Computer Modeling in Engineering & Sciences

Tech Science Press

https://doi.org/10.32604/cmes.2026.082504

ARTICLE

5G-SliceMatch: A Slice-Aware Semi-Supervised Learning Framework for Malicious Traffic Detection in 5G Networks

Jinha Kim[1] **and Hwankuk Kim**[2,*]

[1]Department of Cyber Security, Kookmin University, Seoul, Republic of Korea
[2]Department of Information Security, Cryptography and Mathematics, Kookmin University, Seoul, Republic of Korea
*Corresponding Author: Hwankuk Kim. Email: rinyfeel@kookmin.ac.kr

Received: 17 March 2026; Accepted: 02 May 2026; Published: 30 June 2026

ABSTRACT: The advent of 5th Generation (5G) mobile networks has introduced Network Slicing as a core mechanism for supporting heterogeneous vertical services—such as enhanced Mobile Broadband (eMBB), Ultra-Reliable Low-Latency Communication (URLLC), and massive Machine-Type Communication (mMTC) over a shared physical infrastructure, thereby significantly expanding the attack surface at the User Plane Function (UPF). Securing this multi-slice environment requires intrusion detection systems that can simultaneously accommodate the statistical heterogeneity of per-slice traffic and the stringent Quality of Service (QoS) constraints of real-time slices, yet the practical cost of obtaining high-quality labeled traffic in operational 5G cores remains prohibitive. This study proposes 5G-SliceMatch, a Slice-Aware Semi-Supervised Learning framework, to address the challenge of malicious traffic detection in 5G network slicing environments under severe label scarcity. Traditional intrusion detection systems fail to account for the heterogeneous traffic characteristics of different 5G slices, leading to significant performance degradation in resource-constrained slices like URLLC. To overcome this, 5G-SliceMatch integrates a Slice-Aware Teacher Model with dedicated classification heads, slice-specific differential Feature Masking, and a progressive Self-Training strategy. Experimental results on the 5G-SliciNdd dataset demonstrate that 5G-SliceMatch consistently outperforms state-of-the-art baselines. Specifically, in an extreme scenario with only 1% of labeled data, 5G-SliceMatch achieved an F1-Score of 0.9393, outperforming XGBoost by 2.82%. Moreover, it achieves 98.1% of the performance of a fully supervised XGBoost (100% labels) using only 5% of the data, effectively reducing the manual labeling workload by 95%. This work proves that 5G-SliceMatch—by integrating slice-aware architectural design with efficient semi-supervised learning is critical for achieving high-resolution security visibility while satisfying the stringent QoS requirements of complex 5G network environments.

KEYWORDS: 5G network slicing; network IDS; semi-supervised learning

1 Introduction

The 5th Generation (5G) mobile communication has adopted Network Slicing as its core enabler to implement distinct service objectives-Enhanced Mobile Broadband (eMBB), Ultra-Reliable Low-Latency Communication (URLLC) and Massive Machine-Type Communication (mMTC)—within a single physical infrastructure [1,2]. While legacy networks relied on a 'Best-effort' approach providing uniform connection quality to all users, 5G allocates customized resources optimized for industry-specific requirements based on Software Defined Networking (SDN) and Network Function Virtualization (NFV) [3–5]. This paradigm shift is accelerating the digital transformation of Vertical Industries that demand high reliability, such as smart

factories, autonomous driving and telemedicine, making network security a prerequisite that determines the availability of national and industrial infrastructure beyond simple data protection [6].

However, as network flexibility and service complexity increase, security threats are evolving into more intelligent and complex forms. In particular, the User Plane Function (UPF), where traffic from all slices converges and terminates, serves as the gateway to the 5G Core Network, making it an optimal detection point for integrated traffic visibility as well as a primary target for attackers and a critical security vulnerability [7,8]. Since the UPF simultaneously processes traffic from multiple slices with different Quality of Service (QoS) policies, a detection failure at this point can lead to Threat Contagion to other slices sharing the same physical infrastructure, beyond the disruption of a specific service [9,10]. For example, a large-scale DoS (Denial of Service) attack using mMTC devices could increase latency in a URLLC slice sharing the same UPF, potentially causing catastrophic accidents in autonomous driving or telemedicine systems.

Recently, research on Deep Learning-based Intrusion Detection Systems (IDS) has been active; however, existing studies that fail to reflect the complex characteristics of actual 5G operational environments face severe limitations in practical application. The most significant obstacle is the extreme statistical heterogeneity of traffic across slices [11]. According to 3GPP (3rd Generation Partnership Project) standards, eMBB exhibits massive packet sizes and a 'Bursty' transmission pattern to secure high bandwidth, whereas URLLC transmits a minimal amount of data periodically with ultra-low latency [12,13]. This difference in base distributions causes the Shortcut Learning problem, where a single detection model becomes overly reliant on the dominant statistics of a specific slice, ignoring subtle anomalies in other slices or misclassifying normal traffic as attacks [14]. Consequently, data heterogeneity obscures the model's Decision Boundary, degrading overall detection reliability. Furthermore, the high cost of obtaining high-quality labeled data and the problem of data scarcity are decisive factors hindering the actual deployment of deep learning models. In 5G environments, the combination of dynamic IP allocation and GTP-U (GPRS Tunneling Protocol-User Plane) tunneling makes it difficult to track sophisticated attack flows using traditional 5-tuple-based analysis alone [15]. The process of security experts manually analyzing slice types and attack types to assign labels whenever a new attack emerges is extremely time-consuming and costly. Even though there is a practical requirement to maintain high performance in extreme environments where less than 1% of traffic collected in real-world operations is labeled, most existing research is based on the unrealistic assumption that tens of thousands of labeled data points exist. This poses a risk of the model suffering from Overfitting or Confirmation Bias during the initial training phase with insufficient labels. Finally, there is a conflict between detection precision and the unique real-time requirements of 5G. As model complexity increases to enhance security through deeper feature analysis, Inference Latency increases proportionally. This implies that in URLLC slices, where even millisecond (ms)-level latency is unacceptable, the security system itself can become an obstacle that degrades Quality of Service (QoS). In particular, Feature Interference between classes that occurs when attempting to process distinct attack types with a single classifier leads to a vicious cycle of making the model heavier to improve detection accuracy.

This study proposes 5G-SliceMatch, a slice-aware semi-supervised learning framework designed to structurally resolve these multifaceted challenges. 5G-SliceMatch adopts a multi-head architecture that utilizes a shared encoder to extract common traffic features while deploying independent classification heads for each slice, fundamentally blocking statistical interference between them. Furthermore, to maximize generalization performance with minimal labeled data, the framework integrates a slice-aware feature masking strategy optimized for each slice's QoS profile and a τ-relaxation-based progressive self-training technique that consistently enhances the quality of pseudo-labels.

The key contributions of this paper are summarized as follows:

1. **Slice-Aware Heterogeneity Resolution:** Effectively resolved statistical interference and Shortcut Learning problems in heterogeneous slice environments through slice-specific differential feature masking and a Multi-head Architecture.
2. **High Label Efficiency:** Successfully captured potential patterns in unlabeled data through a semi-supervised learning loop even in extreme shortage scenarios where only 1% of total data is labeled, demonstrating over 95% cost reduction compared to full labeling.
3. **Domain-Specific Augmentation:** Proposed a masking strategy leveraging domain knowledge of 5G slicing rather than simple statistical noise. By applying differential masking ratios (eMBB/mMTC: 15%–25%, URLLC: 10%–15%) based on slice characteristics, this study secured detection robustness exceeding existing general augmentation techniques (e.g., SMOTE, Mixup).

2 Background

2.1 5G Network Slicing

Unlike the Evolved Packet Core (EPC) of 4G Long-Term Evolution (LTE), the 5G Core Network (5GC) utilizes a Service Based Architecture (SBA), which decouples Network Functions (NFs) into independent software modules and interconnects them via standardized APIs [3]. In an SBA, each NF operates in a relationship as a service producer and consumer, interacting through interfaces defined in 3GPP TS 23.501 [16]. This design enables the independent deployment and horizontal scaling of network functions, assuming flexible utilization within a cloud-native environment.

Key NFs constituting the 5GC perform clearly separated roles: the Access and Mobility Management Function (AMF) handles User Equipment (UE) registration, authentication, and mobility management, while processing Non-Access Stratum (NAS) signaling between the UE and the Core Network. The Session Management Function (SMF) manages the establishment, modification and release of Protocol Data Unit (PDU) sessions and controls the operation of the User Plane Function (UPF) via the N4 interface. As such, the 5GC is designed with a clear separation between the Control Plane and the User Plane, with the latter composed of the UPF. The UPF performs the actual processing and forwarding of user data traffic, handling packet routing and forwarding, Quality of Service (QoS) policy enforcement, traffic usage reporting and Uplink Classifier functions. Structurally, the UPF is the point that all data traffic originating from a UE must traverse before exiting to an external Data Network (DN), and it connects to the internet or private networks via the N6 interface. The security requirements for these external interfaces and the UPF are detailed in TS 33.501 [17].

Consequently, in the field of security, the UPF holds two contradictory implications: First, as traffic from all network slices converges here, it serves as the optimal point to gain integrated visibility across the entire network architecture. Second, as it is the first point where external attacks enter the internal network, a failure in detection at this point immediately impacts the service integrity of the entire slice. In a network slicing environment, these structural characteristics of the UPF are considered even more critical.

Building on this architecture, 5G simultaneously operates multiple network slices—such as eMBB, URLLC, and mMTC (Media-centric Internet of Things, MIoT)—over a single physical infrastructure, each tailored to distinct service requirements. Each slice is uniquely identified by Single-Network Slice Selection Assistance Information (S-NSSAI), ensuring independent resource allocation and specialized Quality of Service (QoS) policies [16,18].

To provide a concrete implementation of these slices, the industry utilizes the Generic Network Slice Template (GST) and its specific instances known as Network Slice Types (NEST), as defined in GSMA

NG.116 [19]. This study integrates the GSMA NEST attributes with the 3GPP TS 23.501 5QI (5G QoS Identifier) mapping to define the operational boundaries of the proposed detection system. By deploying the 5G-SliceMatch framework at the N6 interface of the User Plane Function (UPF), it becomes possible to achieve integrated visibility across all traffic regardless of the slice type, while maintaining the statistical isolation required for high-accuracy analysis. Table 1 summarizes the standardized QoS requirements derived from this integration.

Table 1: Standardized QoS requirements for 5G network slices.

Slice Type	5QI Value	Latency (PDB)	Reliability (PER)
eMBB	1, 2, 5, 6, 7, 8, 9	100–300 ms	10^{-2}~10^{-6}
URLLC	82	10 ms	10^{-4}
mMTC	9	300 ms	10^{-6}

Note: PDB and PER are based on 3GPP TS 23.501 and GSMA NG.116. Note that 5QI 9 is shared by both eMBB and mMTC slices.

2.2 Semi-Supervised Learning

Semi-supervised Learning (SSL) is a technique that trains a model by leveraging both a small amount of labeled data and a large volume of unlabeled data. While supervised learning relies on sufficient labeled data, in real-world environments, the cost and time required for labeling are far greater than for data collection. SSL serves as a practical alternative for enhancing model generalization under these realistic constraints and has proven its effectiveness across various fields, including image and natural language processing [20].

Self-training is the most classical yet practical approach in SSL. Its fundamental principle is to treat unlabeled samples that the current model predicts with high confidence as Pseudo-Labels and include them in the training dataset to increase its volume, as illustrated in Fig. 1. The performance of Self-training heavily depends on the Confidence Threshold setting. If the threshold is too high, the utility of Unlabeled Data is low because few samples meet the requirement; if it is too low, it leads to the confirmation bias problem, where many erroneous Pseudo-Labels are generated, causing the model to learn in the wrong direction [21].

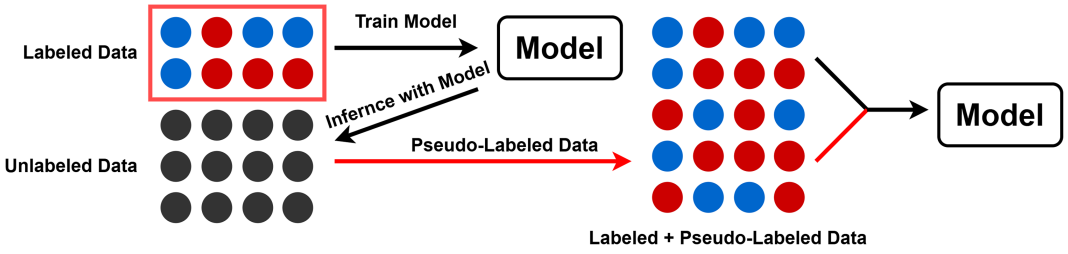

Figure 1: Self-training architecture.

Augmentation is a technique that introduces artificial variations into existing training data to increase data volume and improve model generalization. Recently, in SSL frameworks, augmentation is used as a core component for Consistency Regularization. Masking-based augmentation, which intentionally omits specific features of input data or replaces them with mean values, is receiving attention as a powerful means to prevent Shortcut Learning, where a model relies solely on specific identifying features to make predictions [22].

3 Related Work

This section reviews prior studies relevant to 5G-SliceMatch along two complementary axes that correspond to the core challenges addressed by the proposed framework. The first axis, *5G Network Security*, covers intrusion detection systems that explicitly target the architectural and protocol-level characteristics of 5G networks—including slicing, the service-based core, programmable user planes, and Open Radio Access Network (O-RAN) deployments—and establishes the domain-specific constraints within which any detector operating at the user plane must function. The second axis, *Label-Efficient Learning*, covers methodologies that mitigate the prohibitive cost of traffic labeling through semi-supervised learning, self-training, and consistency regularization, and provides the algorithmic foundation for operating under severe label scarcity. Table 2 summarizes the representative studies reviewed in each category, and the subsequent subsections discuss their contributions and limitations in detail.

Table 2: Summary of related work.

Category	Study	Dataset	Methodology	Key Results
5G Network Security	[23]	O-RAN KPI	Transformer, CNN, Ensemble, Soft Voting	Accuracy 98.3%, +1.43% over single model
	[24]	5G-NIDD	Dataset Construction, ML Benchmark	First 5G IDS benchmark established
	[25]	–	P4 Programmable Switch, Tree Classifier	Accuracy 98%, line-rate detection
	[26]	5G-NIDD	LSTM, GRU, Attention	Best on 5G-NIDD, early detection
	[27]	–	Protocol Analysis, Vulnerability Verification	DoS via GTP-U tunnel demonstrated
	[28]	DataCon, CIC-IDS, USTC	Evidential Learning, NIG Distribution, Autoencoder	AUC +3.7%, 2.55M parameters
	[29]	DataCon, CIC-IDS, USTC	Knowledge Distillation, Masking, Local-Global Attention	AUC +5.1%
	[30]	UNSW-NB15	Feature Selection, Grid Search, Decision Tree	Complexity reduced by 99.78%
Label-Efficient Learning	[31]	–	Semi-supervised Learning, Expert Feedback, SVM	Performance gain with minimal labels
	[32]	Unlabeled Traffic	Ensemble, Pseudo-labeling, Self-training	Improved detection accuracy and FNR
	[33]	–	Survey Analysis, Taxonomy	SSL with hierarchical detection identified
	[34]	CIFAR-10	Consistency Regularization, Pseudo-label, Augmentation	94.93% accuracy with only 250 labels

3.1 Intrusion Detection in 5G Network Environments

Research on Intrusion Detection Systems (IDS) in 5G network environments has evolved to address the structural characteristics of 5G architecture and its emerging threat vectors. Initial studies primarily focused

on applying standard deep learning techniques directly to intrusion detection [35]. Subsequent research has expanded to incorporate 5G-specific properties such as network virtualization and slicing.

