical features.

(1) Statistical Filtering by using Pearson correlation coefficient (PCC): In the first stage, the 2D5NS computed the PCC to quantify the relationship between each pair of features and the binary attack labels (benign or malicious). Only those attributes that demonstrated a statistical dependency with the target variables are retained.

(2) Domain-Knowledge Optimization (O-RAN Constraints): Features requiring Deep Packet Inspection (DPI) or complex string analysis (e.g., payload content matching) and metrics that cannot be directly retrieved from typical network monitoring mechanisms (such as typical flow monitoring sub-system) without inspecting the packets themselves are all removed.

After that, the remaining 16 features are strictly categorized into three dimensions essential for characterizing anomalies.

(1) Inter-Arrival Time (IAT) Statistics: Features, such as Flow IAT Mean, Flow IAT Std, and Fwd IAT Max that measure the temporal rhythm of the traffic, are crucial for detecting automated attacks (e.g., DDoS, Botnets), which often exhibit fixed, machine-like transmission frequencies distinct from the bursty nature of human behavior.

(2) Throughput and Volume Statistics: Flow Bytes/sec, Flow Packets/sec and Flow Duration that provide a holistic view of traffic intensity, are effective in identifying volumetric attacks or "low-and-slow" intrusions by monitoring abnormal surges in data rates or extended connection durations.

(3) Packet Length Variation: Features, like Fwd Packet Length Std and Fwd Packet Length Mean, can capture the variability of payload sizes. Since specific attack tools often generate packets of uniform size, the standard deviation of packet length serves as a powerful malicious-behavior indicator.

3.3.2 Detection Procedure

Our traffic management and packet classification procedure as shown in Fig. 3 starts when receiving coming packets. The RF Model classifies a packet p to an appropriate network slice, and assigns a DSCP value to p.

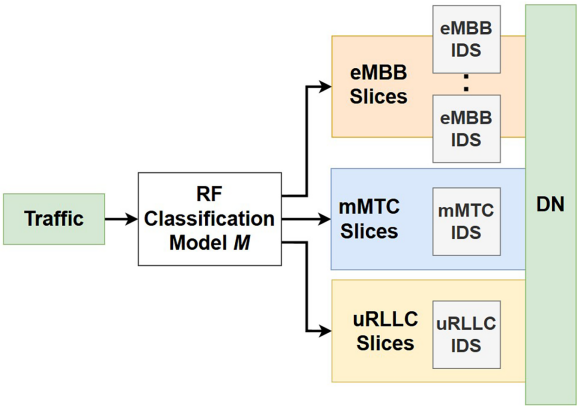

Figure 3: Traffic management and packet classification of 2D5NS (eMBB with multiple IDS).

Following that, the traffic is routed to its a slice-specific IDS. Fig. 4 shows the 2D5NS detection procedure, which is described as follows.

(1) The XGBoost model inspects packets within the window $W_{XGBoost}$, a memory space for XGBoost to inspect incoming packet, i.e., p, where packets are classified based on the quadruple Quad (see Eq. (9)). If p is classified as benign, it proceeds to "Normal packet transmission".
(2) When the model flags p as potentially malicious, p's Source IP (Src IP) is extracted and stored in a temporary "Suspicious IPs Buffer".
(3) Entropy analyses [9] are triggered only when the size of this buffer exceeds a predefined threshold thB (in this study, thB is set to 20), even there are duplicated IPs in the buffer.
(4) The 2D5NS calculates the traffic entropy of each suspicious source.
(5) If an entropy value of a suspicious IP is below a threshold thE (in this study, thE is set to 0.8) as a low diversity, the 2D5NS blocks the Src IP. Otherwise, the system recovers the QoS flow to normal packet transmission.

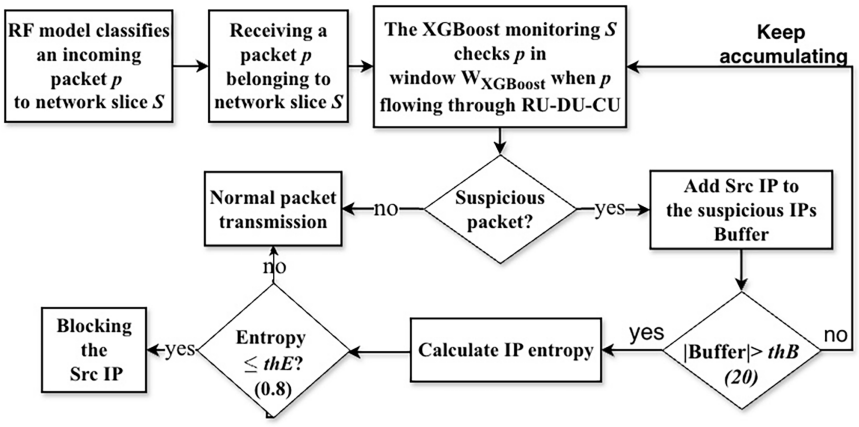

Figure 4: Anomaly Detection procedure of the 2D5NS (e.g., thE is set to 0.8 and thB is set to 20).

In Ref. [9], $W_{entropy}$ is defined as an entropy window employed for attack signature analysis. Its size is fixed at q packets sent by an Src IP along its connection toward a target IP, aiming to detect malicious behaviors of this IP. In this study, when |Suspicious IPs Buffer| $\geq thB$ (Step (3)), the consecutive q packets of $W_{entropy}$(Src IP) belonging to a suspicious Src IP in the buffer are retrieved. In Step (4), the entropy of this

Src-IP is then calculated by using Eq. (2) with $H(\text{Src IP}) = thE(\text{Src IP})$, where x_i is the IAT of two adjacent packets of the q packets, $n = m$ the number of IAT groups in W_{entropy}, $p(x_i)$ the distribution of x_i, $m \leq q$. Here, $thE = 0.8$ and $thB = 20$ are our choice. Note that based on our algorithms and datasets, this combination of thE and thB as an elbow point of our experiments has achieved the best identification accuracy.

At CU side of Fig. 1, the function of blocking a Src IP or reducing its bandwidth is achieved via RIC policies. A program Q was implemented for CU. Q follows the RIC policies updated by IDS to analyze entropy of a suspected QoS flow [9], reduce the bandwidth and block the Src IP when necessary. At the UPF side, the function is triggered by SDN controller and UPF by using Flow table and Meter table.

3.4 Queueing Analysis for RF Classification–M/M/1/K Model

Here, the RF model as an architecture of one server with one queue, denoted by *Queue* (RF), was implemented. Let K be the maximum length of this queue. Then, this is a M/M/1/K queueing model. Assume that its arrival rate is λ and departure rate is μ which is the service rate of RF model. Let $\rho = \lambda/\mu$. According to [31], from the boundary condition, $\sum_{n=0}^{K} P_n = 1$ where P_n is the probability of n packets waiting in *Queue* (RF), $n \leq K$,

$$\sum_{n=0}^{K} P_n = \sum_{n=0}^{K} \rho^n P_0 = 1 \tag{10}$$

and

$$P_0 = \frac{1}{\sum_{n=0}^{K} \rho^n}. \tag{11}$$

(1) When $\rho = 1$, i.e., $\lambda = \mu$, since $P_0 = P_n = \frac{1}{K+1}$ [31], the expected length of *Queue* (RF), denoted by L is

$$L = \sum_{n=0}^{K} nP_n = \frac{\sum_{n=0}^{K} n}{K+1} = \frac{\frac{K(K+1)}{2}}{K+1} = \frac{K}{2}. \tag{12}$$

This means the average length is half full.

(2) If $\rho \neq 1$,

$$L = \sum_{n=0}^{K} nP_n = P_0\rho \sum_{n=0}^{K} n\rho^{n-1} = P_0\rho \frac{1-(K+1)\rho^K + K\rho^{K+1}}{(1-\rho)^2} \tag{13}$$

As shown, $L = \text{f}(P_0, \rho, K)$. Number of packets in *Queue* (RF) in steady state, denoted by L_q, is

$$L_q = L - (1 - P_0) = L - \frac{\rho(1-\rho^K)}{1-\rho^{K+1}} \tag{14}$$

Since based on [31], as $\rho \neq 1$, $P_0 = \frac{1-\rho}{1-\rho^{K+1}}$, meaning $P_0 = \text{f}(\rho, K)$.

