Lightweight Sketch-Based Telemetry for QoS Anomaly Detection in 5G User Planes

Niloy Saha*, Noura Limam*, Yang Xiao*, and Raouf Boutaba*
{n6saha, noura.limam, yang.xiao, rboutaba}@uwaterloo.ca, *University of Waterloo, Canada

***Abstract*—Detecting QoS anomalies in the 5G user plane requires fine-grained visibility into per-flow performance, but existing telemetry approaches face a fundamental trade-off. 3GPP performance counters are lightweight but expose only coarse aggregates, while postcard sampling provides richer features at the cost of prohibitive overhead. We present *Kestrel*[1], a sketch-based telemetry system that compresses per-packet measurements into compact per-(TEID,QFI) summaries while retaining queue-sensitive signals such as latency distributions, inter-arrival variability, and policing color fractions. *Kestrel* introduces a target-occupancy binning rule that maximizes anomaly separability, and a data plane design that fits within the resources of a single Tofino pipeline. We implement *Kestrel* end-to-end and evaluate it on a 5G testbed with real traces and injected anomalies. Our results show that *Kestrel* achieves up to $3\times$ higher detection accuracy than 3GPP counters while reducing export overhead by $10\times$ compared to postcard sampling. It detects anomalies within sub-second delay and sustains predictable performance with modest switch resource usage, demonstrating that sketch-based telemetry is both practical and effective for next-generation mobile networks.**

I. Introduction

Modern 5G and beyond networks are increasingly expected to support *network slicing*, where a slice is a logical network dedicated to a tenant or application (e.g., smart factory, healthcare) [1]–[3]. Each slice must meet its Service Level Agreements (SLAs), often defined in terms of latency, loss, and bandwidth [4, 5]. Traffic from multiple such slices is aggregated at the user-plane function (UPF), which forwards traffic over GTP-U tunnels, identified by Tunnel Endpoint Identifiers (TEIDs). Each such TEID may, in turn, carry multiple *QoS flows*, identified by QoS Flow Identifier (QFI), that map to standardized 5G QoS profiles. These QoS flows are often enforced by a small set of hardware queues in the UPF, often shared across slices. Consequently, the UPF is a critical vantage point for detecting anomalies such as congestion, jitter, or microbursts that can violate slice SLAs.

The monitoring dilemma. Detecting anomalies requires *fine-grained, real-time telemetry*. Today's slice monitoring relies on 3GPP-defined UPF counters [6], which export per-QFI packet/byte counts, loss, and average delay every few seconds (e.g., in Open5GS [7]). These suffice for tracking throughput or loss, but are too coarse for anomaly detection: they blur per-TEID behavior when multiple TEIDs share a QFI and miss millisecond-scale bursts.

[1]Named after the kestrel falcon, noted for its efficient flight and sharp eyesight, reflecting our goal of low-overhead yet fine-grained visibility.

At the other extreme, *per-packet postcards* provide full visibility into queues, latency, and QoS identifiers. But postcards are prohibitively expensive: even a 10 Gbps UPF with 200 B packets would generate >3 Gbps of telemetry (>30% overhead), and commercial UPFs scale beyond 500 Gbps. Programmable-hardware UPFs [8, 9] can in principle support such visibility, but postcard export is impractical at scale.

This sets up the core challenge: *how to obtain per-TEID/QFI visibility without postcard-level cost?* Per-QFI counters average across TEIDs and seconds, hiding short-lived bursts. For example, in one 1s window counters may report a healthy 2.3 ms average delay, yet per-TEID histograms show TEID 0x1002 briefly spiked, with 6% of its packets delayed >20 ms. Postcards would expose this, but only at an an unsustainable cost.

Our approach. Our key insight is that anomalies manifest in the *distribution* of per-packet metrics, not coarse averages. Histograms are the right abstraction to capture these anomalies, since they preserve how latency and inter-arrival times evolve within a window rather than collapsing everything to a single mean. However, maintaining full histograms for every active TEID is infeasible at UPF scale, where tens of thousands of TEIDs may be multiplexed concurrently and each histogram requires multiple bins of state.

We present *Kestrel*, a telemetry system that bridges this gap by using *sketch-based data structures* to capture per-TEID/QFI distributions at a fraction of postcard cost. *Kestrel* maintains compact sketches within the UPF that summarize latency, inter-arrival times, color markings, and packet/byte counters for each flow, preserving essential distributional information while avoiding per-TEID state explosion. This requires addressing key design questions: how to structure sketches, configure bin sizes, and allocate memory to preserve anomaly-detectable features while optimizing data plane resource usage. The resulting summaries provide richer input to downstream AI/ML pipelines at 10× lower cost than postcards.