Siriwardhana et al. [24] addressed the limitation of relying on outdated datasets that fail to reflect real-world 5G environments by constructing the 5G-NIDD dataset. Collected directly from the 5G Test Network (5GTN) at the University of Oulu, Finland, it captures realistic traffic, including normal mobile device activity and various DoS attacks (e.g., ICMP, UDP, SYN, HTTP Flood). By demonstrating the performance of machine learning models on this dataset, they established a benchmark that is now widely utilized in 5G intrusion detection research.

Akem and Fiore [25] proposed a real-time detection structure using P4-programmable switches as a User Plane Function (UPF) to perform machine learning inference directly within the user plane. By deploying tree-based classifiers as Packet Detection Rules (PDRs) on P4 switches, they achieved up to 98% accuracy in line-rate attack classification with minimal resource overhead. However, this approach is confined to in-UPF detection, differing from external service-based deployment structures via the N6 interface.

Djaidja et al. [26] tackled the latency issues inherent in flow-based IDS by proposing an early detection model that combines RNNs (LSTM/GRU) with attention mechanisms. By extracting features solely from the initial packet headers, they enabled detection before flow termination. While this model showed superior performance on 5G-NIDD, it remains limited by its reliance on a fully supervised learning environment with abundant labeled data.

Jeon et al. [30] proposed the Suboptimal Feature Selection Model (SFSM) for malicious traffic detection on resource-constrained lightweight devices. By combining permutation importance-based feature ranking with a grid search over a reduced candidate set, SFSM reduced latency and complexity by approximately 96% and 99.78%, respectively, compared to exhaustive search, while keeping the accuracy drop within 6%. This work highlights that judicious feature selection is a prerequisite for practical IDS deployment on devices with limited computational budgets—a constraint equally relevant to the UPF-level detection targeted in the present study.

Lee and Kim [23] proposed a malicious traffic detection methodology for Open RAN by leveraging the O-RAN AI/ML Framework. Their ensemble Transformer–CNN model, deployed as xApps or rApps through KServe, achieved an average accuracy of approximately 98.3% and demonstrated that soft voting outperforms hard voting by up to 1.06% in accuracy and 1.48% in F1-score. Crucially, inference latency was measured at 3.8–5.6 ms within the Near-RT RIC control loop, establishing the feasibility of real-time anomaly detection within O-RAN's 10 ms–1 s operational boundary. While this approach addresses RAN-layer KPI anomalies, it does not account for the statistical heterogeneity introduced by multi-slice traffic at the 5G Core level, which is the primary challenge addressed by 5G-SliceMatch.

Amponis et al. [27] investigated protocol vulnerabilities in the 5G core, specifically focusing on the PFCP (Packet Forwarding Control Protocol) at the N4 interface between the SMF and UPF. They demonstrated that unauthorized PFCP session control packets could cause DoS attacks, effectively disrupting GTP-U tunnels while bypassing standard security checks. This highlights that session-level security context, such as the Session Endpoint Identifier (SEID), must be a critical factor in IDS design.

3.2 Label-Efficient Learning for Intrusion Detection

Label-efficient learning has evolved from simple self-training to advanced semi-supervised learning methods to mitigate the high costs of data labeling.

Goernitz et al. [31] redefined anomaly detection as a semi-supervised learning task by integrating expert feedback. Their framework, which combines the distributional structure of unlabeled data with small amounts of labeled information, provided the theoretical foundation for hybrid approaches that integrate active and semi-supervised learning.

Alsulami et al. [32] proposed an ensemble-based automatic pseudo-labeling tool to leverage massive amounts of unlabeled traffic data. By granting pseudo-labels only to samples where multiple classifiers agree, they successfully reduced label noise, confirming significant improvements in both detection accuracy and False Negative Rate.

Mvula et al. [33] provided a comprehensive survey on the structural challenges of label scarcity in cybersecurity. They systematically categorized various methodologies—including active learning, self-training, and consistency regularization—and identified the need to combine semi-supervised learning with hierarchical detection structures as a key direction for future research.

Sohn et al. [34] proposed FixMatch, a state-of-the-art semi-supervised learning framework that achieves high performance by combining consistency regularization and pseudo-labeling. The core mechanism involves a "weak-to-strong" augmentation pipeline: the model generates a pseudo-label from a weakly-augmented version of an unlabeled sample, which is then used as a training target for its strongly-augmented counterpart, provided the prediction exceeds a high confidence threshold. This approach demonstrated remarkable label efficiency, achieving 94.93% accuracy on CIFAR-10 with only 250 labels, thereby proving that a simple yet rigorous consistency constraint can significantly mitigate the costs associated with data labeling.

4 Methodology

4.1 Problem Definition

When performing flow-based intrusion detection in the 5G network environment addressed in this study, several complex issues arise: (1) statistical heterogeneity across slices, (2) cross slice feature discrepancies, and (3) the high cost of data labeling. In particular, due to the large variance in feature distributions among eMBB, mMTC, and URLLC, Machine Learning (ML)-based intrusion detection under small-scale label conditions is highly likely to prioritize learning slice identity and slice-specific base-rates over the actual attack behavior itself. This leads to Overfitting according to slice data and induces models that are vulnerable to changes in attack patterns by slice, ultimately resulting in the degradation of generalization performance. Accordingly, this study defines these problems in detail and proposes a framework incorporating techniques designed to address each issue.

4.1.1 Problem 1: Statistical Heterogeneity across Slices

5G Network Slicing logically partitions a single physical infrastructure to allow eMBB, mMTC, and URLLC to meet different service requirements. Due to these structural characteristics, network traffic occurring in each slice has a distinct Statistical Distribution. For instance, the packet size or transmission volume of an eMBB slice supporting high-capacity streaming is overwhelmingly higher than that of a URLLC slice aimed at ultra-low latency; such numerical differences become the basis for determining which slice the traffic belongs to, regardless of whether it is an attack. The problem is that in an environment with such clear distribution differences, if the labeled data is extremely scarce, there is a high risk that the model will fall into Shortcut Learning by excessively relying on slice-specific base-rates or slice-identifying features within the training data, rather than learning the intrinsic patterns of actual malicious behavior. If the proportion of malicious samples in the URLLC slice happens to be high in a specific training set, the model might

misinterpret the distorted correlation of "High Rate -> URLLC Slice -> Malicious Traffic" as the correct answer instead of analyzing complex attack mechanisms. Such Shortcut Learning transforms the model's optimization goal from attack detection to slice estimation, which eventually causes an Overfitting problem where generalization performance drops sharply when deployed in a real network environment with different classifications from the training data. Therefore, the model must be prevented from becoming immersed in local information like slice identity and instead be encouraged to capture common attack signs that penetrate across slices. In this study, this problem is resolved using a technique called "Slice-Aware Feature Masking."

4.1.2 Problem 2: Cross-Slice Feature Discrepancies

Since each slice in a 5G Network Slicing environment is designed to meet different QoS requirements, a problem arises where the patterns of change shown by traffic features conflict across slices, even under the same attack scenario. This means that beyond simple differences in numerical feature distributions, the directionality of signatures for determining an attack itself changes. For example, when a DoS attack occurs, the URLLC slice, where low latency is key, might show a pattern of rapidly increasing packet transmission rates, whereas in mMTC for massive connectivity or eMBB slices centered on high bandwidth, the transmission rate might plummet or latency might abnormally spike due to network congestion. If a conventional model with a Single-head is applied in this situation, the model faces a scenario where it must simultaneously learn conflicting logics: "numerical increase = attack" and "numerical decrease = attack" for the same feature. The Gradient Conflict occurring during this process hinders training convergence and leads the model to converge to a low, generic level of accuracy that fails to achieve optimal performance in any slice. In this study, this problem is addressed through a technique called "Slice-Aware Teacher Model."

4.1.3 Problem 3: The High Cost of Data Labeling

Since 5G networks process high-speed, large-capacity traffic and use complex slicing protocols, labeling the accurate attack status for all collected flow data requires immense time and the intervention of professional experts. Due to these practical constraints, an Extreme Label Scarcity problem occurs in real operational environments, where only a tiny fraction of the total data can be labeled. When a model is trained relying only on a small amount of labeled data, there is a high risk that the model will fail to sufficiently learn the overall manifold of the data and will overfit to the noise or local patterns of specific samples. Conversely, although the vast amount of unlabeled data contains the latest attack trends and various volatilities of normal traffic, it is completely excluded from training in conventional Supervised Learning methods, resulting in the waste of important information. To compensate for this, a simple Self-training technique can be introduced; however, if incorrect predictions are adopted as Pseudo-labels due to the low reliability of the initial model, a Confirmation Bias problem occurs where errors gradually accumulate and propagate. Therefore, Semi-supervised Learning (SSL) is required to stably extract information from unlabeled data even in a label-scarce environment while minimizing error propagation to step-by-step enhance the model's generalization performance. In this study, this problem is resolved through "Progressive Self-Training" using Pseudo-labels and augmented data.

4.2 Overall Structure

Fig. 2 illustrates the deployment point of the Intrusion Detection System (IDS) proposed in this study. In a 5G network, traffic generated from the User Equipment (UE) is transmitted to an external Data Network (DN) through the Next Generation Node B (gNB) and the 5G Core's User Plane Function (UPF). The IDS in this study is deployed within the 5G Core, operating on the N3 interface between the gNB and the UPF. By intercepting and analyzing all traffic flows traversing the N3 interface in real-time, the IDS can

monitor traffic at an internal core network level before it reaches the UPF. This internal deployment approach offers two advantages. First, since the IDS operates inside the 5G Core on the N3 interface, it can inspect traffic prior to UPF processing, enabling earlier detection of threats within the user plane path. Second, by leveraging the standardized N3 interface, the IDS integrates naturally into the existing 5G architecture without requiring modifications to external network components, thereby maintaining the stability and integrity of the Core Network.

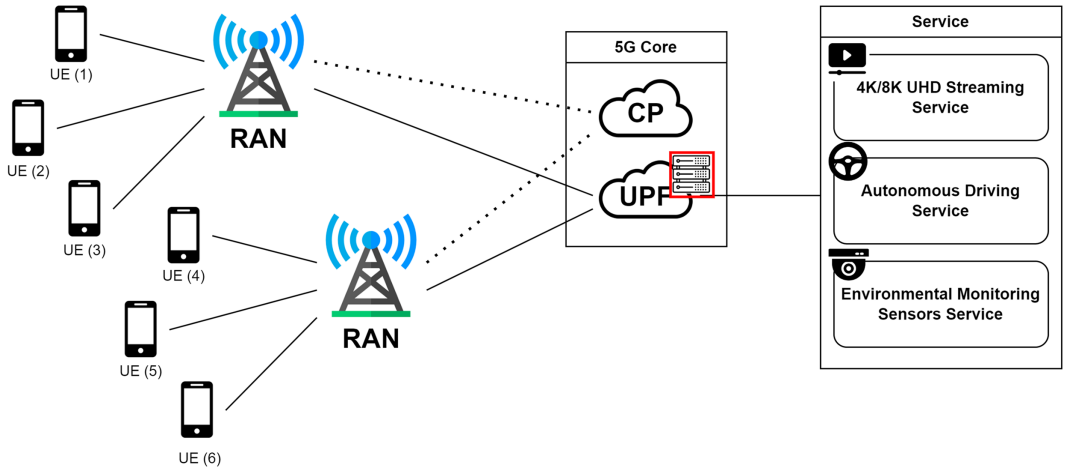

Figure 2: Deployment of the proposed IDS in 5G networks.

As shown in Fig. 3, the proposed 5G-SliceMatch framework consists of a model training process in four stages: Preprocessing, Masking & Augmentation, Pseudo-Labeling, and Self-Training. It operates through an Inference Process that ultimately determines the final detection results.

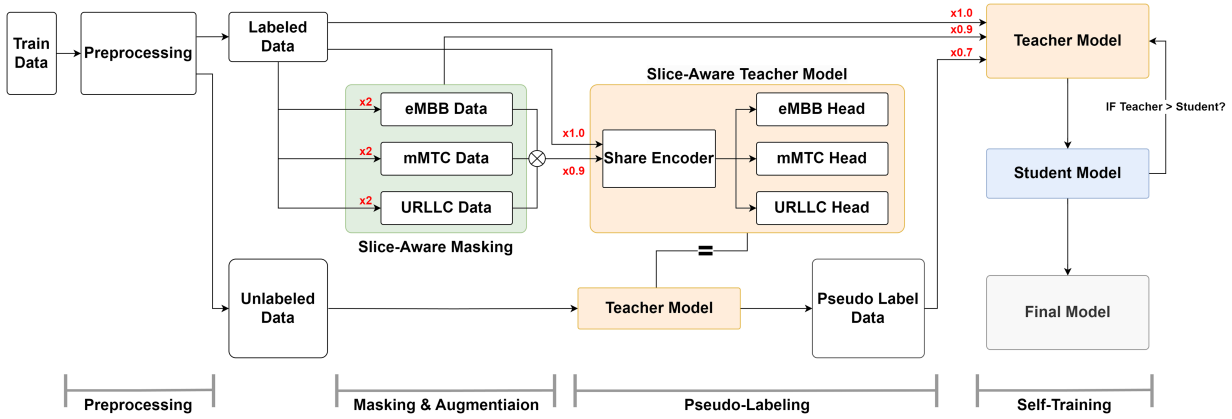

Figure 3: Architecture of the proposed 5G-SliceMatch framework.

4.3 Preprocessing

In the preprocessing stage, the 5G-SliciNdd raw data was consistently refined into a model input format prior to SSL. First, identifying features such as X, Seq, and Unique were removed, and sVid and dVid, which have extremely high missing rates, were excluded to maintain only features meaningful for training. The classification target was defined by converting the original Label into a binary format (Benign = 0, Malicious = 1), and slice information was stored separately for subsequent Slice-Aware routing. Input features

were separated into categorical and numerical types; categorical missing values were replaced with 'unknown' followed by One-hot Encoding, while numerical features were unified into numeric types with missing values imputed using the median value per slice. Through this process, the final input dimension was set to 69, combining 29 categorical and 40 numerical dimensions. Data splitting was performed based on slice and label data to preserve the joint distribution of slices and labels. The total dataset was split into Train/Validation/Test sets at a 70/10/20 ratio, and the Train set was further categorized into labeled and unlabeled data according to the ground-truth label ratios of 1%, 2%, 5%, 10%, 20%, and 100%. Finally, normalization was performed by applying the StandardScaler based on the labeled data, with the same transformation applied to the Validation, Test, and Unlabeled data. This procedure is a fundamental preprocessing design intended to simultaneously suppress distribution distortion and data leakage that may occur in environments with small amounts of labeled data.