Let W be the average waiting time of a packet in *Queue* (RF), according to Little's formula [32], $W = L/\lambda'$ where $\lambda' = \mu\left(L - L_q\right)$, i.e., the mean rate of packets actually entering the *Queue* (RF), also known as the effective arrival rate. The average waiting time a packet waits in this queue, denoted by W_q (excluding RF model's service time $1/\mu$),

$$W_q = W - 1/\mu = L_q/\lambda' \tag{15}$$

As shown, $W_q = \mathrm{f}(P_0, \rho, K, \mu)$.

In fact, the queueing models of eMBB and mMTC slices are also the same as that of RF classification model, i.e., M/M/1/K. Thus, they will not be redundantly specified here.

3.5 Queueing Analyses on the Detection System–M/M/c/K Model

Currently, most network packets belong to eMBB slice. AR/VR/IP TV/multi-media programs are examples. To improve the performance of the detection of malicious eMBB packets, a cluster system with C IDSs was employed to serve this slice (see Fig. 3). Let $IDS(\text{eMBB}) = \{IDS_1, IDS_2, \ldots, IDS_C\}$. When the master node M_a, i.e., the RF model, retrieves/receives an eMBB packet p from $Queue(\text{RF})$, it delivers p to the queue of eMBB, denoted by $Queue(\text{eMBB})$ served by $IDS(\text{eMBB})$. If an IDS is idle, e.g., IDS_n, it retrieves a packet from $Queue(\text{eMBB})$ for intrusion detection, $1 \le n \le c$.

In other words, $IDS(\text{eMBB})$ cluster system forms a M/M/c/K model, where K is the buffer size of $Queue(\text{eMBB})$. Let λ_n be the arrival rate of $Queue(\text{eMBB})$. According to [31],

$$\lambda_n = \begin{cases} \lambda & 0 \le n < K \\ 0 & n \ge K \end{cases} \tag{16}$$

That is, when buffer is not full, the arrival rate is λ. Otherwise, arrival rate is 0, accepting no packets.

Let μ_n be the departure rate of the cluster system.

$$\mu_n = \begin{cases} n\mu & 0 \le n < c \\ c\mu & c \le n \le K \end{cases}$$

where μ is the service/departure rate of an IDS under the assumption that all IDSs are identical. When n IDSs are detecting attacks, $0 \le n < c$, the system service rate μ_n is $n\mu$. When $n \ge c$, $\mu_n = c\mu$, since there are only c IDSs. Let $q = \lambda/\mu$. The probability of n packets in the buffer, denoted by P_n is

$$P_n = \begin{cases} \frac{1}{n!} q^n P_0 & 0 \le n < c \\ \frac{1}{c^{n-c} c!} q^n P_0 & c \le n \le K \end{cases}$$

When $n \ge c$, only c IDSs are detecting attacks. Since $\sum_{n=0}^{K} P_n = 1$, thus

$$P_0 = \left[\sum_{n=0}^{c-1} \frac{1}{n!} q^n + \sum_{n=c}^{K} \frac{1}{c^{n-c} C!} q^n \right]^{-1} \tag{17}$$

Ref. [31] mentioned that the expected queue length L_q,

$$L_q = \sum_{n=c}^{K} (n-c) P_n = L - \sum_{n=0}^{c-1} (n-c) P_n - c \tag{18}$$

Thus,

$$L = L_q + c + \sum_{n=0}^{c-1} (n-c) P_n = L_q + c - \sum_{n=0}^{c-1} (c-n) P_n \tag{19}$$

the expected waiting time in $Queue(\text{eMBB})$, denoted by W,

$$W = \frac{L}{\lambda'},\ \lambda' = \lambda\left(1 - P_K\right) \tag{20}$$

Let W_q be the expected waiting time W excluding an IDS's service time $1/\mu$.

$$W_q = W - 1/\mu = L_q/\lambda' \tag{21}$$

which is the same as that of Eq. (15).

4 Experimental Results and Validation

In the following, three experiments are performed and the parameters involved are listed in Tables 3 and 4. The former (The latter) trains and validates our RF model (XGBoost model), while the third evaluates the 2D5NS (RF Model and XGBoost IDS model).

Table 3: Parameters of RF classification model.

Category	Parameter Name	Parameter Value	Description
Data Preprocessing	Original Data Set	Dataset-Unicauca-Version2-87Atts.csv	The raw network traffic dataset used in our experiments
	Training Samples	500,000	The number of samples resampled for the training set
	Imbalance Handling	SMOTE	A hybrid sampling method combining SMOTE and Tomek Links
ML Model	Model Type	Bagged Decision Trees (Random Forest)	A classifier trained using fitcensemble
	Number of Trees	100	The number of estimators in the forest
	Tree Parameters	Max Depth: 20, Min Leaf: 20	Constraints to prevent overfitting
	Validation Method	Train-Test Split (70/30) & 3-Fold CV	Evaluating performance on unseen data
Target Variables	Target Slice (Class)	eMBB, uRLLC, mMTC, Others	The 4 target network slices

Table 4: Parameters of XGBoost IDS model.

Category	Parameter Name	Parameter Value	Description
Data Preprocessing	Original Data Set	CSE-CIC-IDS2018	Uses Parquet files located in the csecicids2018 directory.
	Sample Size (Total)	300,000	240,000 samples for the training pool and 60,000 samples for simulation holdout.
	Imbalance Handling	Stratified Split	Uses the stratify parameter in train_test_split for stratified sampling; SMOTE is not in use.
	Model Type	XGBoost Classifier	The code imports and employs import xgboost as xgb for model training.
ML Model	Validation Method	3-Fold CV & Train-Test Split (80/20)	Validated using 3-Fold Learning Curves; data is split into 80% for training and 20% for holdout/testing.
Target Variables	Target Slice (Class)	eMBB, mMTC, uRLLC	Attack traffic is mapped into these 3 specific 5G network slices.

4.1 Experimental Parameters

In multi-class classification tasks, two aggregation methods are utilized to summarize the performance:

(1) Macro Average [33]: It is the unweighted mean of a metric calculated independently for each class. As shown in Eq. (22),

$$\text{Macro Average} = \frac{1}{N}\sum_{i=1}^{N} Met_i \tag{22}$$

where Met_i is the metric score of network slice i and N is the number of concerned metrics.

(2) Weighted Average [33]: It is defined in Eq. (23).

$$\text{Weighted Average} = \frac{\sum_{i=1}^{N}(Met_i \times Support_i)}{TotalSamples} \tag{23}$$

where $Support_i$ refers to the number of actual occurrences (true instances) of class i in the dataset. This metric is particularly useful when dealing with imbalanced datasets.

Receiver Operating Characteristic (ROC) and Precision-Recall (PR) [29] curves are also involved to evaluate a model's generalization capability and verify the absence of overfitting. The ROC illustrates the trade-off between the True Positive Rate and the False Positive Rate with the Area Under the Curve (AUC) serving as a key metric for overall classification capability.

Experiments were performed using a computer with 11th Gen Intel Core i7-11700 2.5 GHz CPU, 16 logical and 8 physical cores with 16 GB of main memory and the operating system was Windows 10. Each experiment is performed 10 times to calculate their averages and standard deviations.

4.2 Experiment 1: The RF Classification Model

In the first experiment, the IP Network Traffic Flows Labeled with 75 Apps data set [34], i.e., Dataset-Unicauca-Version2-87Atts.csv, was employed. The data on the Kaggle platform with the Stratified Sampling approach was pre-proposed. A total of 500,000 ($=\sum_{s=1}^{m}|X_s|$, see Eq. (5)) samples were selected to train and validate our RF model M where m is the number of concerned network slices. The 100 decision trees are trained with the procedure shown in Algorithm 1.

4.2.1 Data Preprocessing

To ensure computational efficiency, numerical data types were downcast, e.g., converting float64 to float32 or int64 to int8/16/32, resulting in a reduction in memory usage by 56.5%. To enhance the discriminatory power of M, advanced feature engineering was conducted, including:

(1) Ratio-based features: Computing FBPR (Eq. (6)) and FHPR (Eq. (8)) for capturing the behavioral characteristics of the data exchange.

(2) Statistical features: Deriving Avg.Fwd.Packet.Size and Avg.Packet.Size.Total to further distinguish high-bandwidth eMBB traffic from uRLLC small-packet transmissions, besides using DSCP values.