Contributions. This paper makes the following contributions:

- We demonstrate that existing per-QFI telemetry approaches fail to detect critical anomalies that manifest as fine-grained TEID patterns (e.g., microbursts) or as distributional shifts in metrics such as inter-arrival times (e.g., contention).
- We design and implement *Kestrel*, a sketch-based telemetry system that provides per-TEID/QFI visibility while reducing overhead by 10× compared to postcard telemetry.
- We demonstrate that *Kestrel*'s compact summaries achieve comparable or better anomaly detection accuracy than full

postcard telemetry, despite using 10× less data.
- We implement *Kestrel* using a 5G testbed and Intel Tofino programmable switches and release it as open-source to foster reproducibility.

II. Background and Motivation

We first examine the UPF architecture and related monitoring challenges (§II-A), followed by current telemetry approaches and their limitations (§II-B). We demonstrate through experiments why fine-grained visibility is essential for UPF anomaly detection (§II-C).

A. The TEID Attribution Gap in Hardware-Accelerated UPFs

The 5G User Plane Function (UPF) forwards traffic using GTP-U tunnels, which are identified by *tunnel endpoint identifiers* (TEIDs). For each such tunnel, different service classes are distinguished by QoS Flow Identifier (QFI) and enforced by *QoS enforcement rules* (QERs). Performance anomalies such as bursts, latency or jitter spikes often manifest at the granularity of individual TEIDs, making per-TEID attribution essential for maintaining slice-level SLAs.

Performance at scale comes with architectural limits. To sustain high throughput, UPFs are increasingly being offloaded to programmable hardware such as P4 switches and SmartNICs [8]–[11]. These devices deliver line-rate forwarding and expose rich per-packet metadata, but they impose rigid constraints. Programmable ASICs like Intel Tofino provide a limited number of egress queues (e.g., typically 8 on 10G ports) while even advanced SmartNICs max out at a few hundred queues. Consequently, thousands of TEIDs must be multiplexed onto a handful of shared hardware resources.

The TEID attribution gap. Once TEIDs are aggregated into shared queues, the hardware only exposes queue-level or QFI-level counters. Anomalies that affect one TEID are averaged together with well-behaved flows, masking short-lived events like microbursts and preventing attribution of session-specific misbehavior such as policy abuse. As we show in §II-C, reliable anomaly detection requires temporal, spatial, and multi-metric granularity that such aggregates cannot provide.

To investigate this problem, we implement a representative UPF on Intel Tofino. Our model uses eight queues with strict-priority and weighted round-robin scheduling, and a two-rate three-color marker (TrTCM) for shaping and policing, consistent with prior P4 UPF designs [8, 9]. This provides a realistic environment to evaluate telemetry mechanisms that can bridge the attribution gap.

B. Telemetry in Today's UPFs

UPFs today expose *3GPP performance metrics* [6], implemented in systems such as Open5GS [7], which report per-QFI packet/byte counts, loss, and average delay, typically exported every few seconds. They suffice for throughput accounting and SLA compliance, but collapse per-packet dynamics into coarse aggregates and mask TEID-level variation. As we show in §VI-A, they provide little signal for anomaly detection.

Programmable dataplanes (e.g., SmartNICs, P4 switches) enable *postcard telemetry* [12]–[14], where selected packets trigger export of compact metadata records (queue depth, timestamps, identifiers). Postcards offer full visibility, but at prohibitive cost: at 10 Gbps with 200 B packets, line-rate export exceeds 3 Gbps (>30% overhead). With carrier-grade UPFs operating at hundreds of Gbps, full postcards is infeasible. To reduce overhead, recent systems adopt change-triggered sampling (Δ-SMP) [15, 16], exporting postcards only when monitored metrics change beyond a threshold Δ. This lowers export volume, but events that remain under threshold or overlap with others, may be missed. We use Δ-SMP alongside counters as baselines in our evaluation (§VI).

C. The Need for Fine-Grained Telemetry

The attribution gap raises a fundamental question: *what monitoring granularity is needed to detect and localize anomalies in UPFs?* To explore this, we inject controlled anomalies into our Tofino-based UPF testbed carrying a mix of application traffic, including cloud gaming, live streaming, video conferencing, and IoT. Each application is mapped to standardized QFIs, with multiple TEIDs per application sharing queues.

Figure 1 shows four representative anomaly types. *Congestion*, as in sustained surges or DDoS-like events, produces long-tail latencies concentrated on specific TEIDs (Fig. 1a). *Policy abuse* arises when TEIDs exceed their QER allocations; for example, a rogue flow remapped to a higher-priority class. At the QFI level, throughput appears healthy, however, per-TEID analysis reveals throughput violations and latency spikes (Fig. 1b). *Microbursts*, common in bursty video streams, manifest as millisecond-scale spikes that disappear in one-second QFI aggregates but become visible at finer resolution (Fig. 1c). *Contention*, due to competing traffic in the shared backhaul, increases latency more broadly and induces oscillatory inter-arrival patterns distributed across flows (Fig. 1d).