4.4 Masking & Augmentation

Slice-Aware Masking is a targeted augmentation technique designed to prevent Shortcut Learning, where the model becomes preoccupied with statistical values representing slice identity and misses actual attack patterns. The core idea is to partially weaken the influence of "features that make it easy to guess the slice." After identifying the slice type (eMBB/mMTC/URLLC) of the 5G data, only feature groups with large distribution discrepancies in that slice are used as masking candidates. As shown in Table 3, when comparing specific slice features by their mean values and Coefficient of Variation (CV), features such as DstBytes or Rate show numerical differences ranging from tens to hundreds of times between slices, serving as indicators that identify the slice regardless of the attack status. By replacing these high-discrepancy features with the mean value of the corresponding slice, the model is forced not to rely on the absolute numerical values of the features. The masking ratio for features was selected using a differential strategy per slice: 15%–25% for eMBB, 10%–15% for URLLC, and 15%–20% for mMTC. The reason for setting a lower masking ratio specifically for URLLC is that features like Rate and TcpRtt serve as slice-distinguishing features while simultaneously playing a critical role in attack detection (Note: Rate and TcpRtt tend to decrease when malicious in eMBB slices, while they increase when malicious in URLLC slices.)

Table 3: Selected high-discrepancy features for slice-aware masking (subset).

Slice Group	Feature	eMBB	mMTC	URLLC	Ratio	CV
eMBB	DstBytes	82,679	4785	803	103.0×	1.28
eMBB	DstLoad	360,045	28,144	2,492,826	88.6×	1.14
eMBB	Load	387,531	38,107	2,557,828	67.1×	1.12
URLLC	SynAck	0.014	0.088	0.0004	228.0×	1.19
URLLC	DstRate	37	6	238	42.2×	1.10
URLLC	Rate	69	13	408	32.3×	1.07
mMTC	DstLoss	0.88	1.43	0.003	480.0×	0.76
mMTC	Loss	1.05	3.24	0.01	273.0×	0.94
mMTC	SrcLoss	0.17	1.77	0.01	198.0×	1.22

Note: Max/Min Ratio = $\max(\mu_{\text{eMBB}}, \mu_{\text{mMTC}}, \mu_{\text{URLLC}}) / \min(\mu_{\text{eMBB}}, \mu_{\text{mMTC}}, \mu_{\text{URLLC}})$.

The specific masking and augmentation procedure is a detailed algorithm that differentially assigns masking and augments data per slice, as described in Algorithm 1. For each labeled sample, k features are randomly selected from the feature group defined by slice type. The selected features are replaced with the mean values of the training data per slice rather than simple zeroing, thereby maintaining a realistic

data distribution while mitigating slice discriminability. The reason for not masking all features during the augmentation process is to preserve key signals for attack detection. In this study, 21 features with high slice discriminability are set as masking candidates, while features that commonly show a decreasing trend during attacks across all slices are excluded from masking. Since these common indicators provide universal patterns of malicious traffic regardless of slice type, the model—with its reliance on slice-identifying features lowered through feature masking—shifts its focus to assign higher weights to important information for attack detection.

Algorithm 1: Slice-aware masking

Require: Labeled set $\mathcal{D}_L$, unlabeled set $\mathcal{D}_U$, validation set $\mathcal{D}_{val}$, slice-wise means $\mu_{s,f}$, threshold schedule $\{\tau_t\}_{t=0}^{T-1}$, weights (w_L, w_A, w_P)
Ensure: Augmented dataset $\mathcal{D}_{aug}$

Slice-Specific Feature Masking Augmentation

$\mathcal{D}_{aug} \leftarrow \varnothing$
for $(x, y, s) \in \mathcal{D}_L$ **do**
  for $r = 1$ **to** 2 **do**
    $F_s \leftarrow$ mask feature set of slice s
    $\alpha \sim U([\alpha_s^{min}, \alpha_s^{max}])$ ▷ eMBB/mMTC: 0.15–0.25, URLLC: 0.10–0.15
    $k \leftarrow \max(1, \text{round}(\alpha|F_s|))$
    $M \leftarrow \textsc{RandomSubset}(F_s, k)$
    $x' \leftarrow x$
    for $f \in M$ **do**
      $x'[f] \leftarrow \mu_{s,f}$ ▷ replace with slice mean
    end for
    $\mathcal{D}_{aug} \leftarrow \mathcal{D}_{aug} \cup \{(x', y, s, w_A)\}$
  end for
end for

Finally, two augmented samples (x') with different masked features are generated from each ground-truth label sample (x), constituting the augmented dataset (D_{aug}). In a 5% label scenario, for instance, 1010 augmented data points are generated from 505 labeled data points and used for training alongside the original labeled data. At this stage, a lower confidence weight ($w_A = 0.9$) is assigned to the augmented data compared to the original data ($w_L = 1.0$) to precisely control the process so that artificially modified data does not excessively interfere with the model's optimization.

4.5 Pseudo-Labeling

If data generated through the Masking & Augmentation process is processed by a simple single model, a structural limitation arises where the model fails to understand the differences between slices. Therefore, this study adopts a decoupling structure that combines a Shared Encoder with Slice-specific Heads. The rationale for this design lies in the direction of feature changes observed during attacks in a 5G network environment, which conflicts depending on the slice service characteristics. To establish this claim empirically, the direction of attack-induced change is measured directly on the 5G-SliciNdd dataset across all 27 continuous-valued features. Identifier fields (X, Seq, UniqueID), categorical descriptors (Proto, State, Cause, sDSb, dDSb), high-missing-rate fields (sVid, dVid), and near-constant fields (sTos, dTos, sTtl, dTtl, sHops, dHops, SrcTCPBase, DstTCPBase, Offset, SrcWin, DstWin, SrcGap, DstGap) are excluded from this analysis, and no remaining feature is omitted. For each feature f and slice s, the signed log-ratio

$$\delta_{s,f} = \text{sign}(\mu_{s,f}^{\text{mal}} - \mu_{s,f}^{\text{ben}}) \cdot \log_2\left(1 + \frac{|\mu_{s,f}^{\text{mal}} - \mu_{s,f}^{\text{ben}}|}{|\mu_{s,f}^{\text{ben}}|}\right) \tag{1}$$

is computed, where $\mu_{s,f}^{\text{ben}}$ and $\mu_{s,f}^{\text{mal}}$ denote the mean feature value across benign and malicious samples of slice s, respectively. The features are then organized into eight functional categories (Timing, Rate, Load, Latency/RTT, Loss, Packet Size, Packet Count, Byte Volume). Within each category, $\delta_{s,f}$ is averaged across the constituent features and subsequently rescaled to $[-1, +1]$ by dividing by the maximum absolute value among the three slices. The signed log-ratio is used in place of a raw relative change because benign baselines differ across slices by several orders of magnitude, and a direct relative change would cause one slice with an extreme response to compress the others to near zero on the same axis; the log transform preserves the direction of every change while bringing the magnitudes onto a comparable scale.

Fig. 4 presents the resulting per-category directional displacement. Two distinct patterns emerge. As summarized in Table 4, six of the eight categories—Timing, Rate, Load, Latency/RTT, Loss, and Packet Size—exhibit a *conflict* pattern, in which URLLC moves toward +1 under attack while the corresponding eMBB (and usually mMTC) values move toward zero or into the negative range. The remaining two categories, Packet Count and Byte Volume, exhibit a *common* pattern in which all three slices decrease together and remain on the same side of the zero line. The conflict pattern is consistent with the structural characteristics of each slice: core performance indicators such as Rate, Load, and TcpRtt decrease in eMBB and mMTC under attack-induced resource exhaustion, whereas in URLLC—where ultra-low-latency management is strictly enforced—the same indicators rise sharply due to abnormal traffic injection and retransmission attempts.

This conflict in feature transitions can cause critical contradictions in conventional methods using a single classification head. An increase in the same feature value must be interpreted as a malicious signal in a specific domain while being interpreted as a normal signal in another. Consequently, gradients in opposing directions occur during the backpropagation process during model training, which ultimately hinders model convergence and causes the detection accuracy to deteriorate to a mediocre low level.

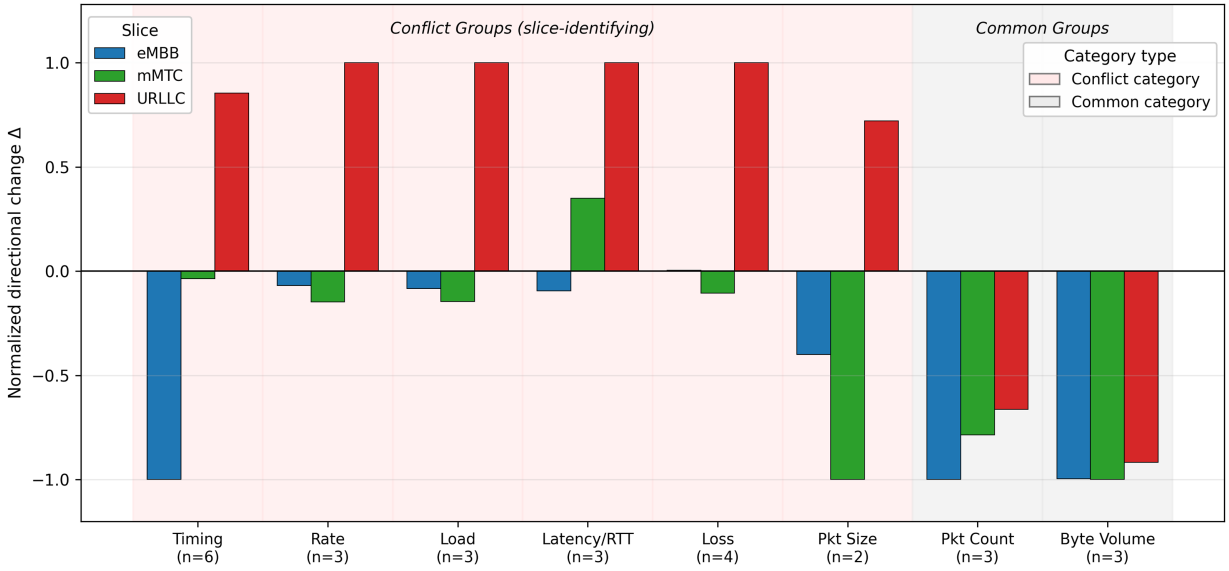

Figure 4: Empirical directional change of feature categories.

Table 4: Directional trends of feature changes during attacks across different slices.

Feature Group	eMBB	mMTC	URLLC	Consistency
Timing	↓	→	↑	**Conflict**
Rate	↓	↓	↑	**Conflict**
Load	↓	↓	↑	**Conflict**
Latency/RTT	↓	↑	↑	**Conflict**
Loss	→	↓	↑	**Conflict**
Packet Size	↓	↓	↑	**Conflict**
Packet Count	↓	↓	↓	**Common**
Byte Volume	↓	↓	↓	**Common**

Note: ↑: Increase, ↓: Decrease, →: Minimal change compared to benign traffic.

The proposed Slice-Aware Teacher Model resolves these conflicts by structurally isolating them. First, the Shared Encoder processes Common Indicators that show common decreasing patterns across all slices, such as sMeanPktSz or TotBytes, to learn universal malicious feature representations applicable regardless of the slice. Simultaneously, the dedicated heads for eMBB, mMTC, and URLLC placed on top of the Shared Encoder independently establish individual classification rules specialized for each slice environment. For example, the URLLC-specific head learns an increase in Rate as strong evidence of an attack, while the eMBB head uses a decrease in the same feature as a malicious signature. Ultimately, by routing each sample to the optimal head according to the slice metadata of the input traffic, the model minimizes inter-slice interference and secures high-precision detection performance that reflects the specificities of each service even in a label-scarce environment. A primary Teacher Model is constructed by inputting the existing ground-truth labeled data (D_L) and the augmented data (D_{aug}).

Once the Slice-Aware Teacher Model is finalized, it undergoes an inference phase using the unlabeled data (D_U) as input to generate Pseudo-Labels. In this stage, with the Confidence Threshold set to 0.9, Pseudo-Labels are initially generated to constitute the pseudo-labeled dataset (D_{pseudo}), which will be used as a dataset in the next stage, Self-Training.

Generalization to Unknown Slice Types. A practical concern in dynamic 5G deployments is that the set of active slice types may not always conform to the three canonical categories (eMBB, mMTC, URLLC) assumed during training. To address this, the proposed architecture incorporates a dedicated *fallback classification head*, which is structurally identical to the slice-specific heads (Linear(64→32) + ReLU + Dropout(0.1) + Linear(32→1)) but is not bound to any particular slice identity. During training, a slice-specific head is activated only when its corresponding slice contributes at least $N_{\min} = 5$ labeled samples; slices below this threshold are excluded from dedicated head training and their flows are instead routed to the fallback head. During inference, any flow whose slice metadata does not match a trained head—whether due to insufficient training coverage or the appearance of a previously unseen slice type—is automatically routed to the fallback head. This design ensures *graceful degradation*: rather than producing a prediction failure or requiring architectural modification, the model falls back to a general-purpose classifier that leverages the shared encoder representations. As a result, 5G-SliceMatch remains operationally robust in environments where slice configurations are dynamic or partially unknown, without compromising detection performance on well-represented slices.

4.6 Self-Training

The final stage of the model training phase is the Progressive Self-Training process, which maximizes the model's generalization performance by integrating Pseudo-label data (D_{pseudo}) into the training process. This process is based on the previously designed Slice-Aware Masking and Slice-Aware Teacher Model architecture, performed through a knowledge transfer loop between the Teacher Model and the Student Model. Algorithm 2 describes this iterative optimization procedure in detail.

Algorithm 2: Progressive self-training

Require: Labeled set $\mathcal{D}_L$, unlabeled set $\mathcal{D}_U$, validation set $\mathcal{D}_{val}$, slice-wise means $\mu_{s,f}$, threshold schedule $\{\tau_t\}_{t=0}^{T-1}$, weights (w_L, w_A, w_P)
Ensure: Final teacher model $M_T^\star$

Progressive Self-Training

for $t = 0$ **to** $T-1$ **do**
  $\tau \leftarrow \tau_t, \mathcal{D}_{pseudo} \leftarrow \varnothing$
  for $x_u \in \mathcal{D}_U$ **do**
    $p \leftarrow M_T(x_u, \text{slice}(x_u))$
    if $p > \tau$ **then**
      $\mathcal{D}_{pseudo} \leftarrow \mathcal{D}_{pseudo} \cup \{(x_u, 1, \text{slice}(x_u), w_P)\}$
    else if $p < 1 - \tau$ **then**
      $\mathcal{D}_{pseudo} \leftarrow \mathcal{D}_{pseudo} \cup \{(x_u, 0, \text{slice}(x_u), w_P)\}$
    end if
  end for
  if $|\mathcal{D}_{pseudo}| = 0$ **then**
    continue
  end if
  Rebuild $\mathcal{D}_{aug}^{(t)}$ by Stage 2 (optional re-sampling)
  Train student M_S on $\mathcal{D}_L(w_L) \cup \mathcal{D}_{aug}^{(t)}(w_A) \cup \mathcal{D}_{pseudo}(w_P)$
  $F_S \leftarrow \text{F1}(M_S, \mathcal{D}_{val})$
  if $F_S > F_T$ **then**
    $M_T \leftarrow M_S, F_T \leftarrow F_S, M_T^\star \leftarrow M_S$
  else
    break ▷ early stop if no validation improvement
  end if
end for
return $M_T^\star$

At each iteration (t) of training, the Teacher Model identifies the slice information of individual flows included in the generated pseudo-labeled data (D_{pseudo}) and calculates the malicious probability (p) through the corresponding slice-dedicated head. High-confidence samples exceeding the threshold are selected according to a predefined confidence threshold schedule ($\{\tau_t\}$) to assign Pseudo-Labels. This dynamic threshold application acts as a safeguard against the Confirmation Bias problem, where erroneous classification results from the Teacher Model in the early stages of training are transferred to the Student Model, thereby degrading performance. Once the pseudo-label dataset (D_{pseudo}) is constructed, the new Student Model performs training by integrating the original labeled data (D_L), the augmented data with Slice-Aware masking applied (D_{aug}), and the newly generated pseudo-label data (D_{pseudo}). Different weights

(w_L, w_A, w_P) are applied to the loss function based on the data confidence to control the process so that pseudo-label data, which carries relative uncertainty, does not distort the model's overall parameter updates.