(3) Logarithmic transformation: Applying np.log1p transformation (defined as $\ln(1+x)$, where x is the raw feature value) to highly limit the values of those features with larger values, such as Flow.Duration and Total.Length, helping compress and normalize their distributions.

(4) Rare classes: Classes with fewer than r instances were removed where $r = 2$.

4.2.2 RF Model Training

After labeling NSSAI_Type, the dataset was partitioned into training set S_{train} and testing set S_{test} with a 70:30 ratio. Following model training, a PR curve (show as Fig. A1 where Ai means the i^{th} figure in the Appendix A of this paper) was plotted specifically for the uRLLC. The curve illustrates that increasing the threshold significantly improves precision while moderately reducing recall. The hyperparameters are obtained by invoking

rand_search_rf = RandomizedSearchCV(estimator = rf_base, param_distributions = param_dist, n_iter = 10, cv = 3) where a distribution of hyperparameters to search over

param_dist = {

'n_estimators': randint(80, 150),

'max_depth': randint(10, 20),

'min_samples_leaf': randint(5, 25),

'max_features': ['sqrt']}

After that grid_search_rf.fit(X_train, y_train) is then called to train the RF model.

4.2.3 RF Model Evaluation

Algorithm 2 details the model evaluation procedure implemented in Python using the scikit-learn library. This study employs a 3-fold cross-validation (3-fold CV) strategy on the training set. The results were listed in Table 5. The average accuracy was 81.33%.

Algorithm 2: Model evaluation on RF classification model M

Input: Resampled Training Data (S_{train}), Independent Test Data (S_{test}), Model Hyperparameters.
Output: CV Accuracy, Confusion Matrix, Classification Results.

Step 1: Model Initialization

1. Initializing RF classification model M with optimized hyperparameters.

Step 2: 3-Fold Cross-Validation

2. Partitioning S_{train} into (k = 3) folds.
3. Training and validating iteratively to obtain average accuracy (Table 5).

Step 3: Final Inference on Test Set

4. Training RF model M on full S_{train}.
5. Predicting labels on S_{test}: $\hat{y} = M(X_{test})$.
6. Generating Confusion Matrix (Table 6).
7. Computing Accuracy, Precision, Recall, F1-Score (Table 7), Marco Avg, Weighted Avg (Table 8) and AUC scores from ROC Curves (see Fig. A2).

Table 5: 3-Fold cross validation on M's training stage by using the training data extracted form [34] (%).

	Fold1	Fold2	Fold3	Average
Accuracy	81.25	81.33	81.41	81.33

After that, the RF model M was evaluated on S_{test}, which comprises approximately 1.07 million samples. Table 6 shows the Confusion Matrix and Table 7 lists the Classification results. Table 8 shows the overall weighted Average accuracy [33] of 82.98%, not far away from 81.33% listed in Table 5.

Table 6: Network slices' confusion matrix in its training stage.

	Predicted			
True	**eMBB**	**mMTC**	**uRLLC**	**Others**
eMBB	598,767	620	32,587	99,814
mMTC	56	573	6	4
uRLLC	2311	24	6428	2011
Others	56,290	101	9421	264,175

Table 7: Model's classification/evaluation results in Training Stage (X/Y represents μ/σ).

	Metrics				
Slice	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1 Score (%)**	**Support**
eMBB	82.14/4.33	91.08/3.67	81.82/2.87	86.20	731,788
mMTC	99.92/5.21	43.47/6.76	89.67/3.84	58.56	639
uRLLC	95.68/5.45	13.27/7.01	59.66/6.62	21.71	10,774
Others	84.38/3.98	72.18/2.99	80.06/5.46	75.91	329,987

Table 8: M's macro/weighted average results by using the unseen data extracted from [34] in M's training stage.

	Metrics				
Average	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1 Score (%)**	**Support**
Macro (Eq. (22))	90.53	55.00	77.80	60.60	1,073,188
Weighted (Eq. (23))	82.98	84.46	81.06	82.37	1,073,188

Further, Tables 6–8 also indicate several achievements.

(1) The results highlight the challenges in detecting minority network slices (uRLLC and mMTC) within imbalanced 5G traffic.
   (1) The model exhibited strong performance for the dominant eMBB slice, achieving a Precision of 91.08% and a Recall of 81.82% (see Table 7).
   (2) The application of the SMOTETomek resampling strategy successfully boosted the Recall for minority classes, i.e., the mMTC (uRLLC) slice achieved a high Recall of 89.67(59.66)% (the 4th column of Table 7).
   (3) Table 6 reveals that 32,587 eMBB samples and 9421 Others samples were misclassified as uRLLC, resulting in a lower Precision 13.27% ($= \frac{6428}{32{,}587+6+6428+9421} \times 100$) for the uRLLC class. This suggests that while the model achieves a Recall of 59.66% ($= \frac{6428}{2311+24+6428+2011} \times 100$) for uRLLC, it generates false positives due to the aggressive oversampling of minority classes.

(2) The area under an AUC score (Fig. A2) shows the model's performance across all possible probability thresholds using a One-vs.-Rest (OvR). All network slices exceed 0.90 (eMBB: 0.90, mMTC: 0.99, uRLLC: 0.91, Others: 0.91). The PR Curve (Fig. A3) further confirms that eMBB's average precision (AP = 0.95) is easily distinguished, and uRLLC (AP = 0.31) exhibits lower performance. uRLLC suffers from high False Positives. 32,587 eMBB instances are misclassified as uRLLC (see Table 6).

4.3 Experiment 2: XGBoost-Based IDS Model

Experiment 2 trains and evaluates the XGBoost-based IDS system by adopting the dataset derived from the CSE-CIC-IDS2018 [35,36], different from the one for training and validating RF model. This dataset is comprehensive benchmark that includes diverse DDoS attack scenarios. Algorithm 3 lists the procedure. The hyperparameters are generated like that of Experiments 1. But 'max_features': ['auto',: 16] is increased since it would be better to reduce the number of features, particularly by invoking PCC.

Algorithm 3: Slice-specific XGBoost-based IDS model training and validation

Input:

Raw network traffic dataset D: CSE-CIC-IDS2018 (CSV files).
Slice Logic for eMBB ($PktLen > 1000$B), mMTC ($Label = 'Bot'$), uRLLC ($0 < IAT < 1000$ µs).
Vector of 16 features (e.g., flow_bytes_per_s, flow_iat_min, etc.).
Hyperparameters θ:n_estimators, max_depth, learning_rate, subsamples, etc.

Output:

Trained XGBoost IDS Model M_{XGBoost}: JSON file.
Normalization Parameters: Mean (μ) and Standard Deviation (σ) of the training set.
Metrics: Accuracy, Precision, Recall and F1-Score.

4.3.1 Data Preprocessing

The dataset is first pre-processed.

(1) **Data set classification**

To individually train the three XGboost-Based IDS models (see Figs. A4–A6), the 2D5NS categorizes samples in the dataset into three sets.

(1) eMBB Slice: The 2D5NS filtered QoS flows with a fwd_packet_length_mean greater than 1000 bytes to simulate data-intensive applications consuming high bandwidth, such as 4K video streaming.
(2) mMTC Slice: The 2D5NS mapped flows labeled as 'Bot' to this slice, reflecting the high density of compromised IoT devices often found in botnets.
(3) uRLLC Slice: Those flows with flow_iat_mean (Inter-Arrival Time, IAT) between 0 and 1000 µs are selected, representing time-sensitive signals.

Remaining flows belong to general background traffic, i.e., Others.

This process comprises distinct attack classes, including DoS, DDoS, Botnet, Brute Force variants and the Benign classes.

(2) **Feature selection and slice-specific classification**

After PCC analysis, only correlation coefficients between 0.9 and 1 or between −0.9 and −1 are retained, remaining 16 features (see Table 9), which are classified into three types.

(1) Volume-Based Features for eMBB Defense

DDoS targeting eMBB, such as UDP floods, are considered attacks that saturate bandwidth. Typical features, include bwd_packets_length_total, quantifying the total byte count of payloads received in the backward direction, and flow_bytes_per_s, measuring the average throughputs of a QoS flow.

(2) Behavioral State Features for mMTC Defense

mMTC devices are characterized by periodic transmission patterns driven by energy-saving sleep and wake cycles, interchangeably. Botnet attacks often disrupt this periodicity by forcing compromised devices to remain active for malicious communication. Three typical features are selected, including flow_packets_per_s, measuring packets transmitted per second, active_max, recording the maximum duration a flow remained active before going idle, and idle_min, identifying the minimum period a flow stayed inactive.