While these anomalies differ in their origins and effects, they share a common characteristic: their **signatures become visible only when measured with appropriate granularity**. These experiments highlight three requirements for effective anomaly detection: (1) *sub-second temporal resolution* to expose short bursts, (2) *per-TEID visibility* to attribute misbehavior within shared queues, and (3) *distributional features* that capture latency, inter-arrival, and throughput correlations rather than simple averages.

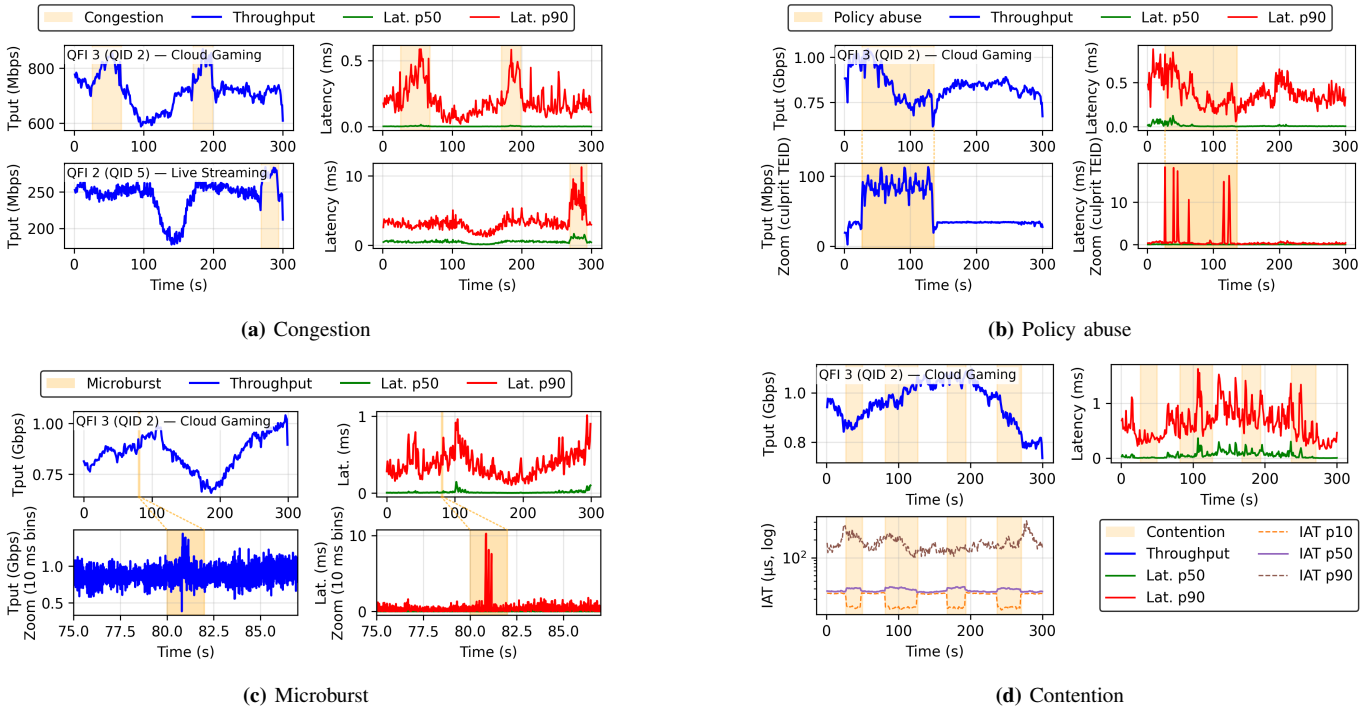
Neither counters nor postcards strike the right balance: the former masks anomalies, while the latter overwhelms the system. We need a telemetry design that preserves per-TEID distributions, key signals for QoS anomalies, while remaining lightweight enough for deployment at scale.

III. Design of *Kestrel*

A. Design Objectives

Our aim is to bridge the attribution gap (§II) by providing fine-grained visibility at a sustainable cost. Based on this, we distill three requirements for any viable telemetry primitive:

1) **Export efficiency.** Telemetry must scale to carrier-grade UPFs without overwhelming collectors or analytics pipelines. The system should bound export cost by design, rather than growing with packet rate or flow count.