The trained model is evaluated based on the F1-Score using a separate validation dataset (D_{val}). If the Student Model's performance outperforms the current Teacher Model, it is updated to become the Teacher Model to lead pseudo-label generation in the next iteration. This process continues until improvement in validation performance ceases, resulting in a final detection model ($M_T^\star$) that ensures a deep understanding of the overall data manifold and responds fluidly to slice-specific attack scenarios.

4.7 Inference Process

After the training and validation processes conclude, an inference process is performed to evaluate the final real-world detection performance of the proposed framework. In this stage, this study utilizes the 20% test dataset (D_{test}) that was completely isolated from model training and validation in the data preprocessing stage. This ensures the prevention of Data Leakage and allows for an objective validation of the model's practical generalization ability regarding new network flows. The inference process is based on the final Teacher Model generated through Progressive Self-Training. The input test data undergoes the same preprocessing and is converted into potential features via the model's Shared Encoder. Subsequently, the slice metadata assigned to the flow is identified, and it is routed to the corresponding dedicated head (eMBB, mMTC, or URLLC) to determine the presence of an attack based on the probability value calculated by the head. The detection results derived through this process demonstrate how precisely the proposed method reflects the heterogeneous slice environment of 5G networks to detect intrusions even in label-scarce environments.

5 Experimental Design

5.1 Dataset

This study utilizes 5G-SliciNdd, a derivative dataset of the 5G Network Intrusion Detection Dataset (5G-NIDD) publicly released by Siriwardhana et al. [24]. The original 5G-NIDD was collected from the 5G Test Network (5GTN) at the University of Oulu, Finland, providing a commercial-grade 5G environment based on the Nokia Flexi Zone Indoor Pico base station. Traffic was captured at a Multi-access Edge Computing (MEC) server; attack traffic was generated from Raspberry Pi 4 devices connected via Huawei 5G modems, while normal traffic was generated from actual mobile devices. The dataset includes nine attack types across two categories: DoS attacks (ICMP Flood, UDP Flood, SYN Flood, HTTP Flood, and Slowrate DoS) and Port Scans (SYN Scan, TCP Connect Scan, and UDP Scan).

The 5G-SliciNdd dataset extends the original 5G-NIDD by consolidating three physically separated slice-specific datasets into a unified file. The original 5G-NIDD was collected from three distinct 5G network slices (eMBB, mMTC, and URLLC), each producing independent traffic captures. The 'predicted' column in 5G-SliciNdd preserves the original slice membership of each flow, reflecting the actual slice from which it was captured rather than a heuristic classification. Consequently, these slice labels carry no estimation noise, as they are ground-truth identifiers derived from the network configuration at the time of data collection. It consists of 14,456 labeled network flows with 52 attributes. After removing identifiers (X, Seq, UniqueID) and high-missing-rate features (sVid 97.4%, dVid 99.9%), and applying One-hot Encoding to five categorical features (Proto, sDSb, dDSb, Cause, State), the final input vector is composed of 69 features (40 numerical + 29 categorical). Table 5 summarizes the dataset characteristics.

Table 5: 5G-slicing dataset statistics.

Item	Value
Total Flows	14,456
Normal/Malicious	7057 (48.8%)/7399 (51.2%)
eMBB/mMTC/URLLC	5808/4615/4033
Number of Features	69
Attack Types	9 (5 DoS + 4 Port Scan)
Capture Environment	5GTN, Oulu, Finland

Missing values in numerical features are imputed with the median value per slice and normalized using a StandardScaler fitted to the labeled training data. The data is split into Train (70%, 10,119 flows), Validation (10%, 1446 flows), and Test (20%, 2891 flows) sets using stratified sampling to preserve the slice type and label ratio across all partitions. To simulate a label-scarce environment, this study defines six label ratio scenarios (1%, 2%, 5%, 10%, 20%, 100%) within the training set. In each scenario, only the designated training data retains its labels, while the rest are treated as unlabeled data. Table 6 details the data availability for each scenario. To address the high variance inherent in extreme low-label regimes, all experiments were conducted across 20 independent random seeds, yielding 120 total runs (20 seeds × 6 label ratios). The seeds were constructed from three groups to ensure split diversity: (1) three original seeds (42, 123, 456) retained for backward compatibility, (2) commonly adopted benchmark seeds in reproducibility-oriented ML studies (789, 1024, 2024, 3141, 9999), and (3) twelve additional seeds spanning a wide numerical range (314, 1111, 1357, 2222, 2468, 2718, 5555, 6174, 7777, 8888, 27182, 31415) to minimize systematic bias in stratified sampling. All reported metrics are the mean ± standard deviation over the 20 runs.

Table 6: Label ratio scenarios.

Label Ratio	Labeled Data	Unlabeled Data	Malicious Samples
1%	101	10,018	17
2%	202	9917	34
5%	505	9614	85
10%	1011	9108	172
20%	2023	8096	344
100%	10,119	0	1722

This study uses the Macro F1-Score as the primary evaluation metric, AUC-ROC for threshold-independent assessment, and Precision and Recall for false positive/negative analysis. Furthermore, this study reports slice-specific F1-Scores for eMBB, mMTC, and URLLC individually to verify performance differences between slice types.

5.2 Experimental Sets

The shared encoder comprises three fully connected layers with output dimensions of 256, 128, and 64, respectively. Each of the first two layers is followed by Batch Normalization, ReLU activation, and Dropout (rate = 0.3), while the third layer applies a reduced Dropout rate of 0.2 to preserve the 64-dimensional representation passed to the slice-specific heads. Each slice-dedicated head (eMBB, mMTC, URLLC) consists

of a Linear(64→32) layer followed by ReLU, Dropout (rate = 0.1), and a final Linear(32→1) output layer, yielding 2081 parameters per head. A structurally identical fallback head is activated for any slice that does not meet the minimum training sample threshold (N_{min} = 5). The model is optimized using Adam (lr = 0.001, weight decay = 10^{-4}) with a ReduceLROnPlateau scheduler (factor = 0.5, patience = 10), a batch size of 32, and early stopping with patience of 15 epochs over a maximum of 200 epochs. The loss function is BCEWithLogitsLoss with sample-wise weighting, applying weights of w_L = 1.0, w_A = 0.9, and w_P = 0.7 to labeled, augmented, and pseudo-labeled samples, respectively. The complete hyperparameter configuration is summarized in Table 7.

Table 7: Hyperparameter configuration of 5G-SliceMatch.

Component	Hyperparameter	Value
Shared Encoder	Layer 1	Linear(69→256) + BN + ReLU + Dropout(0.3)
	Layer 2	Linear(256→128) + BN + ReLU + Dropout(0.3)
	Layer 3	Linear(128→64) + BN + ReLU + Dropout(0.2)
	Output dim	64
	Total parameters	52,736
	Activation	ReLU
Slice-Specific Heads	Architecture	Linear(64→32) + ReLU + Dropout(0.1) + Linear(32→1)
	Heads	eMBB, mMTC, URLLC (+ fallback)
	Parameters per head	2081
Training	Optimizer	Adam
	Learning rate	0.001
	Weight decay	10^{-4}
	Batch size	32
	Max epochs	200
	Early stopping patience	15
	LR scheduler	ReduceLROnPlateau (factor = 0.5, patience = 10)
Semi-Supervised	τ schedule	0.95 → 0.90 → 0.85 → 0.80 → 0.75
	Max ST iterations	5
	Pseudo-label weight (w_P)	0.7
	Augmentation weight (w_A)	0.9
	Augmentations per sample (n_{aug})	2
Loss	Function	BCEWithLogitsLoss
	Reduction	Sample-weighted mean

This study designs three experiments to evaluate (1) the overall effectiveness of the proposed method compared to existing baselines, (2) the individual contribution of the slice-specific feature masking component, and (3) the individual contribution of the progressive self-training component. All experiments share the same data split, preprocessing pipeline, and evaluation protocol described in Section 5.1.

5.2.1 Experimental 1: Overall Performance Comparison

The first experiment evaluates the overall performance of the proposed Full Model (Feature Masking + Self-Training) by comparing it against six baseline models across the entire range of label ratios (1%–100%).

As summarized in Table 8, Traditional Machine Learning. Traditional machine learning techniques are utilized to establish performance benchmarks. Random Forest defines the performance floor in label-scarce environments using a bagging-based ensemble ($n\_estimators$ = 300), while XGBoost acts as a performance ceiling in label-sufficient environments through robust gradient boosting ($n\_estimators$ = 500, max_depth = 6, lr = 0.1).

Table 8: Summary of baseline model configurations.

Model	Category	Key Characteristics & Settings
Random Forest	Traditional ML	Bagging, n_estimators = 300, class_weight = balanced
XGBoost	Traditional ML	Boosting, n_estimators = 500, max_depth = 6, lr = 0.1
MLP	Deep Learning	Same backbone as proposed (Dense-BN-ReLU-Dropout)
MLP+SMOTE	Augmentation	SMOTE applied (k = 3, 3× oversampling)
FixMatch	SSL	Consistency reg., τ = 0.95, weak/strong augmentation
LabelSpreading	SSL	k-NN kernel (n_neighbors = 7, α = 0.2)
ConMD	Augmentation	Masking reconstruction + local-global attention + distillation
UnDiff	Deep Learning	Evidential learning, NIG distribution, AE backbone, 2.55M params

Note: NIG: Normal-Inverse Gaussian, AE: Autoencoder.

Deep Learning and Augmentation. The MLP shares the exact same architecture (Dense-BN-ReLU-Dropout) as the proposed model, ensuring that any performance gap is attributed to the proposed components (Feature Masking, Self-Training, etc.) rather than structural differences. Additionally, MLP+SMOTE applies a general-purpose oversampling technique (k = 3, 3× expansion) to directly compare the performance between slice-specific Feature Masking and conventional feature space augmentation strategies.

Semi-Supervised Learning (SSL). Two methods are employed to compare against state-of-the-art SSL frameworks. FixMatch combines consistency regularization with pseudo-labeling (τ = 0.95) to verify the superiority of the proposed method over general-purpose SSL that lacks slice-aware design. LabelSpreading utilizes a k-NN-based graph propagation technique ($n\_neighbors$ = 7, α = 0.2) and subsamples unlabeled data to 5000 instances to manage computational complexity.

Inference Throughput and Latency Measurement. To evaluate the real-time applicability of 5G-SliceMatch in operational 5G environments, this study measures per-model inference throughput and P99 latency and benchmarks the results against the QoS Packet Delay Budget (PDB) requirements defined in 3GPP TS 23.501 for each slice type: 10 ms for URLLC, 100–300 ms for eMBB, and 300 ms for mMTC. All latency measurements are conducted on a standardized CPU-only environment to reflect realistic edge deployment conditions where GPU acceleration may not be available. For each model, the full preprocessed test set ($\mathcal{D}_{test}$, 2891 flows) is passed through the inference pipeline in a single batch, and wall-clock time is recorded using high-resolution system timers. Throughput is computed as the total number of flows divided by the elapsed inference time (flows/sec). P99 latency is measured by repeating the single-sample inference 1000 times and reporting the 99th percentile of the recorded latency distribution, thereby capturing tail latency behavior that is most relevant to URLLC slice constraints. For 5G-SliceMatch specifically, the measured latency includes the full inference path: shared encoder forward pass, slice metadata lookup, and slice-specific head routing. This end-to-end measurement ensures that the reported latency reflects the complete per-flow decision pipeline rather than the encoder alone.

All baselines use the same preprocessed 69-dimensional input and are evaluated across all six label ratio scenarios (1%, 2%, 5%, 10%, 20%, 100%).

5.2.2 Experimental 2: Ablation Study on Feature Masking

The second experiment analyzes the contribution of slice-specific Feature Masking (FM) from three perspectives.

Part A: ON/OFF Pairwise Comparison

This study constructs four pairs of models where FM is the only variable. In each pair, the model architecture (MLP or SliceHead) and the use of Self-Training (ST) are fixed, while only the application of FM varies. Table 9 presents the configuration. The $\Delta F1 = F1_{FM\ ON} - F1_{FM\ OFF}$ in each pair quantifies the pure contribution of FM in that specific environment. If $\Delta F1 > 0$ is consistently observed across the four distinct environments, the robustness of FM's contribution is verified, regardless of the presence of other components.

Table 9: Feature masking ablation pair of models.

Pair	FM OFF → FM ON	Arch	ST	Verified Effect
1	MLP → MLP+FM	MLP	OFF	FM alone
2	MLP+ST → MLP+FM+ST	MLP	ON	FM with ST
3	Slice → Slice+FM	SliceHead	OFF	FM with SliceHead
4	Slice+ST → Slice+FM+ST	SliceHead	ON	FM in full setting

Beyond the aggregate F1 differences between MLP and SliceHead architectures that emerge from this pairwise design, the underlying reason for the gap is measured at the gradient level. For the standard single-head MLP baseline (Pair 1, FM OFF), the inter-slice gradient cosine similarity is computed at every training epoch. At the end of each epoch, three balanced mini-batches are drawn separately from the labeled training split—one from each slice (eMBB, mMTC, URLLC)—and the gradient of the binary cross-entropy loss with respect to the encoder parameters is computed independently for each mini-batch. The pairwise cosine similarity

$$\cos(\theta_{ij}) = \frac{\langle g_i, g_j \rangle}{\|g_i\| \, \|g_j\|} \tag{2}$$

is evaluated for all three slice pairs, where a negative value corresponds to destructive interference—an update favorable to slice i simultaneously increases the loss of slice j. The fraction of epochs within a 10-epoch sliding window in which $\cos(\theta_{ij}) < 0$ is reported as the *conflict frequency* for each slice pair. In parallel, the training loss and validation Macro F1 trajectory of the same single-head baseline are recorded to verify whether the gradient conflict translates into observable convergence obstacles. All gradient-level measurements are conducted at the 10% label ratio and averaged over the 20 random seeds introduced in Section 5.1.