(3) Timing and Latency Features for uRLLC Defense

Attacking uRLLC slices often introduces jitter or delay. Inter-Arrival Time (IAT) statistics are sensitive to these timing irregularities. For instance, a sudden drop in flow_iat_min, indicating a packet flood disrupting the expected regular intervals of control signals; flow_iat_max and flow_iat_min, representing the maximum and minimum time intervals between consecutive packets in a flow, respectively; other features include fwd_iat_min, flow_duration, etc.

Table 9: Selected features for XGBoost-based IDS model slice DDoS detection.

Flow	Backward	Forward and Others
1. flow_duration	6. bwd_packets_length_total	12. fwd_iat_min
2. flow_iat_min	7. bwd_iat_total	13. fwd_iat_total
3. flow_iat_max	8. bwd_iat_min	14. fwd_iat_max
4. flow_bytes_per_s	9. bwd_iat_max	15. fwd_header_length
5. flow_packets_per_s	10. bwd_packet_length_max	16. packet_length_max
	11. init_bwd_win_bytes	

In summary, comparing the 3-fold results of Experiment 1 (Table 5) and Experiment 2 (Table 10), the approach adopted by Experiment 2 is more effective.

Table 10: 3-Fold CV Results for eMBB, mMTC and uRLLC (%).

	Fold1	Fold2	Fold3	Avg Accuracy
eMBB	99.98	99.99	99.98	99.98
mMTC	99.97	99.98	99.99	99.98
uRLLC	93.39	93.21	93.37	93.32

4.3.2 XGBoost_Based IDS Model Training and Validation

The hyperparameter optimization for the XGBoost model was performed on the Kaggle platform using RandomizedSearchCV. The Feature Importance Analysis results are listed in Table A1 in the Appendix A of this paper. Features, such as 'flow_bytes_per_s' for detecting eMBB (ranked no. 5 in column 2 of Table A1) and 'flow_iat_min' for detecting uRLLC (ranked no. 3 in column 4 of Table A1), respectively quantify the throughput surges for eMBB saturation and the timing irregularities associated with uRLLC latency violations.

Table 10 lists the accuracies of eMBB, mMTC and uRLLC on 3-fold CV. The eMBB and mMTC exceed 99.98%, while the uRLLC's is 93.32%. The reasons why the eMBB and mMTC are not 100% accurate probably because some of their slide-defining features (see Table A1) are the same, like flow_iat_min (both ranked no. 3) and bwd_iat_min (both ranked no. 7), even with different weights. Also, some of their individual features are correlated, like eMBB's bwd_packet_length_max and mMTC's packet_length_max (both ranked no. 8).

(1) Performance of eMBB and mMTC Slices

Tables 11 and 12 present the eMBB's and mMTC's confusion matrixes on D_{test}, respectively. Table 13 (Table 14) lists the performance of eMBB (mMTC) where N in Eqs. (22) and (23) is 2 which are Normal and attack. eMBB (mMTC) achieves an overall Accuracy and F1-Score higher than 0.9982 (0.996). uRLLC's confusion matrix will be discussed later.

Table 11: eMBB confusion matrix.

	Predicted	
Actual	**Normal**	**Attack**
Normal	12,907	21
Attack	12	6770

Table 12: mMTC confusion matrix.

	Predicted	
Actual	**Normal**	**Attack**
Normal	14,835	23
Attack	6	1497

Table 13: eMBB classification performance (X/Y represents μ/σ).

Class	Accuracy	Precision	Recall	F1 Score	Support
Normal	0.9984/0.031	0.9984/0.042	0.9984/0.051	0.9984	12,928
Attack	0.9982/0.045	0.9982/0.038	0.9982/0.073	0.9982	6782
Macro Avg (Eq. (22))	0.9983/0.069	0.9983/0.062	0.9983/0.057	0.9983	19,710
Weighted Avg (Eq. (23))	0.9983/0.055	0.9983/0.062	0.9983/0.055	0.9983	19,710

Table 14: mMTC classification performance (X/Y represents μ/σ).

Class	Accuracy	Precision	Recall	F1 Score	Support
Normal	0.9985/0.044	0.9985/0.053	0.9985/0.047	0.9985	14,858
Attack	0.996/0.049	0.996/0.051	0.996/0.046	0.996	1503
Macro Avg (Eq. (22))	0.9972/0.059	0.9972/0.047	0.9972/0.052	0.9972	16,361
Weighted Avg (Eq. (23))	0.9982/0.047	0.9982/0.051	0.9982/0.043	0.9982	16,361

(2) Performance of uRLLC Slice

The evaluation of the uRLLC presents a different scenario due to its strict latency constraints. Table 15 shows its confusion matrix.

(1) The accuracy achieves 93.59% (= $\frac{8693+2472}{8693+2472+743+22} \times 100$).
(2) The Precision is 99.12% (= $\frac{2472}{2472+22} \times 100$).
(3) The Recall is 76.88% (= $\frac{2472}{2472+743} \times 100$).
(4) The ROC Curve presented in Fig. A6 demonstrates an AUC of 93.19% when FP rate is 0.25% (= $\frac{22}{8693+22} \times 100$).

Table 15: uRLLC confusion matrix.

	Predicted	
Actual	**Normal**	**Attack**
Normal	8693	22
Attack	743	2472

The uRLLC classification performance is listed in Table 16.

Table 16: uRLLC classification performance (X/Y represents μ/σ).

Class	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	Support
Normal	99.75/4.58	92.13/5.78	99.75/3.87	95.79	8715
Attack	76.88/5.44	99.12/6.01	76.89/4.63	86.60	3215
Macro Avg	88.32/4.87	95.62/5.55	88.32/4.33	91.20	11,930
Weighted Avg	93.59/5.88	94.01/4.98	93.59/5.47	93.31	11,930

4.4 Experiment 3: 2D5NS Evaluation

In the following, the 2D5NS and other ML schemes, including Random Forest (RF), Decision Tree (DT) and Support Vector Machine (SVM), will be evaluated on test data. The reasons why they were chosen are that DT is a tree. The results it generates are obvious sometimes with bias and not adjustable. RF as an ensemble system consists of m DTs, $m > 1$. Its classification results follow majority voting of the m DTs', often more accurate and trustable. In fact, the XGBoost is also an ensemble scheme During establishment, the system tries to solve the drawbacks and biases of previously generated trees to robust its prediction results. SMV is an excellent classification approach by maximizing the margin of support vectors. Also, to fairly compare their performance, all schemes are trained (tested) by using the same training set (test set) on the same computer environment. The dataset is also CSE-CIC-IDS2018.

4.4.1 Evaluation of Classification Models

The results are listed in Table 17, indicating that ensemble learning approaches XGBoost achieved the highest accuracy of 86.07%, and RF accuracy is 83.59% on all network slices, surpassing the single models DT (81.65%) and SVM (75.97%) (see column 3). Readers may say that in Tables 5, 8 and 17, the accuracies when training the RF model are different, i.e., 81.83%, 82.98% and 83.59%, respectively. In fact, the difference is conducted due to employing different datasets.

Also, the RF model achieved the highest Recall of 89.67% in the mMTC slice. In contrast, the SVM showed a very low F1-score of 0.94% in the uRLLC slice, indicating tree-based ensemble models captured the features of uRLLC traffic more effectively than the hyperplane-based SVM method.

In summary, the XGBoost attained the highest overall accuracy, i.e., 86.07% and F1-Score, i.e., 90.23%.

4.4.2 IDS Evaluation

This section evaluates our XGBoost-based 2D5NS and other ML/AI detection methods, including ATD (Anomaly Traffic Detector) [17], Chang`s scheme (hereafter Chang, an CNN-LSTM based) [18], CBA (Precision AI, ID-CNN and BiLSTM based) [19], Hasan`s approach (hereafter Hasan) [37] and Alnatsheh`s scheme (hereafter Alnatsheh) [38]. Table 18 presents their practical performance in real-time training and attack detection.

Table 17: Performance of different schemes for training by using the data extracted from [35] (X/Y represents μ/σ).