Part B: Comparison of Augmentation Strategies

To evaluate the advantages of the proposed slice-specific masking over alternative augmentation methods, this study compares five strategies using the same MLP backbone without Self-Training.

- *No Augmentation*: A control group trained only on D_L without augmentation.
- *SMOTE*: k-NN interpolation-based synthetic oversampling (k = 3, 3× expansion). While it generates synthetic data via interpolation between minority class samples in the feature space, it has the limitation of potentially creating physically impossible values (e.g., fractional packet counts).

- *Gaussian Noise*: A simple perturbation method adding Gaussian noise (σ = 0.1) to all features.
- *Uniform FM*: Replaces 15%–25% of all 69 features with the global mean, without slice distinction. This configuration isolates the effect of the masking technique itself from the effect of slice-specific design.
- *Slice FM (Proposed)*: The slice-specific differential masking technique described in Section 4.4.

Through this comparison, this analysis incrementally isolates whether performance gains stem from (a) the effect of augmentation itself, (b) the effect of the feature masking technique, or (c) the effect of slice-specialized design.

Part C: Sensitivity Analysis of Masking Ratio

To verify that the proposed masking ratio provides an optimal balance, this study compares three levels of masking intensity.

- *Low* (5%–10%): Conservative masking to minimize information loss.
- *Mid* (10%–25%, Proposed): Slice-specific differential application (eMBB/mMTC: 15%–25%, URLLC: 10%–15%).
- *High* (25%–40%): Aggressive masking to maximize regularization, but with risks of information loss.

Excessive masking ratios may destroy information critical for attack detection, leading to performance degradation, while ratios that are too low may have negligible effects on preventing shortcut learning. This analysis identifies the optimal balance point between these two conflicting factors.

Part D: Sensitivity to Mask Replacement Value

Since Algorithm 1 replaces masked features with the slice mean, a static constant, an additional experiment is conducted to verify whether the choice of replacement value has any measurable impact on detection performance. Three alternative replacement strategies are compared against the proposed Static Mean while keeping the mask-candidate feature set and the masking ratio fixed:

- *Static Mean (Proposed)*: Each masked feature is replaced with the per-slice mean computed from the labeled training data, providing a deterministic and parameter-free neutral substitution anchored at the center of the slice-specific feature distribution.
- *Empirical Sampling*: Each masked feature is replaced by a value drawn uniformly at random from the observed values of that feature within the same slice, introducing stochastic in-distribution variation across augmented copies.
- *Bounded Noise*: Each masked feature is replaced with the slice mean perturbed by uniform noise bounded within one standard deviation, i.e., $\mu_{s,f} + \mathcal{U}(-\sigma_{s,f}, +\sigma_{s,f})$, ensuring plausible substitutions while introducing controlled stochasticity.
- *Gaussian Feature Noise*: Each masked feature is replaced with a value sampled from $\mathcal{N}(\mu_{s,f}, \sigma^2_{s,f})$, representing the most stochastic strategy and allowing occasional out-of-distribution substitutions in high-variance features.

Each strategy is evaluated across all six label ratios with 20 random seeds per configuration. Performance is reported along four axes—Macro F1, eMBB F1, mMTC F1, and URLLC F1—so that slice-specific effects that may be obscured by the aggregate Macro score are exposed. Paired t-tests are applied between the Static Mean strategy and each alternative at every combination of label ratio and metric.

5.2.3 Experimental 3: Ablation Study on Self-Training

The third experiment analyzes the contribution of Progressive Self-Training (ST) from two perspectives.

Part A: ON/OFF Pairwise Comparison

Symmetrically to Experiment 2, this study constructs four pairs of models where ST is the only variable. Table 10 presents the configuration. In each pair, the architecture and FM settings are fixed, and only the application of ST varies. This study quantifies the pure ST contribution as $\Delta F1 = F1_{ST\ ON} - F1_{ST\ OFF}$. By verifying this across four environments—combining the presence or absence of FM and SliceHead, this study determines whether the benefits of ST are consistent regardless of other components.

Table 10: Self-Training ablation pair of models.

Pair	ST OFF → ST ON	FM	Arch	Verified Effect
1	MLP → MLP+ST	OFF	MLP	ST alone
2	MLP+FM → MLP+FM+ST	ON	MLP	ST with FM
3	Slice → Slice+ST	OFF	SliceHead	ST with SliceHead
4	Slice+FM → Slice+FM+ST (Full)	ON	SliceHead	ST in full setting

Specifically, Pair 2 (MLP+FM → MLP+FM+ST) and Pair 4 (Slice+FM → Full) measure the additional effect of ST when FM is already applied, serving as a basis for judging the synergy between the two components. This is because if FM improves the quality of the initial Teacher, it leads to more accurate pseudo-label generation, which can amplify the effectiveness of ST.

Part B: Convergence and Pseudo-Label Quality Analysis

Aggregated ΔF1 values alone are insufficient to explain why ST is effective. To compensate for this, this study tracks two process-level metrics for each Self-Training iteration. First, this study records the validation F1 trajectory. This study logs the F1-Score on the validation set after each ST iteration to analyze whether the progressive τ relaxation schedule (0.95 → 0.90 → 0.85 → 0.80 → 0.75) achieves monotonic improvement and how many iterations are required for convergence.

Second, this study measures the accuracy of Pseudo-Labels. Since the ground-truth labels for the unlabeled data D_U are available in this dataset (though excluded during training), this study can directly measure the accuracy of generated pseudo-labels against the actual labels. This allows tracking changes in pseudo-label quality as τ is relaxed iteration by iteration and identifying the point at which quality degradation begins.

Part C: Calibration

To evaluate whether the confidence threshold τ used in the Self-Training loop functions as a meaningful probabilistic signal rather than a heuristic filter, this study additionally measures the Expected Calibration Error (ECE) of 5G-SliceMatch across label ratios. For each label ratio, predictions on the validation set are partitioned into 10 equal-width confidence bins, and the absolute difference between the mean predicted probability and empirical accuracy within each bin is weighted by the bin frequency to yield ECE. A lower ECE indicates closer alignment between predicted confidence and empirical correctness, which is a prerequisite for reliable pseudo-label selection under progressive τ-relaxation. This analysis is conducted at 1%, 5%, and 10% label ratios with 20 random seeds per configuration and is complemented by reliability diagrams for visual inspection.

6 Experimental Result

This section describes the experimental results, categorized into three parts as outlined in the experimental design. Section 6.1 compares the overall performance of the proposed method with six baseline models (Experiment 1). Section 6.2 evaluates the contribution of slice-specific feature masking

(Experiment 2), and Section 6.3 analyzes the contribution of Self-Training through controlled ablation studies (Experiment 3).

6.1 Experimental 1: Overall Performance Comparison

6.1.1 F1-Score Comparison

Fig. 5 visualizes these results across all label ratios, clearly illustrating the dominance of 5G-SliceMatch (red) in the low-label regime. The widening gap between 5G-SliceMatch and the baselines as the label ratio decreases, together with its notably tighter error bars, visually confirms both the performance superiority and the reproducibility advantages discussed above.

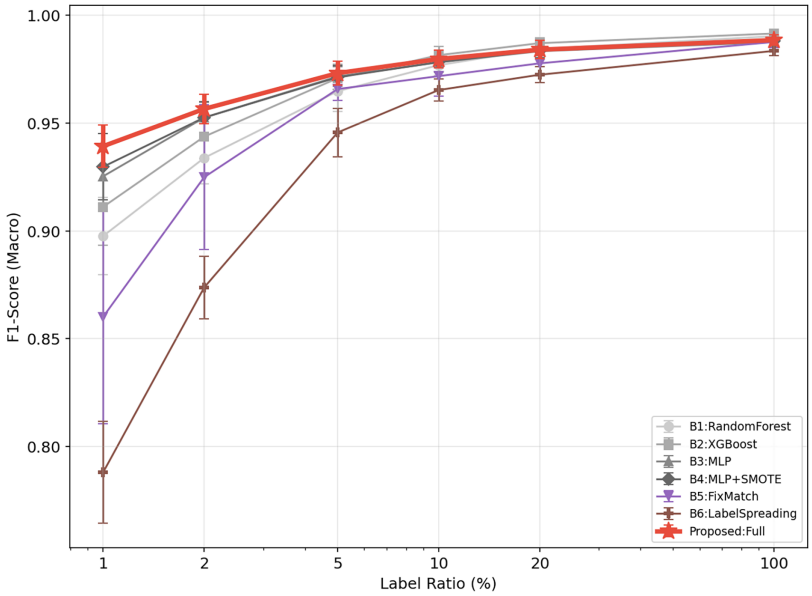

Figure 5: Overall performance.

Table 11 shows the F1-Score (Macro) of all models across eight label ratio scenarios (0.5%, 0.8%, 1%, 2%, 5%, 10%, 20% and 100%), averaged over 20 random seeds per configuration. 5G-SliceMatch decisively outperforms all nine competing models across every label-scarce scenario, establishing a clear and consistent dominance in the regime that matters most for real-world 5G deployments, where labeled slice data is inherently expensive to obtain. In the extreme 0.5% label ratio (approximately 50 labeled samples), 5G-SliceMatch reached an F1-Score of 0.8901, decisively outperforming every baseline: 0.0302 over XGBoost, 0.0158 over MLP, 0.0209 over ConMD, 0.2063 over FixMatch, and a remarkable 0.2178 over LabelSpreading. This margin is particularly striking given that ConMD emerges as a strong competitor at higher label ratios; under extreme scarcity, its performance collapses while 5G-SliceMatch remains robust. At 0.8%, 5G-SliceMatch (0.9321) continues to lead the field, outperforming the second-best ConMD (0.9244) by 0.0077 and widening the gap against XGBoost (0.8940) to 0.0381, confirming that its advantage is not a one-off but a systematic property of the method. At the critical 1% label ratio (101 labeled samples), 5G-SliceMatch reached 0.9393, surpassing XGBoost by 0.0282, MLP by 0.0140, FixMatch by 0.0946, and LabelSpreading by a decisive 0.1514. Even the strongest competing baseline, MLP+SMOTE (0.9297), trails by 0.0096—a gap that is both statistically meaningful and practically relevant. Critically, 5G-SliceMatch records the lowest variance among all nine models across these label-scarce ratios, proving that its superiority is not an artifact of favorable data splits but a reliable, reproducible characteristic observed across 20 independent random seeds.

This combination of higher mean performance and lower variance is a decisive advantage: 5G-SliceMatch is not merely better on average, it is more trustworthy in deployment. At 2%, 5G-SliceMatch (0.9566) once again leads the field, maintaining its top position against ConMD (0.9565) and extending its streak of dominance across four consecutive label-scarce ratios—no other model achieves this level of consistency. Beyond 2%, XGBoost and ConMD edge ahead on the mean F1, but the margins are negligible (within 0.003 at 5%, within 0.002 at 10%, and 0.003 at 20%), and this regime is precisely where semi-supervised learning is no longer needed: when labels are abundant, any competent supervised baseline suffices, and the defining value of 5G-SliceMatch lies in the label-scarce regime where it is unrivaled. The most compelling evidence of this practical superiority emerges from the label efficiency analysis: 5G-SliceMatch achieves 0.9732 at just 5% labels, recovering 98.1% of the performance that XGBoost requires 100% labels to reach (0.9916). Even more remarkably, at only 0.8% labels, 5G-SliceMatch already recovers 94.0% of this full-label performance. This translates to a reduction in labeling workload of up to 99.2% with negligible accuracy loss—an efficiency gain that no competing method in Table 11 comes close to matching.

Table 11: F1-score (Macro) comparison across label ratios (mean over 20 seeds).

Model	0.5%	0.8%	1%	2%	5%	10%	20%	100%
5G-SliceMatch	**0.8901**	**0.9321**	**0.9393**	**0.9566**	0.9732	0.9798	0.9841	0.9886
XGBoost	0.8599	0.8940	0.9111	0.9437	0.9709	**0.9816**	**0.9871**	**0.9916**
MLP	0.8743	0.9143	0.9253	0.9525	0.9717	0.9782	0.9834	0.9876
MLP+SMOTE	0.8683	0.9154	0.9297	0.9526	0.9713	0.9784	0.9837	0.9883
RF	0.8679	0.8914	0.8976	0.9338	0.9648	0.9769	0.9844	0.9904
FixMatch	0.6838	0.8879	0.8447	0.9186	0.9634	0.9722	0.9794	0.9876
LabelSpread	0.6723	0.7566	0.7879	0.8737	0.9457	0.9654	0.9725	0.9835
ConMD	0.8692	0.9244	0.9280	0.9565	**0.9735**	0.9779	0.9798	0.9700
UnDiff	0.4291	0.5028	0.4227	0.4403	0.4397	0.4152	0.3816	0.3563

Note: Bold values indicate the best performance per column; bold entries for 5G-SliceMatch highlight label-scarce regimes (≤2%) where the proposed model achieves the highest F1-score.

6.1.2 Slice-Wise Analysis

Table 11 shows the F1-Score (Macro) of all models across eight label ratio scenarios (0.5%, 0.8%, 1%, 2%, 5%, 10%, 20% and 100%), averaged over 20 random seeds per configuration. 5G-SliceMatch achieved the highest F1-Score in the four most label-scarce scenarios (0.5%, 0.8%, 1%, and 2%), demonstrating its effectiveness in environments where existing methods are most vulnerable. In the extreme 0.5% label ratio (approximately 50 labeled samples), 5G-SliceMatch reached an F1-Score of 0.8901, showing improvements of 0.0302 over XGBoost, 0.0158 over MLP, 0.2063 over FixMatch, and 0.2178 over LabelSpreading. Notably, even ConMD (0.8692), which becomes one of the strongest baselines at higher label ratios, falls 0.0209 below 5G-SliceMatch, indicating that its advantage erodes sharply under extreme label scarcity. At 0.8%, the gap remains substantial: 5G-SliceMatch (0.9321) leads the second-best ConMD (0.9244) by 0.0077 and XGBoost (0.8940) by 0.0381. At the 1% label ratio (101 labeled samples), 5G-SliceMatch reached 0.9393, showing improvements of 0.0282 over XGBoost, 0.0140 over MLP, 0.0946 over FixMatch, and 0.1514 over LabelSpreading. Although MLP+SMOTE (0.9297) ranked second, it still showed a performance gap of 0.0096. In addition, 5G-SliceMatch records the lowest variance among all nine models at these label-scarce ratios, indicating that the reported gains are consistent across 20 independent random splits rather than artifacts of favorable data splits. As the label ratio increases, the performance gap between 5G-SliceMatch and the baselines narrows. At 2%, 5G-SliceMatch (0.9566) remains the top performer but leads ConMD (0.9565)

by only 0.0001, placing the two methods in a practically equivalent range. From 5% onward, ConMD (at 5%) and XGBoost (at 10%, 20%, 100%) slightly surpass 5G-SliceMatch on the mean F1, with differences staying within 0.003 at 5%, within 0.002 at 10%, and 0.003 at 20%. Beyond 2%, XGBoost and ConMD surpass 5G-SliceMatch; however, since the strength of 5G-SliceMatch is focused on label-scarce environments, additional gains from semi-supervised learning become unnecessary in label-sufficient environments. A practically significant observation appears in the label efficiency analysis: 5G-SliceMatch achieved a score of 0.9732 at 5% labels, recovering 98.1% of the performance of XGBoost trained with 100% labels (0.9916) using only 5% labeled data. More strikingly, at just 0.8% labels, 5G-SliceMatch already recovers 94.0% of this full-label performance, implying that the labeling workload can be reduced by up to 99.2% while still retaining competitive classification accuracy. As shown in Fig. 6, at both 1% and 2% label ratios, 5G-SliceMatch achieves the highest F1-Score across all three slices (eMBB, mMTC and URLLC).

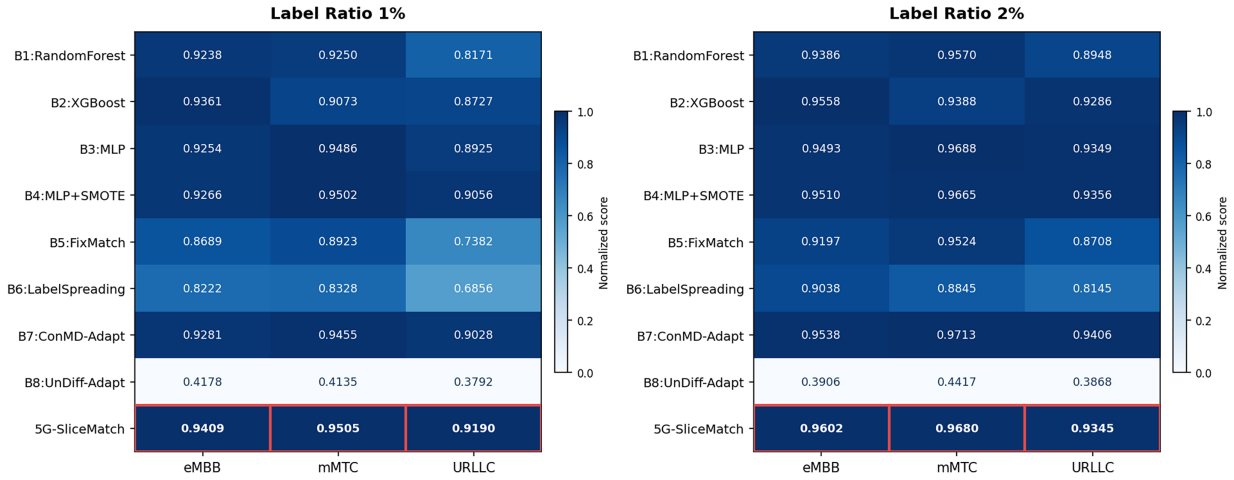

Figure 6: Slice-wise F1-score heatmap at 1% and 2% label ratios.

6.1.3 Per Slice Inference Latency CDF

Figs. 7 and 8 present the inference throughput and P99 inference latency measured via Cumulative Distribution Function (CDF) of all models, benchmarked against the QoS Packet Delay Budget (PDB) requirements of each 5G slice type.

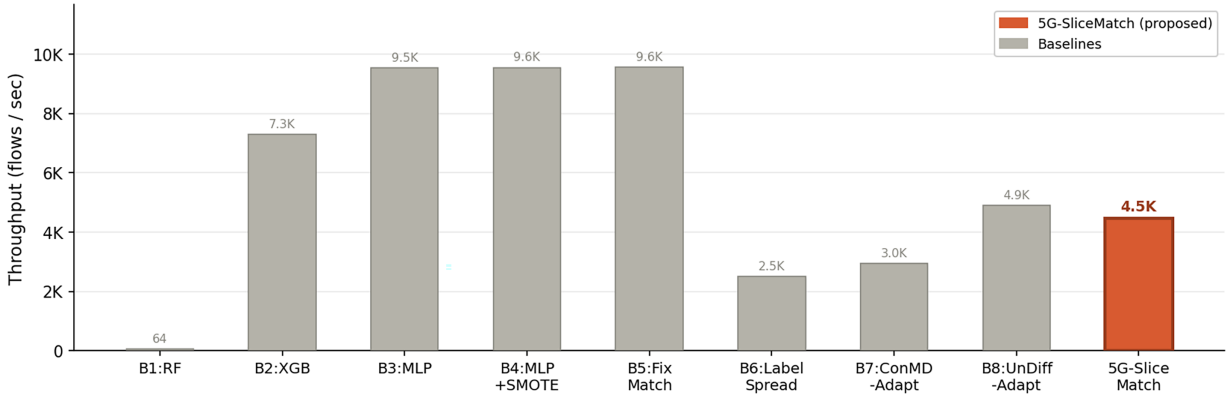

Figure 7: Per-model inference throughput comparison.

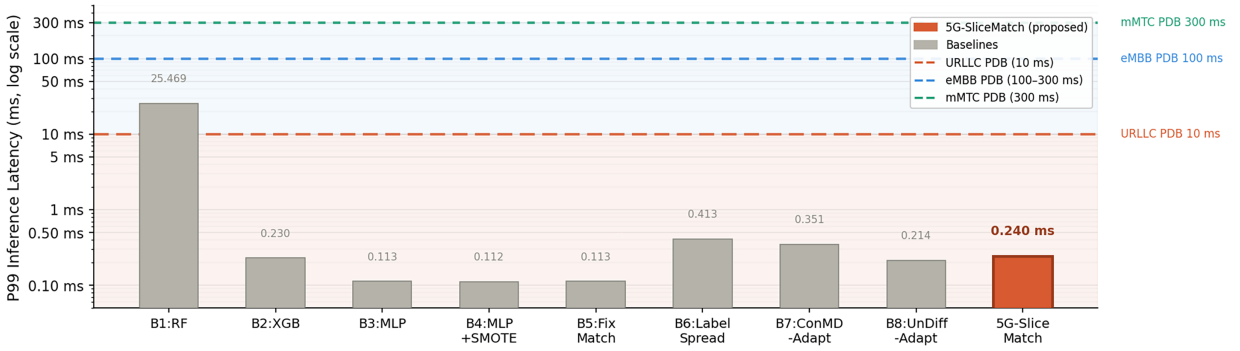

Figure 8: Per-model P99 inference latency vs. QoS PDB requirements.

In terms of throughput, 5G-SliceMatch processes 4500 flows per second (4.5K flows/sec), outperforming LabelSpreading (2.5K) and ConMD-Adapt (3.0K) despite the additional routing overhead introduced by its multi-head architecture. However, it falls below MLP-based models (9.5–9.6K) and XGBoost (7.3K), a gap attributable to the structural overhead of per-slice head computation and batch routing. Random Forest records the lowest throughput of all models at 64 flows/sec, a consequence of its single-threaded tree traversal mechanism.

In terms of latency, 5G-SliceMatch achieves a P99 inference latency of 0.240 ms, which satisfies the eMBB PDB (100–300 ms) and mMTC PDB (300 ms) requirements by a margin exceeding 41×, and likewise meets the URLLC PDB (10 ms) threshold by more than 40×. This demonstrates that the shared encoder effectively suppresses latency growth even under a multi-head routing structure. Among the baselines, Random Forest is the only model to exceed the URLLC PDB at 25.469 ms, rendering it inapplicable in real-time 5G environments. MLP-based models (0.112–0.113 ms), XGBoost (0.230 ms), and UnDiff-Adapt (0.214 ms) achieve comparable or marginally lower latency than 5G-SliceMatch; however, these models were shown in prior experiments to underperform 5G-SliceMatch in detection accuracy under label-scarce conditions. Collectively, these results demonstrate that 5G-SliceMatch is the only model to simultaneously satisfy the QoS requirements of all 5G slice types—including URLLC—at a P99 latency of 0.240 ms, while maintaining superior detection performance in data-constrained scenarios.

6.2 Experimental 2: Ablation Study on Feature Masking

In this section, this study verifies the contribution of the slice-specific Feature Masking (FM) based on the four analytical perspectives (Parts A–D) defined in Section 5.2.2.

6.2.1 Part A: Pair-Wise ON/OFF Comparison

Fig. 9 quantifies the contribution of FM by comparing four model pairs where FM is the only variable. Across all four pairs, FM demonstrates a consistent positive impact. At a 1% label ratio, all pairs show positive ΔF1, with an average contribution of +0.0161. This trend holds through 2% (avg. +0.0076) and 5% (avg. +0.0036). Of the 24 total pair-ratio combinations, 21 cases (87.5%) show a positive contribution, while the three negative cases are negligible (<0.001) and occur only at label ratios ≥10%. Notably, in Pair 2 (MLP+ST), FM provides the largest contribution (+0.0271 at 1%), suggesting that FM and Self-Training create a synergistic effect. By enhancing the initial Teacher Model quality, FM facilitates more accurate pseudo-label generation, leading to iterative improvements. In contrast, Pairs 3 and 4 (SliceHead) show smaller FM contributions (1% ratio: +0.0099 and +0.0069, respectively), as the SliceHead architecture already performs structural pattern separation, partially overlapping with the role of FM. Slice-wise, the eMBB slice benefits

the most (+0.0319 at 1%), as its extreme discrepancies in volume features (e.g., 103× difference in DstBytes) make it the most vulnerable to shortcut learning without masking.

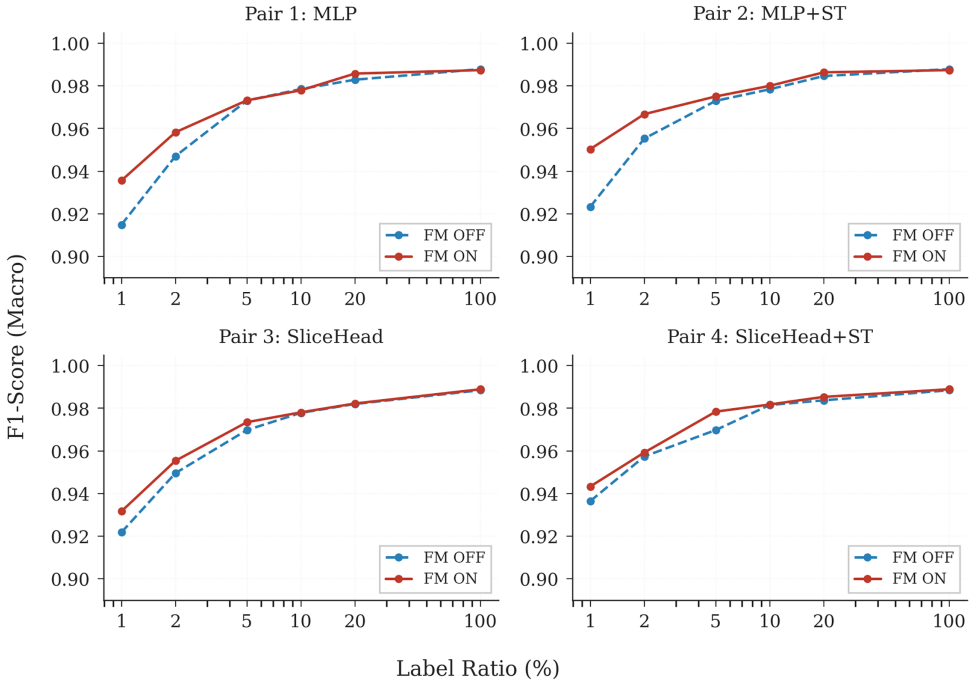

Figure 9: F1-score comparison between FM ON and OFF pairs across label ratios.

Beyond the aggregate Δ F1 values, the gradient-level measurements on the single-head MLP baseline provide a mechanistic explanation for the observed performance gap. Fig. 10a presents the epoch-wise inter-slice gradient cosine similarity. The eMBB↔URLLC pair remains below the zero line throughout all 100 epochs, with values concentrated between −0.05 and −0.30, confirming persistent destructive interference. The eMBB↔mMTC pair shows mildly positive alignment during the first 40 epochs, then crosses into negative territory. The mMTC↔URLLC pair oscillates near zero throughout training. Fig. 10b reports the corresponding conflict frequency, which reaches 1.0 for eMBB↔URLLC from the first sliding window onward, indicating that destructive interference between these two slices is a constant feature of the optimization rather than a transient effect.

The impact of this conflict on training is visible in Fig. 10. Although the training loss decreases smoothly (Fig. 10c), the validation Macro F1 trajectory (Fig. 10d) reveals an extended plateau between epochs 40 and 100, during which F1 grows only marginally from 0.95 to 0.97. This plateau coincides with the window in which the eMBB↔mMTC conflict frequency climbs from 0.1 to 1.0 and the eMBB↔URLLC conflict frequency remains saturated. The temporal alignment indicates that once multiple slice pairs enter persistent gradient conflict, the single-head model spends most of its remaining training budget reconciling contradictory updates rather than improving on any individual slice, which explains why FM alone cannot close the gap to SliceHead-based variants (Pair 3 and Pair 4 in Table 9).

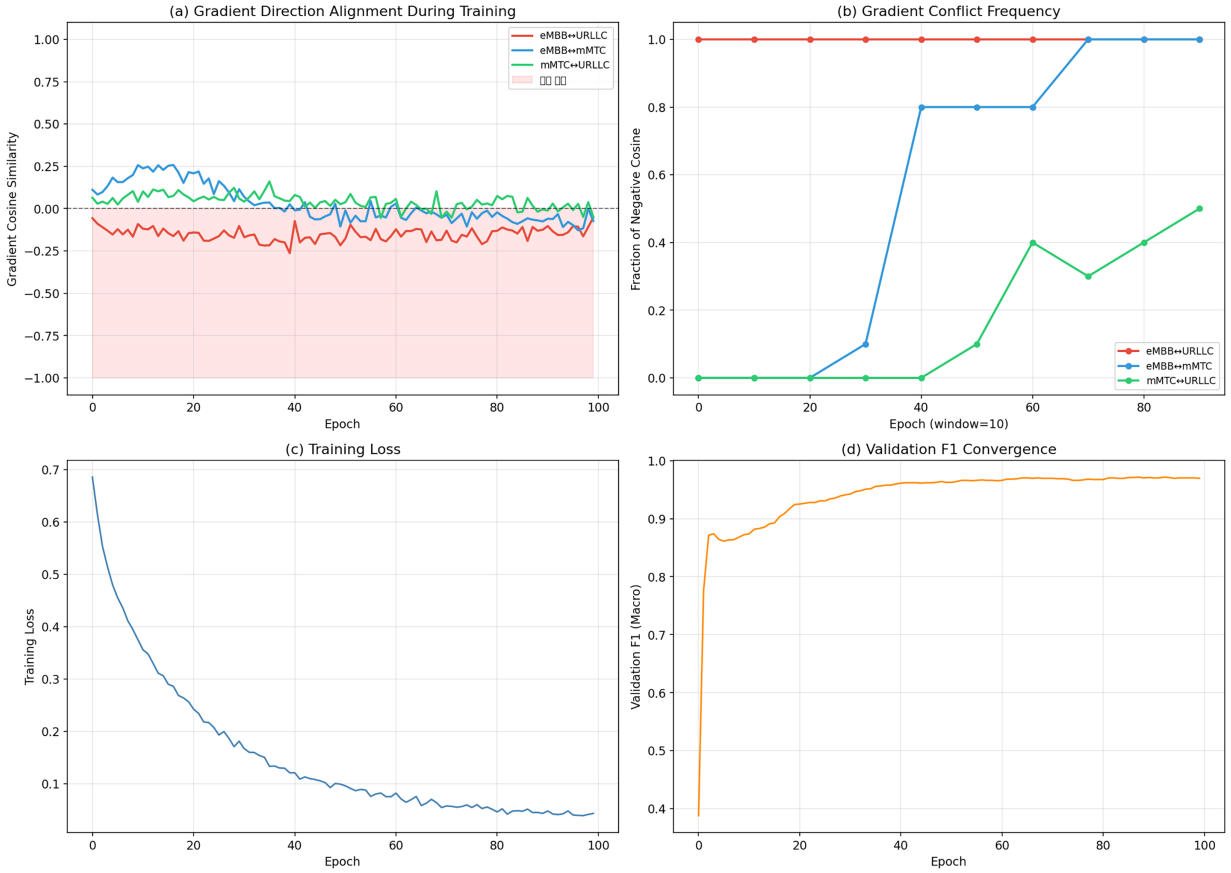

Figure 10: Gradient cosine similarity analysis of single-head MLP.