Slice	Model	Accuracy (%)	Precision (%)	Recall (%)	F1_Score (%)
eMBB	XGBoost	86.07/6.98	88.47/5.87	92.06/6.44	90.23
	RF	83.59/5.88	89.63/10.3	86.20/8.55	87.88
	DT	81.65/6.92	87.07/8.73	87.01/8.99	87.04
	SVM	75.97/7.88	80.91/7.02	85.44/10.1	83.11
mMTC	XGBoost	86.07/7.64	56.28/5.49	84.82/8.67	67.67
	RF	83.59/5.33	43.47/3.86	89.67/10.45	58.56
	DT	81.65/6.77	45.91/3.98	65.88/6.67	54.11
	SVM	75.97/10.33	53.59/8.7	82.94/9.08	65.11
uRLLC	XGBoost	86.07/8.05	43.38/7.03	41.47/4.47	42.40
	RF	83.59/7.41	72.58/6.68	13.68/3.77	23.02
	DT	81.65/7.2	19.25/5.43	39.72/4.01	25.93
	SVM	75.97/9.1	5.13/1.01	0.52/0.1	0.94
others	XGBoost	86.07/7.01	81.57/9.65	74.26/7.78	77.74
	RF	83.59/6.88	72.18/8.63	80.06/9.89	75.91
	DT	81.65/5.89	73.66/6.85	71.15/8.73	72.38
	SVM	75.97/8.99	63.51/5.09	57.43/6.98	60.32

The 2D5NS achieves superior results on the eMBB and mMTC slices with accuracy and F1-scores all ranging from 99.89% to 100.00%, effectively matching or slightly surpassing the performance of ATD and Alnatsheh's schemes.

The uRLLC slice shows a more challenging detection result. The 2D5NS attains the highest Accuracy (93.43%), Precision (93.89%) and F1-Score (93.13%). Notably, it outperforms not only statistical methods like Alnatsheh's (90.73%, 91.75% and 90.05%, respectively), but also machine learning approaches, including Chang's model (92.05%, 92.15% and 91.76%, respectively) and the CBA model (91.41%, 91.37% and 91.16%, respectively).

Furthermore, computational efficiency is a key factor of performance. The training time for the 2D5NS is extremely low, ranging from 0.29 to 0.59 s in average. In contrast, CBA (Chang) requires up to the range between 15.4 (7.33) s for uRLLC and 23.07 (13.14) s for eMBB, meaning that the 2D5NS provides a better trade-off between high classification performance and computational efficiency.

To validate the real-time responsiveness of the proposed model, simulations are conducted using MATLAB to measure the "Time to First Detection" (the last column of Table 18) defined as the time duration from the time point when an attack starts to the time point when the attack is first discovered, indicating how fast the system can trigger mitigation mechanism right after the attack begins. The results are summarized in the last column of Table 18.

Table 18: Performance evaluation for training and time to first detection (X/Y represents μ/σ).

Slice	Model	Accuracy (%)	Precision (%)	Recall (%)	F1_Score (%)	Training Time (s)	Time to First Detection (ms)
eMBB	2D5NS	99.89	99.89	99.89	99.89	0.59/0.12	3.68/1.11
	ATD	99.98	99.98	99.98	99.98	5.92/1.01	4.25/1.32
	Hasan	99.95	99.95	99.95	99.95	1.32/0.33	1.2/0.4
	Alnatsheh	97.79	97.82	97.79	97.79	188.59/23.1	2.05/1.01
	Chang	98.06	98.02	98.00	98.00	13.14/3.21	22.5/5.53
	CBA	98.28	98.28	98.28	98.07	23.07/8.93	55.62/9.64
mMTC	2D5NS	100.0	100.00	100.00	100.00	0.30/0.09	3.55/1.12
	ATD	99.99/0	99.99	99.99	99.99	3.74/1.2	4.20/1.37
	Hasan	99.93	99.93	99.93	99.93	0.68/0.21	1.2/0.29
	Alnatsheh	99.75	99.75	99.75	99.75	12.95/5.33	2.00/0.67
	Chang	99.78	99.78	99.78	99.77	8.4/2.45	21.4/5.36
	CBA	99.82	99.82	99.82	99.82	18.56/5.04	55.26/6.77
uRLLC	2D5NS	93.43	93.89	93.43	93.13	0.29/0.09	3.46/1.1
	ATD	92.19	92.23	92.19	91.94	3.69/1.03	4.17/1.52
	Hasan	89.47	89.35	89.47	89.40	0.47/0.11	1.1/0.33
	Alnatsheh	90.73	91.75	90.73	90.05	372.38/89.3	2.05/0.47
	Chang	92.05	92.15	92.05	91.76	7.33/2.03	22.1/6.63
	CBA	91.41	91.37	91.41	91.16	15.4/3.05	55.4/6.87

(1) With small packet sizes and low bandwidth, i.e., mMTC and uRLLC, all evaluated models exhibited low latencies. Specifically, the 2D5NS maintained a rapid detection time between 3.46 ms (uRLLC) and 3.68 ms (eMBB). While this is slightly higher than the simplest statistical methods (e.g., Hasan's 1.1 ms on uRLLC).

(2) The eMBB slice, characterized by high throughput and bursty traffic, serves as a stress test for model stability. The 2D5NS demonstrated significant advantages, particularly when compared to deep learning approaches, like Alnatsheh, Chang, and CBA.

A. While providing robust detection, Chang's and the CBA frameworks spent 22.5 and 55.62 ms, respectively. The 2D5NS achieved 83% (= $\frac{22.5-3.68}{22.5}$) and 93% (= $\frac{55.62-3.68}{55.62}$) reduction.

B. The 2D5NS maintains a lightweight structure to process high-volume data streams efficiently without the computational bottlenecks observed in deep learning models, like Chang and CBA.

C. Although Hasan achieved the lowest latency (1.2 ms), it often compromises detection accuracy in low-and-slow DDoS attacks, like uRLLC. The 2D5NS latency of 3.68 ms remains well within the 1-s latency constraint defined for Near-RT RIC (xApps).

In summary, Tables 5 and 10 show that the RF model adopted by Experiment 2 generates higher accuracy than that of Experiment 1. Even the performance of RF model in Experiment 3 is lower than that of Experiment 2 (Please compare Table 10 with Table 17), when comparing Tables 13, 14 and 16 with Table 18, readers can see the detection accuracies of both experiments for 2D5NS/XGBoost are high. uRLLC achieves at least 93.59% and 93.43% and eMBB's and mMTC's are higher than 99%.

4.4.3 Queue Length and Waiting Time

Fig. 5 shows the queue length L of our RF model. When ρ is higher than 0.9 under the assumption that μ = 3000 packets/s, the length starts growing. L_q is not shown since in Eq. (14), i.e., $L_q = L - \frac{\rho(1-\rho^K)}{1-\rho^{K+1}}$, when

K is huge, $L_q \cong L - \rho$ and ρ is less than 1. The theorical length at $\rho = 1$ is $K/2$. When $\rho = 0.999$ and $K = 1000$, 3000, 5000 and 7000, the L in average is respectively about 418, 1265, 1515 and 1568 packets waiting in the RF buffer. They indicate that the usage of longer buffer is low. The only benefit is that when $\rho > 1$, i.e., sudden rush traffic, the buffer can hold more packets. But buffer will soon full. Table 19 lists the queue lengths W of RF buffer in which X/Y represents that X is its theorical value and Y is the measured length in average.

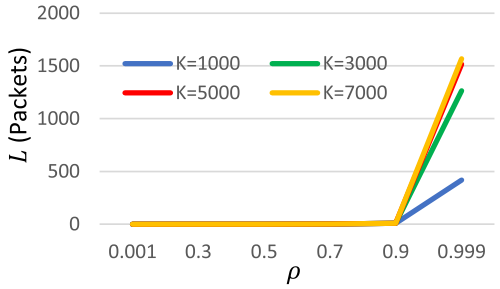

Figure 5: Queue length of RF Model ($M/M/1/K$).

Table 19: The queue length of RF buffer (X/Y: X is its theorical value and Y is the measured queue length).

ρ	$K = 1000$	$K = 3000$	$K = 5000$	$K = 7000$
0.001	0/0	0/0	0/0	0/0
0.3	0/0	0/0	0/0	0/0
0.5	1/0	1/0	1/0	1/0
0.7	2/0	2/0	2/0	2/0
0.9	9/11	9/8	9/5	9/6
0.999	417/412	1265/1258	1515/1501	1568/1551

Figs. 6 and 7 illustrate the theoretical waiting time W and W_q, respectively, on $K = 1000$ and $\mu = 3000$. As shown, the two figures are almost the same since the service time 1/3000 s is short. W and the measured W, denoted by W_m, on $K = 1000$ are listed in Table 20 in which W_m is longer than W since 2D5NS needs some overheads for processing the measured data.