6.2.2 Part B: Augmentation Strategy Comparison

As shown in Fig. 11, all augmentation methods outperform the non-augmented baseline (avg F1: 0.9641), confirming the general benefit of augmentation in label-scarce settings. Three key findings emerge:

1. **SMOTE** achieves the highest performance (0.9405) at 1% by synthesizing samples via k-NN interpolation. However, at the average F1-Score (0.9700), it is nearly identical to Slice FM (0.9697) while suffering from the risk of generating physically impossible values (e.g., fractional packet counts).
2. **Uniform FM (0.9652)** yields only marginal gains over the non-augmented baseline, lagging significantly behind Slice FM. This confirms that the effectiveness of masking depends on *what* is masked (slice-specific design) rather than the masking technique itself. Random masking often inadvertently destroys critical attack signals.
3. **Slice FM (Proposed)** achieves the best performance at 2% (0.9583) and 20% (0.9858). Gaussian Noise (0.9688) performs better than Uniform FM, as applying uniform noise results in less information loss than replacing features with global means.

6.2.3 Part C: Masking Ratio Sensitivity

Fig. 12 compares three masking intensities (Low, Mid, High) against the baseline. All three intensities consistently outperform the non-masked baseline (0.9148 at 1%). At 1%, High (0.9385) slightly outperforms Low (0.9356) and Mid (0.9356). The performance differences between these settings are minimal, and

they converge as the label ratio increases beyond 10%. This indicates that the core mechanism of FM is the selection of target features rather than the specific ratio. While the 10%–25% range is optimal, the conservative 10%–15% ratio for URLLC remains a critical design choice to protect sensitive features required for low-latency attack detection.

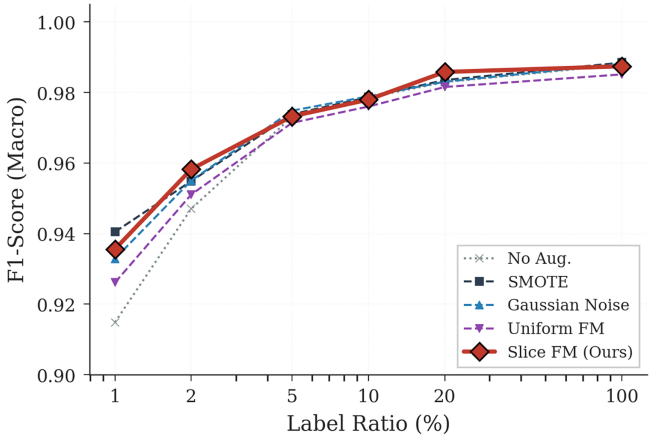

Figure 11: Augmentation strategy comparison.

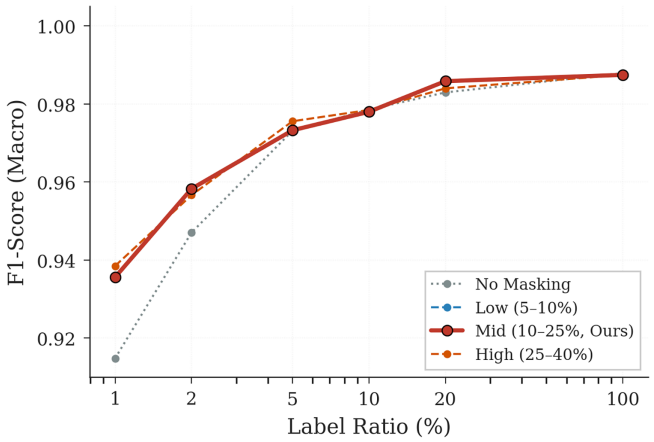

Figure 12: Feature masking ratio sensitivity.

6.2.4 Part D: Robustness across Mask Replacement Strategies

Fig. 13 presents the Δ F1 of three alternative mask replacement strategies relative to the proposed Static Mean across three label ratios (1%, 5%, 10%). All deviations fall within the ±0.001 band, and Wilcoxon signed-rank tests confirm that none of the alternatives produces a statistically significant difference from Static Mean at any label ratio ($p > 0.05$ in all nine comparisons).

At the 1% label ratio, Empirical Sampling yields a marginal gain of +0.0001, while Bounded Noise and Gaussian Feature Noise register slight degradations of −0.0004 and −0.0001, respectively. At 5%, the pattern is consistent: Empirical Sampling again provides a negligible improvement (+0.0001), whereas both noise-based strategies result in minor performance losses (−0.0002 each). At 10%, Empirical Sampling shows the largest positive deviation (+0.0007) and Bounded Noise a small gain (+0.0003), while Gaussian Feature Noise exhibits the only notable negative deviation (−0.0008); however, even this extreme remains within the negligible range and does not reach statistical significance ($p = 0.2774$).

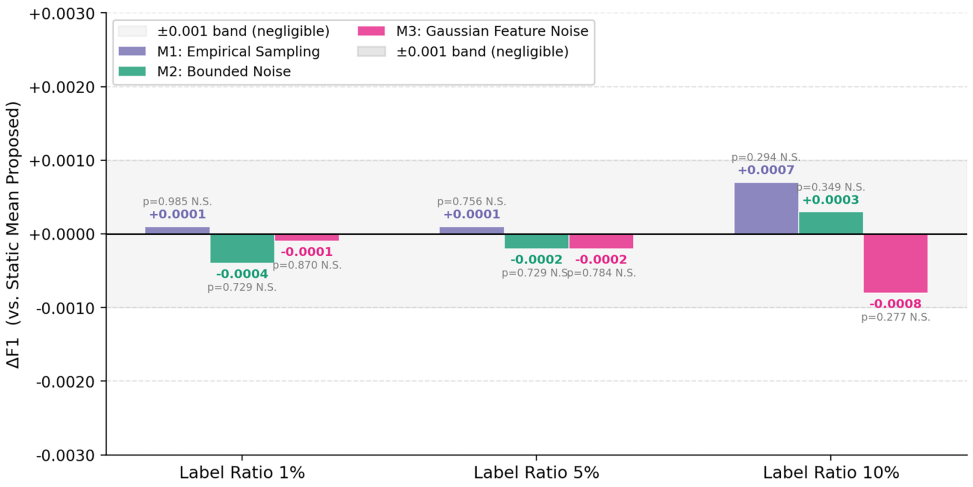

Figure 13: F1 of alternative mask replacement strategies vs. static mean.

These results demonstrate that the detection performance of 5G-SliceMatch is robust to the choice of mask replacement value. The insensitivity arises because the primary mechanism of Slice-Aware Masking is the *selection* of high-discrepancy features to suppress, not the specific value used for substitution. Once slice-identifying features are weakened, the model's optimization naturally shifts focus toward universal attack indicators regardless of whether the replacement is a static slice mean, an empirically sampled value, or bounded noise. Consequently, Static Mean is retained as the default strategy for its simplicity, determinism, and the absence of additional hyperparameters, without incurring any measurable performance penalty.

6.3 Experimental 3: Ablation Study on Self-Training

This section validates the contribution of Progressive Self-Training (ST) based on the three analyses defined in Section 5.2.3.

6.3.1 Part A: Pair-Wise ON/OFF Comparison

Fig. 14 quantifies the contribution of ST using four model pairs where ST is the unique variable. In the 1% label ratio scenario, all four pairs yield a positive ΔF1, with an average contribution of +0.0124. This positive trend continues at 2% (avg. +0.0071). As label availability increases, the contribution rapidly diminishes; at 100%, the contribution is exactly zero, as no unlabeled data remains to be processed. Among 24 total combinations, 17 (70.8%) show a positive contribution, and in the target scenario (1%–2%), all 8 cases (100%) demonstrate a positive impact of ST.

Pair 2 (MLP+FM) and Pair 3 (SliceHead) exhibit the highest contribution at 1% (+0.0147). The synergy in Pair 2 suggests that FM enhances the initial Teacher's quality, leading to more accurate pseudo-labels that amplify the subsequent ST effect. Meanwhile, Pair 4 (Full Model) maintains consistently higher contributions than other pairs across the 5%–20% range, demonstrating that ST remains effective in intermediate label-scarce environments when all components are integrated.

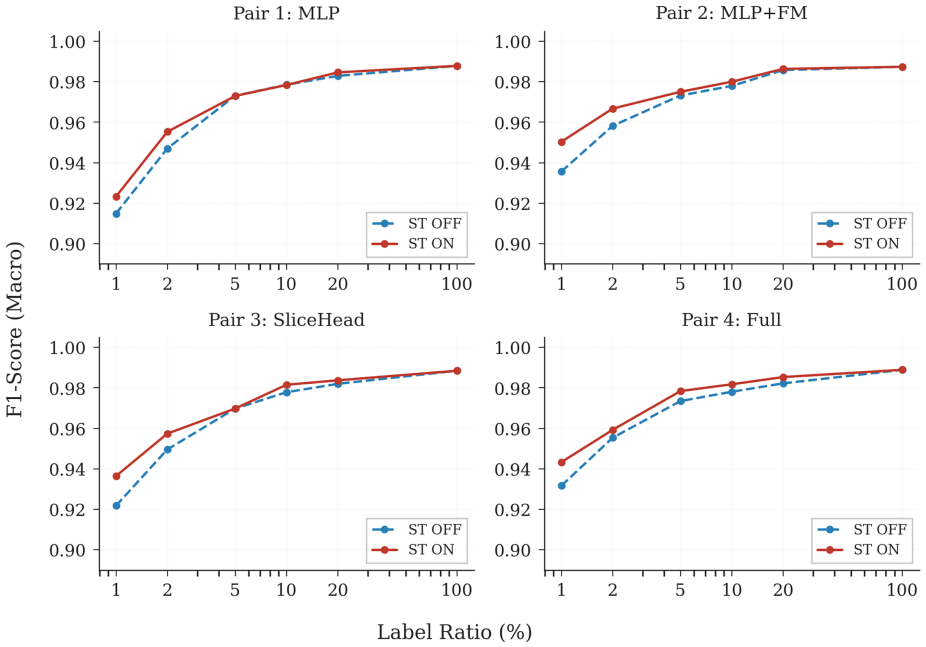

Figure 14: F1-score comparison between ST ON and OFF pairs across label ratios.

6.3.2 Part B: Configuration-Dependent Effects and Convergence

Fig. 15 visualizes the ST contribution via a heatmap, revealing three distinct patterns. First, the 1%–2% columns are entirely green (positive), confirming the universal benefit of ST in extreme label-scarce settings. Second, the 100% column shows zero contribution, validating that ST naturally deactivates when no unlabeled data exists. Third, the "Full" model row maintains a consistent green shade even in the 5%–20% range, highlighting how the integration of all components extends the utility of ST across a broader spectrum of label ratios.

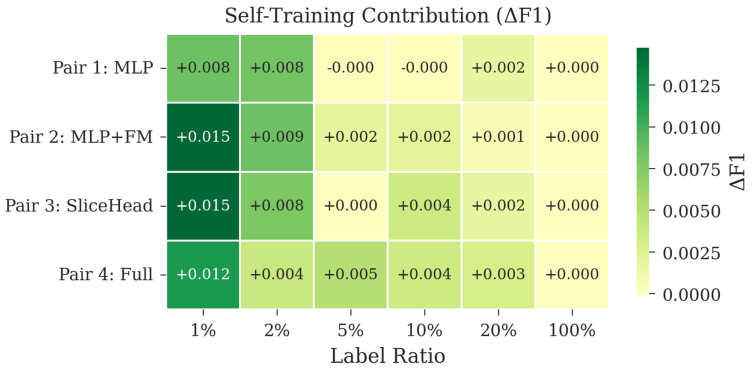

Figure 15: Heatmap of self-training contribution (ΔF1) across different configurations and label ratios.

Tracking the convergence process reveals adaptive behavior:

1. **1% Scenario:** The Full Model (seed = 42) starts with a validation F1-score of 0.9258 and reaches 0.9466 over five iterations, a total gain of +0.0208. The first iteration ($\tau = 0.95$) provides the largest single jump (+0.0126). The pseudo-label accuracy remains high, starting at 95.3% and stabilizing at 94.0%–94.3% as τ relaxes.

2. **5%–10% Scenarios:** Starting with a high initial performance (0.9716 for 5%), the model converges quickly after 1–3 iterations. With pseudo-label accuracy exceeding 98%, most valid unlabeled data is absorbed in a single pass.

These observations confirm the adaptive nature of the progressive τ-relaxation strategy. The model fully utilizes five iterations to gradually expand knowledge when labels are scarce (1%), while employing early-stopping mechanisms to prevent error accumulation through excessive iterations when labels are relatively abundant (≥5%).

6.3.3 Part C: Confidence Calibration Analysis

To quantitatively evaluate the calibration quality of the proposed framework in semi-supervised settings, this study measures the Expected Calibration Error (ECE) of 5G-SliceMatch across three label ratios, averaged over 20 random seeds. Fig. 16 presents the resulting reliability diagrams, and Table 12 summarizes the ECE values. At 1%, ECE is 0.0510 ± 0.0087, reflecting moderate overconfidence attributable to the limited initial teacher quality under extreme label scarcity. As label availability increases, ECE decreases consistently to 0.0189 ± 0.0045 at 5% and 0.0127 ± 0.0036 at 10%. This monotonic reduction confirms that the confidence threshold τ functions as a meaningful probabilistic signal rather than a heuristic filter: as the self-training loop incorporates more reliable pseudo-labels, the model's predicted probabilities become increasingly well-aligned with empirical accuracy. The low standard deviation across seeds at all ratios further indicates that this calibration behavior is stable and not an artifact of favorable data splits.

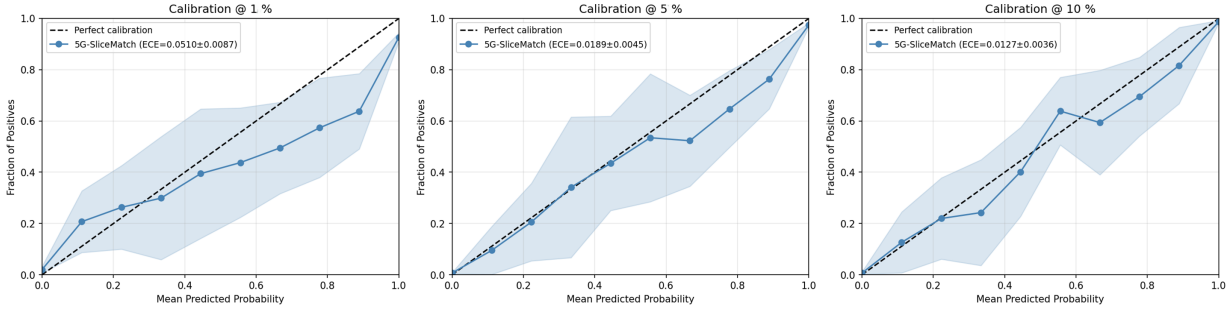

Figure 16: Confidence calibration curves of 5G-SliceMatch.

Table 12: Expected Calibration Error (ECE) across label ratios (mean ± std over 20 seeds).

Label Ratio	ECE
1%	0.0510 ± 0.0087
5%	0.0189 ± 0.0045
10%	0.0127 ± 0.0036

7 Conclusion

This study addressed the challenge of detecting malicious traffic in 5G network slicing environments under severe label scarcity. This study proposed 5G-SliceMatch, a semi-supervised learning framework integrating three core components: Slice-Aware Masking, the Slice-Aware Teacher Model, and Progressive Self-Training. Comprehensive experiments on the 5G-SliciNdd dataset demonstrate that 5G-SliceMatch consistently outperforms existing baselines in label-scarce environments. Notably, in the extreme 1% label

scenario (101 samples), the proposed method achieved an F1-Score of 0.9393, outperforming XGBoost by +0.0282 and FixMatch by +0.0946. Crucially, 5G-SliceMatch achieves 98.1% of the performance of a fully supervised XGBoost (100% labels) using only 5% of the data, effectively reducing the manual labeling workload by 95%.

These achievements confirm that the Slice-Aware Design of 5G-SliceMatch accurately captures the distinct characteristics of heterogeneous 5G slices. In the URLLC slice, where only 17 malicious samples were available, the proposed model recorded an F1-Score of 0.9190, while baselines failed to capture critical attack patterns. This stems from 5G-SliceMatch learning independent decision criteria for features with conflicting trends (e.g., Rate, TcpRtt), thereby structurally resolving the "Gradient Conflict" inherent in single-head models. Furthermore, the ablation study verified the independent efficacy of Feature Masking and Self-Training, confirming that their integration within 5G-SliceMatch generates a synergy that exceeds the simple sum of their individual contributions.

While gradient boosting models like XGBoost may prove effective in environments with abundant labels (≥20%), 5G-SliceMatch remains highly competitive across all ratios, providing superior stability in data-constrained scenarios. This study provides empirical evidence for the "Shortcut Learning" problem in 5G slice traffic and offers a domain-specific augmentation methodology to resolve it through 5G-SliceMatch. Practically, this framework overcomes the significant time and economic costs of manual labeling in large-scale 5G network operations. Moreover, the lightweight nature of the inference phase makes 5G-SliceMatch immediately applicable to real-time 5G monitoring environments where low latency is paramount.

A further limitation of the present work concerns the granularity of the classification target. The current 5G-SliceMatch framework formulates intrusion detection as a binary task (Benign vs. Malicious), aggregating the nine distinct attack types in the 5G-SliciNdd dataset (five DoS variants and four Port Scan variants) into a single malicious class. While this formulation is consistent with the operational objective of a first-line IDS at the user plane, it does not expose per-attack detection behavior. In particular, under the 1% label ratio—where only 101 labeled training samples are retained—it is statistically likely that certain rare attack types are severely underrepresented or absent in the labeled pool, which may cause the initial Teacher Model to generate confident but incorrect pseudo-labels for those unseen signatures. Although the progressive τ-relaxation schedule and the calibration behavior reported in Section 6.3.3 (monotonic ECE reduction from 0.0510 to 0.0127) jointly suppress such error amplification at the binary level, a per-attack-type performance breakdown is required to fully characterize failure modes under extreme label scarcity. This study therefore plans to extend 5G-SliceMatch to a multi-class attack-type classification setting in future work, accompanied by a stratified labeling strategy that guarantees minimum coverage of each attack category in the labeled seed set.

Another limitation concerns the adversarial robustness of the slice-routing mechanism itself. Because 5G-SliceMatch structurally routes flows to slice-specific detection heads based on slice metadata, a sophisticated adversary may attempt evasion strategies that target this routing layer. Two representative attack vectors are envisioned: a statistical mimicry attack, in which URLLC-originating malicious flows are deliberately shaped to resemble eMBB volume characteristics so that they are routed to a head optimized for a different attack profile; and a slice metadata spoofing attack, which presupposes partial compromise of the control plane and injects falsified S-NSSAI tags into the flow context, coercing malicious flows toward a mismatched head. The proposed architecture provides two built-in properties that partially mitigate these concerns—the shared encoder captures universal attack representations from Common Indicators (e.g., packet count and byte volume categories) whose directional behavior is consistent across all slices, and the fallback head described in Section 4.5 acts as a safety net for flows with inconsistent or unseen slice metadata. Nevertheless, a comprehensive defense against slice-routing-targeted evasion is beyond the scope of the present work,

and future research will extend 5G-SliceMatch with adversarial training against feature-space perturbations, cross-head consistency checks, and control-plane attestation of S-NSSAI integrity.

In future research, this study aims to extend the 5G-SliceMatch architecture from flow-level statistical analysis to packet-level time-series analysis. While flow-level summary statistics are efficient for identifying global traffic trends, they may lose fine-grained security features such as Inter-Arrival Time (IAT) or sequential packet size variations. Therefore, this study plans to integrate sequence modeling architectures, such as Transformers or LSTMs, into the 5G-SliceMatch framework to track individual packet header and payload features in real-time. Such precision analysis at the packet level will be critical for detecting sophisticated threats, such as subtle latency manipulation attacks in URLLC slices or Advanced Persistent Threats (APTs). Ultimately, the evolution of 5G-SliceMatch toward packet-level deep learning is expected to provide enhanced security visibility while satisfying the stringent QoS requirements of 5G networks.

Acknowledgement: This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT).

Funding Statement: This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2024-00438156, Development of security resilience technology based on network slicing service in the 5G specialized network).

Author Contributions: Conceptualization, Jinha Kim and Hwankuk Kim; methodology, Jinha Kim; software, Jinha Kim; validation, Jinha Kim and Hwankuk Kim; formal analysis, Jinha Kim; investigation, Jinha Kim; resources, Jinha Kim and Hwankuk Kim; data curation, Jinha Kim; writing—original draft preparation, Jinha Kim; writing—review and editing, Jinha Kim and Hwankuk Kim; visualization, Jinha Kim; supervision, Hwankuk Kim; project administration, Jinha Kim and Hwankuk Kim; funding acquisition, Hwankuk Kim. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The data that support the findings of this study are openly available in Figshare at https://doi.org/10.6084/m9.figshare.24446515.v1.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Fuentes M, Carcel JL, DIetrich C, Yu L, Garro E, Pauli V, et al. 5G new radio evaluation against IMT-2020 key performance indicators. IEEE Access. 2020;8:110880–96. doi:10.1109/access.2020.3001641.
2. Dawy Z, Saad W, Ghosh A, Andrews JG, Yaacoub E. Towards massive machine type cellular communications. IEEE Wirel Commun. 2017;24(1):120–8. doi:10.1109/mwc.2016.1500284wc.
3. Mousa M, Bahaa-ElDin AM, Sobh M. Software defined networking concepts and challenges. In: Proceedings of 2016 11th International Conference on Computer Engineering and Systems; 2016 Dec 2–21; Cairo, Egypt. p. 79–90.
4. Hohlfeld O, Kempf J, Reisslein M, Schmid S, Shah N. Guest editorial scalability issues and solutions for software defined networks. IEEE J Sel Areas Commun. 2018;36(10):2153–8. doi:10.1109/jsac.2018.2872214.
5. Mijumbi R, Serrat J, Gorricho JL, Bouten N, De Turck F, et al. Network function virtualization: state-of-the-art and research challenges. IEEE Commun Surv Tutor. 2016;18(1):236–62.
6. Yuan J, Zhang F, Yu L, Zhang H, Sang Y. Research of security of 5G-enabled industrial internet and its application. In: Proceedings of the 2021 IEEE Conference on Telecommunications, Optics and Computer Science; 2021 Dec 10–11; Shenyang, China. p. 428–35.

7. Zhang H, Chen Z, Yuan Y. High-performance UPF design based on DPDK. In: Proceedings of the 2021 International Conference on Communication Technology (ICCT); 2021 Oct 14–17; Tianjin, China. p. 349–54.
8. Chen WE, Liu CH. High-performance user plane function (UPF) for the next generation core networks. IET Netw. 2020;9(6):284–9.
9. Ye Q, Li J, Qu K, Zhuang W, Shen XS, Li X. End-to-end quality of service in 5G networks: examining the effectiveness of a network slicing framework. IEEE Veh Technol Mag. 2018;13(2):65–74.
10. An N, Kim Y, Park J, Kwon DH, Lim H. Slice management for quality of service differentiation in wireless network slicing. Sensors. 2019;19(12):2745. doi:10.3390/s19122745.
11. Pindi NR, Velez FJ. Traffic scheduling and resource allocation for heterogeneous services in 5G new radio networks: a scoping review. Electronics. 2025;14(4):789. doi:10.3390/smartcities8050168.
12. TSGS. TS 122 261-V18.19.0-5G; Service requirements for the 5G system (3GPP TS 22.261 version 18.19.0 Release 18); 2026 [cited 2026 May 1]. Available from: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3107.
13. TSGR. TR 138 913-V18.0.0-5G; Study on scenarios and requirements for next generation access technologies (3GPP TR 38.913 version 18.0.0 Release 18); 2024 [cited 2026 May 1]. Available from: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2996.
14. Kuhn L, Sadiya S, Schlotterer J, Buettner F, Seifert C, Roig G. Efficient unsupervised shortcut learning detection and mitigation in transformers. arXiv:2501.00942. 2025.
15. Hagan M, Kang B, McLaughlin K, Sezer S. Peer based tracking using multi-tuple indexing for network traffic analysis and malware detection. In: Proceedings of the 2018 16th Annual Conference on Privacy, Security and Trust (PST); 2018 Aug 28–30; Belfast, Ireland. p. 1–8.
16. TSGS. TS 123 501-V18.12.0-5G; System architecture for the 5G System (5GS) (3GPP TS 23.501 version 18.12.0 Release 18); 2026 [cited 2026 May 1]. Available from: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3144.
17. TSGS. TS 133 501-V19.5.0-5G; Security architecture and procedures for 5G System (3GPP TS 33.501 version 19.5.0 Release 19); 2026 [cited 2026 May 1]. Available from: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3169.
18. TSGS. TS 123 502-V18.12.0-5G; Procedures for the 5G System (5GS) (3GPP TS 23.502 version 18.12.0 Release 18); 2026 [cited 2026 May 1]. Available from: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3145.
19. GSMA Official Document NG. 116-generic network slice template generic network slice template copyright notice compliance notice; 2024 [cited 2026 May 1]. Available from: https://www.gsma.com/newsroom/gsma_resources/ng-116-generic-network-slice-template-v8-0/.
20. Balcan MF, Blum A. A discriminative model for semi-supervised learning. J ACM. 2010;57(3):19.
21. Amini MR, Feofanov V, Pauletto L, Hadjadj L, Devijver É, Maximov Y. Self-training: a survey. Neurocomputing. 2025;616:128904.
22. Frommknecht T, Zipf PA, Fan Q, Shvetsova N, Kuehne H. Augmentation learning for semi-supervised classification. arXiv:2208.01956. 2022.
23. Lee S, Kim H. An AI/ML framework-driven approach for malicious traffic detection in open RAN. Comput Model Eng Sci. 2025;145(2):2657–82. doi:10.32604/cmes.2025.070627.
24. Siriwardhana Y, Samarakoon S, Porambage P, Liyanage M, Chang SY, Kim J, et al. Descriptor: 5G wireless network intrusion detection dataset (5G-NIDD). IEEE Data Des. 2025;2(6):358–69. doi:10.1109/ieeedata.2025.3592888.
25. Akem ATJ, Fiore M. Towards real-time intrusion detection in P4-programmable 5G user plane functions. In: Proceedings of the 2024 IEEE 32nd International Conference on Network Protocols (ICNP); 2024 Oct 28–31; Charleroi, Belgium. p. 1–6.
26. Djaidja TET, Brik B, Mohammed Senouci S, Boualouache A, Ghamri-Doudane Y. Early network intrusion detection enabled by attention mechanisms and RNNs. IEEE Trans Inf Forensics Sec. 2024;19(5):7783–93. doi:10.1109/tifs.2024.3441862.

27. Amponis G, Radoglou-Grammatikis P, Lagkas T, Mallouli W, Cavalli A, Klonidis D, et al. Threatening the 5G core via PFCP DoS attacks: the case of blocking UAV communications. EURASIP J Wirel Commun Netw. 2022;2022:102. doi:10.21203/rs.3.rs-1708948/v1.
28. Lian X, Cao C, Liu Y, Xu X, Zheng Y, Zheng F. Facing anomalies head-on: network traffic anomaly detection via uncertainty-inspired inter-sample differences. In: Proceedings of the ACM Web Conference 2025; 2025 Apr 28–May 2; Sydney, Australia. p. 3908–17.
29. Lian X, Zheng Y, Liu Y, Zhou F, Peng C, Gao X. Contextual masking distillation for network traffic anomaly detection. IEEE Trans Inf Forensics Sec. 2026;21:1273–86. doi:10.1109/tifs.2026.3655514.
30. Jeon SE, Oh YS, Lee YJ, Lee IG. Suboptimal feature selection techniques for effective malicious traffic detection on lightweight devices. Comput Model Eng Sci. 2024;140(2):1669–87. doi:10.32604/cmes.2024.047239.
31. Goernitz N, Kloft MM, Rieck K, Brefeld U. Toward supervised anomaly detection. J Artif Intell Res. 2014;46:235–62. doi:10.1613/jair.3623.
32. Alsulami B, Almalawi A, Fahad A. Toward an efficient automatic self-augmentation labeling tool for intrusion detection based on a semi-supervised approach. Appl Sci. 2022;12(14):7246.
33. Mvula PK, Branco P, Jourdan GV, Viktor HL. A survey on the applications of semi-supervised learning to cyber-security. ACM Comput Surv. 2024;56(10):248.
34. Sohn K, Berthelot D, Li CL, Zhang Z, Carlini N, Cubuk ED, et al. FixMatch: simplifying semi-supervised learning with consistency and confidence. arXiv:2001.07685. 2020.
35. Yin C, Zhu Y, Fei J, He X. A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access. 2017;5:21954–61. doi:10.1109/access.2017.2762418.