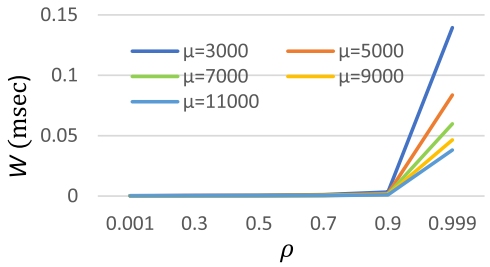

Figure 6: Waiting time including service time.

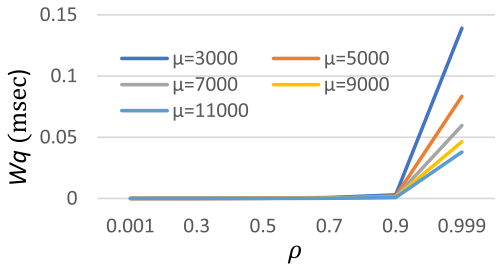

Figure 7: Waiting time excluding service time.

Table 20: The waiting time W (W/W_m: W_m is the measured value) on K = 1000.

ρ	μ = 3000	μ = 5000	μ = 7000	μ = 9000	μ = 11,000
0.001	0.334/0.421	0.2/0.41	0.143/2.67	0.111/0.236	0.091/0.167
0.3	0.476/0.545	0.286/0.356	0.204/0.39	0.159/0.233	0.13/0.19
0.5	0.667/0.72	0.4/0.566	0.285/0.39	0.222/0.421	0.182/0.25
0.7	1.111/1.32	0.667/0.822	0.476/0.499	0.37/0.49	0.303/0.443
0.9	3.333/3.68	2/3.2	1.111/1.89	1.101/1.355	0.91/0.88
0.999	139.5/145.3	83.7/88.64	59.78/63.19	46.5/50.33	38.04/43.45

For the IDS system that detects eMBB slices, the 2D5NS employed 3 (or 5) IDSs, i.e., c = 3 (or 5), to detect malicious behavior. The L, W and W_m are also individually about 1/3 (1/5) of those for $M/M/1/K$ model shown above. Here, they are not redundantly illustrated.

5 Conclusions and Future Studies

This research developed a security scheme, called the 2D5NS, to detect network slicing DDoS attacks mainly at the 5G O-RAN. Experiments demonstrated that this system can effectively and efficiently block malicious packets for DDoS.

Based on the experimental results and validation, the following conclusions can be conducted.

(1) Slice Traffic Classification: Deployed in the SMO with customized DSCP rules, the RF model performance is summarized in Tables 7 and 8. It achieved an overall accuracy (weighted AVG) of 82.98% (see Table 8), demonstrating robust performance for eMBB (91.08%, precision, see Table 7) and high sensitivity for mMTC (89.67%, recall).

(2) Detection Mechanism: The 2D5NS's performance detailed in Table 18 shows high precision in identifying attacking IPs, achieving near-perfect F1-Scores for mMTC (100.00%) and eMBB (99.89%) while leading in uRLLC (93.13%), meaning that real-time mitigation with latencies ranging from 3.46 ms in uRLLC to 3.68 ms in intensive eMBB scenarios. Coupled with rapid training, this offers a highly efficient solution for O-RAN security.

(3) Automated Mitigation Response: Once an attack is discovered, the system analyzes flows' entropies. Once confirmed, it blocks the Src IPs to mitigate the impact.

In other words, the system can accurately identify the Src IPs of abnormal traffic, service attributes (S-NSSAI), and priority level (5QI/QoS) based on a flow's Quadruple Quad (see Eq. (9)).

In the future, our research efforts will focus on two key areas to further enhance the RF-based classification robustness and adaptability. Although the proposed RF model achieves high accuracies for eMBB and mMTC slices, its performance in classifying uRLLC traffic remains suboptimal due to severe class imbalance. To address this, advanced data augmentation techniques, such as new versions of the SMOTE or GANs, will be implemented to generate high-fidelity synthetic uRLLC samples. Next, to optimize the automated mitigation response and its security, our future research intends to integrate the 2D5NS with Deep Reinforcement Learning (DRL) to dynamically adjust entropy thresholds based on real-time network states, aiming to ensure a more adaptive defense against sophisticated, low-rate DDoS attacks. These constitute our future studies.

Acknowledgement: Not applicable.

Funding Statement: This research was funded by National Science and Technology Council (NSTC), Taiwan, grant number [NSTC 113-2221-E-029-027].

Author Contributions: Kun-Lin Tsai proposed the scope of this research, guided the development of research ideals and improved the architecture of this paper. Shih-Ting Chiu developed the detailed algorithms and implemented these algorithms. Chihhsiong Shih analyzed and implemented the queueing models. Fang-Yie Leu proposed the basic ideas of the whole systems, controlled the research quality, validated the outcomes of experiments. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The CSE-CIC-IDS2018 dataset used in this study is publicly available from the University of New Brunswick, https://www.kaggle.com/datasets/solarmainframe/ids-intrusion-csv and The IP Network Traffic Flows Labeled with 75 Apps dataset employed in this study is publicly available at https://www.kaggle.com/datasets/jsrojas/ip-network-traffic-flows-labeled-with-87-apps.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

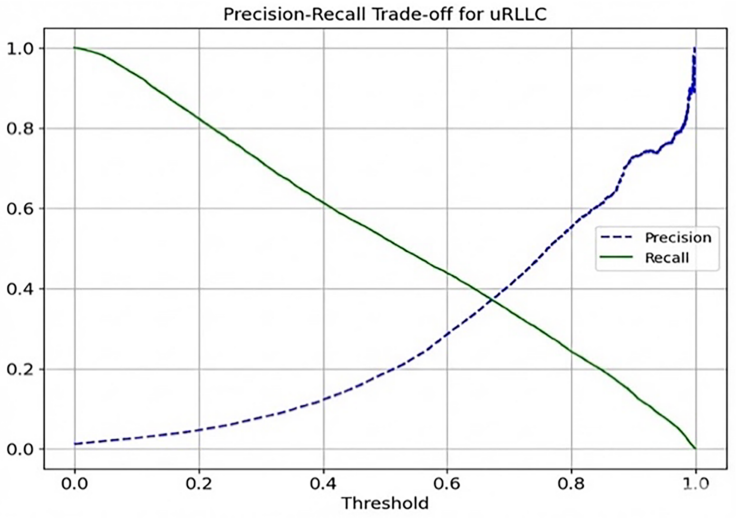

Figure A1: uRLLC's PR curve.

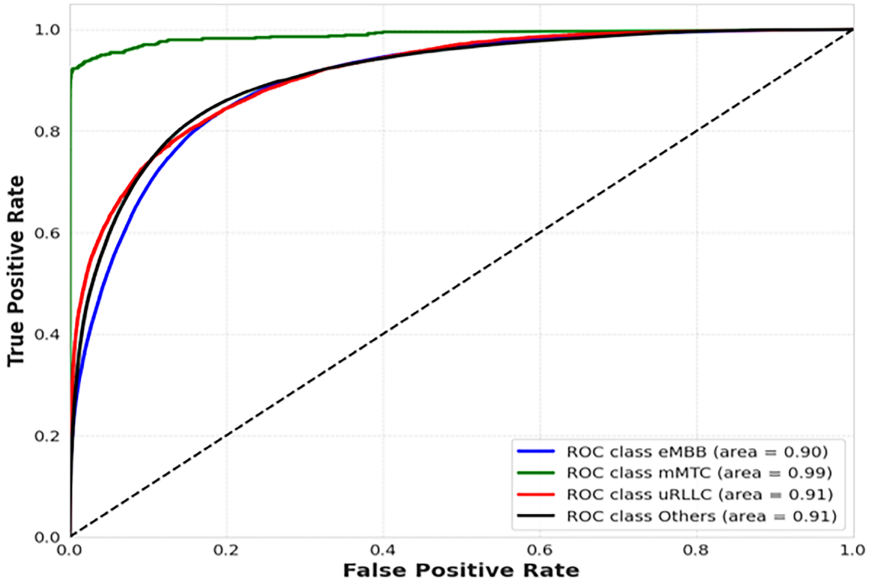

Figure A2: RF's ROC curve.

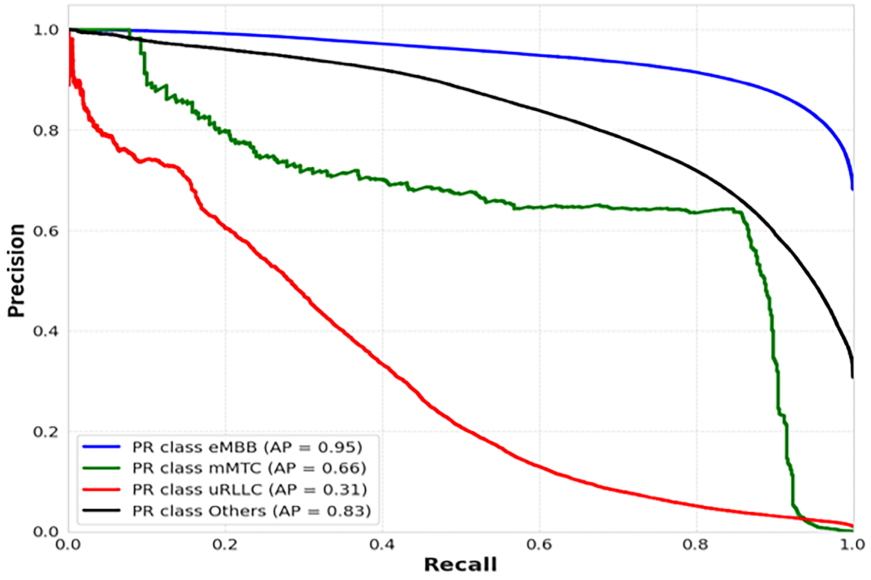

Figure A3: RF classification model's PR curve (AP: average-precision).

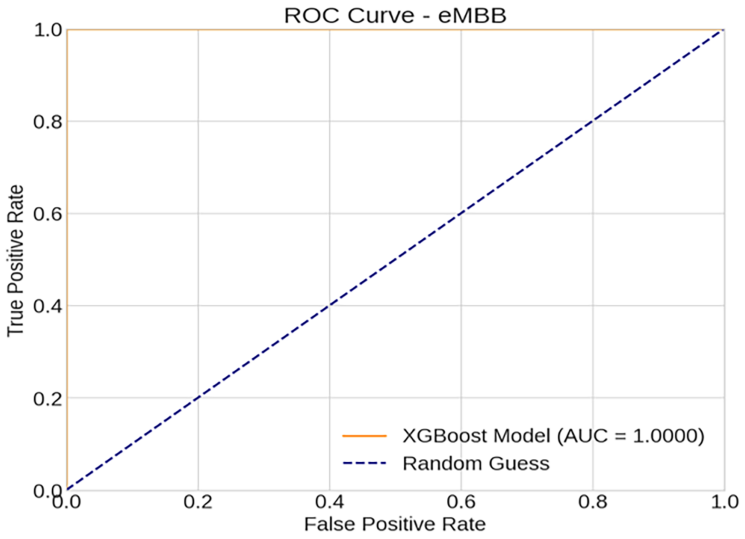

Figure A4: eMBB ROC curve.

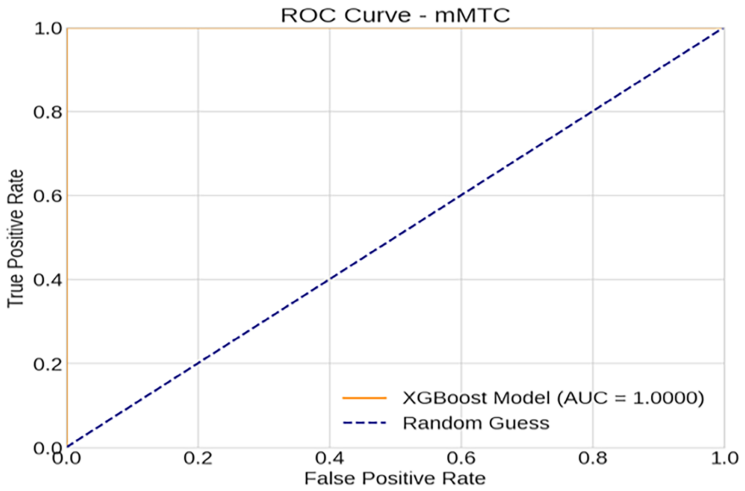

Figure A5: mMTC ROC curve.

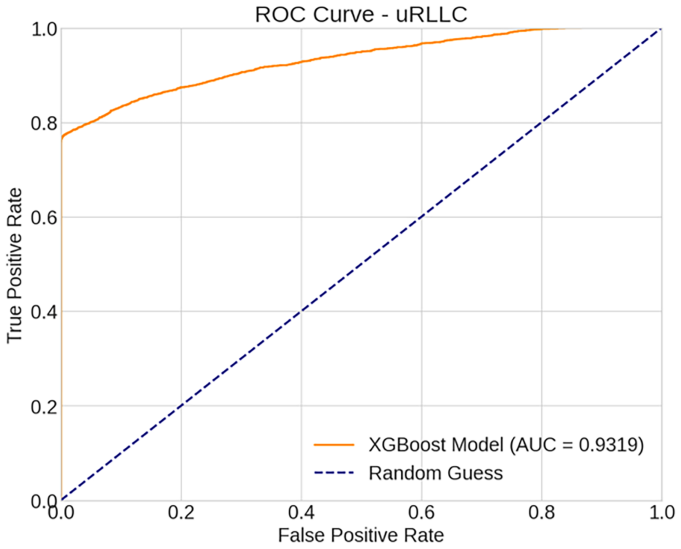

Figure A6: uRLLC ROC curve.

Table A1: The importance of 16 features on eMBB, mMTC and uRLLC.

Rank	Features of eMBB	Weight (eMBB)	Features of mMTC	Weight (mMTC)	Features of uRLLC	Weight (uRLLC)
1	fwd_packets_length_total	51	flow_packets_per_s	79	flow_bytes_per_s	375
2	fwd_header_length	39	init_fwd_win_bytes	75	init_fwd_win_bytes	347
3	flow_iat_min	35	flow_iat_min	72	flow_iat_min	309
4	fwd_packet_length_max	34	init_bwd_win_bytes	39	flow_packets_per_s	268
5	flow_bytes_per_s	31	total_fwd_packets	27	flow_duration	253
6	bwd_packets_length_total	28	fwd_iat_total	24	fwd_packets_length_total	200
7	bwd_iat_min	27	bwd_iat_min	21	bwd_packets_length_total	193
8	bwd_packet_length_max	26	packet_length_max	20	flow_iat_max	191
9	init_fwd_win_bytes	24	flow_duration	19	fwd_iat_min	190
10	bwd_iat_max	18	bwd_header_length	17	init_bwd_win_bytes	120
11	bwd_header_length	13	fwd_iat_min	17	fwd_packet_length_max	119
12	init_bwd_win_bytes	9	flow_iat_max	15	fwd_iat_total	92
13	flow_packets_per_s	9	fwd_packet_length_max	13	fwd_iat_max	76
14	total_backward_packets	9	fwd_iat_max	12	packet_length_max	60
15	fwd_iat_min	8	active_max	10	bwd_iat_min	52
16	fwd_iat_total	6	bwd_packets_length_total	10	bwd_packet_length_max	48

References

1. Samdanis K, Costa-Perez X, Sciancalepore V. From network sharing to multi-tenancy: the 5G network slice broker. IEEE Commun Mag. 2016;54(7):32–9. doi:10.1109/mcom.2016.7514161.
2. O-RAN Working Group 1. ETSI TS 103 982 V8.0.0. [cited 2025 Jan 1]. Available from: https://www.etsi.org/deliver/etsi_ts/103900_103999/103982/08.00.00_60/ts_103982v080000p.pdf.
3. Singh VP, Singh MP, Hegde S, Gupta M. Security in 5G network slices: concerns and opportunities. IEEE Access. 2024;12:52727–43. doi:10.1109/ACCESS.2024.3386632.
4. Lau F, Rubin SH, Smith MH, Trajkovic L. Distributed denial of service attacks. In: Proceedings of the SMC 2000 Conference Proceedings. 2000 IEEE International Conference on Systems, Man and Cybernetics. 'Cybernetics

Evolving to Systems, Humans, Organizations, and Their Complex Interactions' (Cat. No. 0); 2000 Oct 8–11; Nashville, TN, USA. p. 2275–80.
5. Amachaghi EN, Shojafar M, Foh CH, Moessner K. A survey for intrusion detection systems in open RAN. IEEE Access. 2024;12(3):88146–73. doi:10.1109/access.2024.3408690.
6. Thales Analysis Research. 2026 bad bot report—AI bots are changing the rules. [cited 2025 Jan 1]. Available from: https://cpl.thalesgroup.com/ppc/application-security/bad-bot-report.
7. ETRadware Ltd. Radware 2026 global threat report shows DDoS attacks jump 168% as cyber threats escalate across networks and applications. [cited 2025 Jan 1]. Available from: https://seekingalpha.com/pr/20405201-radware-2026-global-threat-report-shows-ddos-attacks-jump-168-percent-as-cyber-threats.
8. Polese M, Bonati L, D'Oro S, Basagni S, Melodia T. Understanding O-RAN: architecture, interfaces, algorithms, security, and research challenges. IEEE Commun Surv Tutor. 2023;25(2):1376–411. doi:10.1109/COMST.2023.3239220.
9. Chiu ST, Leu FY, Tsai KL, Susanto H. 3D5G-O: DDoS detection and defense system of 5G on O-RAN. In: Proceedings of the 9th International Conference on Mobile Internet Security; 2025 Dec 16–18; Sapporo, Japan. (In press).
10. Majeed A, Alnajim AM, Waseem A, Khaliq A, Naveed A, Habib S, et al. Deep learning-based symptomizing cyber threats using adaptive 5G shared slice security approaches. Future Internet. 2023;15(6):193. doi:10.3390/fi15060193.
11. De Alwis C, Porambage P, Dev K, Gadekallu TR, Liyanage M. A survey on network slicing security: attacks, challenges, solutions and research directions. IEEE Commun Surv Tutorials. 2024;26(1):534–70. doi:10.1109/comst.2023.3312349.
12. Allaw Z, Zein O, Ahmad AM. Cross-layer security for 5G/6G network slices: an SDN, NFV, and AI-based hybrid framework. Sensors. 2025;25(11):3335. doi:10.3390/s25113335.
13. Wu ZX, You YZ, Liu CC, Chou LD. Machine learning based 5G network slicing management and classification. In: Proceedings of the 2024 International Conference on Artificial Intelligence in Information and Communication (ICAIIC); 2024 Feb 19–22; Osaka, Japan. p. 371–5.
14. Soleymani SA, Eslamnejad M, Alimohammadi H, Akbas A, Foh CH, Shojafar M. DDoS detection and mitigation using d/xApp in O-RAN. In: Proceedings of the 2024 IEEE Future Networks World Forum (FNWF); 2024 Oct 15–17; Dubai, United Arab Emirates. p. 283–90.
15. El-Hajj M. Secure and trustworthy open radio access network (O-RAN) optimization: a zero-trust and federated learning framework for 6G networks. Fut Internet. 2025;17(6):233. doi:10.3390/fi17060233.
16. Abou El Houda Z, Moudoud H, Brik B. Federated deep reinforcement learning for efficient jamming attack mitigation in O-RAN. IEEE Trans Veh Technol. 2024;73(7):9334–43. doi:10.1109/tvt.2024.3359998.
17. Tsourdinis T, Makris N, Korakis T, Fdida S. AI-driven network intrusion detection and resource allocation in real-world O-RAN 5G networks. In: Proceedings of the 30th Annual International Conference on Mobile Computing and Networking; 2024 Nov 18–22; Washington, DC, USA. p. 1842–9.
18. Chang JE, Chiu YC, Ma YW, Li ZX, Shao CL. Packet continuity DDoS attack detection for open fronthaul in ORAN system. In: Proceedings of the NOMS 2024-2024 IEEE Network Operations and Management Symposium; 2024 May 6–10; Seoul, Republic of Korea. p. 1–5.
19. Baidar R, Maric S, Abbas R. Hybrid deep learning-federated learning powered intrusion detection system for IoT/5G advanced edge computing network. arXiv:2509.15555. 2025.
20. Zadeh FA, Civciss A, Ravihansa V, Sandeepa C, Liyanage M. Descriptor: 5G open radio access network multi-modal intrusion detection dataset (NetsLab-5GORAN-IDD). IEEE Data Descr. 2025;2(3):348–57. doi:10.1109/IEEEDATA.2025.3614167.
21. Liao D, Luo J, Vevstad J, Pappas N. RANGAN: GAN-empowered anomaly detection in 5G cloud RAN. arXiv:2508.20985. 2025.
22. Chen T, Guestrin C. XGBoost: a scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; 2016 Aug 13–17; San Francisco, CA, USA. p. 785–94.
23. McEliece RJ. The theory of information and coding. Cambridge, UK: Cambridge University Press; 2002.

24. Zhang S. An overview of network slicing for 5G. IEEE Wirel Commun. 2019;26(3):111–7. doi:10.1109/mwc.2019.1800234.
25. Ho TK. Random decision forests. In: Proceedings of 3rd International Conference on Document Analysis and Recognition; 1995 Aug 14–16; Montreal, QC, Canada. p. 278–82.
26. Breiman L, Friedman JH, Olshen RA, Stone CJ. Classification and regression trees. Belmont, CA, USA: Wadsworth; 1984.
27. Moore J, Abdalla AS, Khanal P, Marojevic V. Integrated LLM-based intrusion detection with secure slicing xApp for securing O-RAN-enabled wireless network deployments. In: Proceedings of the 2025 IEEE International Conference on Communications Workshops (ICC Workshops); 2025 Jun 8–12; Montreal, QC, Canada. p. 274–9.
28. System architecture for the 5G system (5GS). ETSI TS 123 501 V19.5.0 (2026-01). [cited 2025 Jan 1]. Available from: https://www.etsi.org/deliver/etsi_ts/123500_123599/123501/19.05.00_60/ts_123501v190500p.pdf.
29. Davis J, Goadrich M. The relationship between precision-recall and ROC curves. In: Proceedings of the 23rd International Conference on Machine Learning; 2006 Jun 25–29; Pittsburgh, PA, USA. p. 233–40.
30. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. SMOTE: synthetic minority over-sampling technique. arXiv:1106.1813. 2011.
31. Shortle JF, Thompson JM, Gross D, Harris CM. Fundamentals of queueing theory. Hoboken, NJ, USA: John Wiley & Sons, Inc.; 2018.
32. El-Taha M, Stidham S. Little's formula and extensions. In: Sample-path analysis of queueing systems. Boston, MA, USA: Springer; 1999. p. 159–212.
33. Lee MCH, Braet J, Springael J. Performance metrics for multilabel emotion classification: comparing micro, macro, and weighted F1-scores. Appl Sci. 2024;14(21):9863. doi:10.3390/app14219863.
34. Rojas JS. IP network traffic flows labeled with 75 apps. [cited 2025 Jan 1]. Available from: https://www.kaggle.com/datasets/jsrojas/ip-network-traffic-flows-labeled-with-87-apps.
35. Hewapathirana IU. A comparative study of two-stage intrusion detection using modern machine learning approaches on the CSE-CIC-IDS2018 dataset. Knowledge. 2025;5(1):6. doi:10.3390/knowledge5010006.
36. Kerosene Group. IDS 2018 intrusion CSVs (CSE-CIC-IDS2018). [cited 2025 Jan 1]. Available from: https://www.kaggle.com/datasets/solarmainframe/ids-intrusion-csv.
37. Hasan K, Hossain KS, Alam MS, Islam MZ, Apurbo GM, Ahmed MF, et al. Real-time DDoS detection in software-defined networks using machine learning. In: Proceedings of the International Conference on Computer and Information Technology (ICCIT); 2024 Dec 20–22; Cox's Bazar, Bangladesh. p. 453–8.
38. Alnatsheh A, Alsarhan A, Aljaidi M, Rafiq H, Mansour K, Samara G, et al. Machine learning-based approach for detecting DDoS attack in SDN. In: Proceedings of the International Engineering Conference on Electrical, Energy, and Artificial Intelligence (EICEEAI); 2023 Dec 27–28. Zarqa, Jordan. p. 1–5.