(a) Congestion

(b) Policy abuse

(c) Microburst

(d) Contention

Fig. 1: Anomaly signatures require appropriate monitoring granularity. (a) Congestion produces sustained tail latency on specific QFIs. (b) Policy abuse appears benign in QFI aggregates but is visible at the culprit TEID. (c) Microbursts vanish in second-level averages yet emerge with sub-second resolution. (d) Contention induces distributed oscillations in inter-arrival times across flows.

2) **Signal fidelity.** Summaries must preserve the anomaly signatures that matter for detection, such as latency tails, inter-arrival irregularities, and QoS color fractions, while compressing away redundant detail. Preserving these signals is crucial for downstream AI/ML based anomaly detection to be effective.
3) **Practical deployability.** The telemetry primitive must be deployable on commodity programmable switches, respecting constraints on memory, ALUs, and pipeline operations. Complex features like recirculation or multi-stage coordination should be avoided for realistic deployment in UPFs.

B. Design Challenges and Solutions

Meeting the objectives in §III-A is not straightforward. One natural alternative is to optimize selective postcard sampling, for example, with adaptive thresholds or smarter triggers. But such schemes fundamentally violate Objective (1): their export cost remains traffic dependent, flooding collectors precisely during bursts when visibility is most critical. We instead pursue sketch-based summaries that bound export volume regardless of conditions. For this, we identify two main challenges:

Challenge 1: From volumes to distributions. State-of-the-art sketches such as ElasticSketch [17] and NitroSketch [18] excel at estimating flow volumes and identifying heavy hitters. They achieve high accuracy at scale, but remain tied to *volume metrics*: packet counts, byte counts, and flow sizes. What they cannot capture is how packets are distributed across latency or inter-arrival bins, the very signals that distinguish anomalies such as microbursts, contention, or policy abuse.

Kestrel addresses this gap by *extending Count–Min Sketch (CMS)* to preserve distribution signals. CMS is an attractive foundation because it offers (i) clean analytical error bounds, (ii) a simple hash-and-increment update model that maps directly to ASIC pipelines, and (iii) a lightweight footprint that leaves memory and ALU budget for essential UPF functions such as policing, queuing, and scheduling. While CMS does not minimize overestimation bias as effectively as newer sketches, its simplicity and deployability make it the right starting point.

To further mitigate bias, *Kestrel* maintains *per-QID sketches* rather than a single global structure (§III-C). Since QIDs group flows with similar QoS targets, partitioning reduces cross-class collisions and makes error bounds less pessimistic in practice. Each CMS bucket is *augmented* to hold not only packet and byte counters but also compact histograms of latency, inter-arrival times, and policing colors. These enriched buckets yield one-second summaries that preserve the distribution signals required for anomaly detection. The challenge lies not in augmenting the sketch buckets but in *configuring them correctly*: bin edges must cap diagnostic occupancy, and sketch width w and depth d must be sized to ensure anomalies remain visible despite collisions. We formalize these requirements in §IV, deriving detectability guarantees and sizing rules that make histogram-augmented sketches practical.

Challenge 2: Capturing timing without per-flow state. Objective (2) also requires preserving inter-arrival time (IAT) patterns, which are crucial for distinguishing contention from congestion. A naïve design would timestamp each active flow, requiring $O(F)$ state for F TEIDs, far beyond ASIC budgets and violating Objective (3). *Kestrel* instead timestamps

buckets, not flows. Each CMS bucket stores the arrival time of its last packet. When the next packet (possibly from a colliding flow) arrives, the IAT is computed relative to this stored timestamp and recorded into the bucket's histogram. This reduces state to $O(d \cdot w)$ rather than $O(F)$. While collisions introduce some noise, our analysis (§IV) shows that the diagnostic bins remain sparse under baseline traffic, so anomalies still stand out reliably.

Therefore, adapting sketches for UPF anomaly detection requires rethinking both the *signal model* (distribution vs. volume) and the *state model* (bucket vs. per-flow). We next present *Kestrel*'s data structure that realizes this design.

C. Data Structure and Operations

Architecture overview. *Kestrel* combines two design choices that make sketching effective in the UPF setting.

First, *per-QID partitioning*: rather than one global sketch, *Kestrel* maintains a separate sketch per QID (8 in our prototype). Packets mapped to the same queue by UPF scheduling policies typically share QoS targets (e.g., low-latency cloud gaming vs. best-effort bulk). Per-QID partitioning reduces cross-class collisions, and enables queue-specific binning (a 1 ms latency threshold is meaningful for gaming but irrelevant for best-effort). While this increases the number of sketches, each sketch can be kept small: as we show in §IV and §V, sketches with only w=512 and d=3 suffice for robust anomaly detection. This makes per-QID partitioning practical, since the total memory cost remains well within hardware budgets.

Second, *CMS with enriched buckets* to capture distributional signals. *Kestrel* maintains d rows and w buckets per row, with independent hash functions $h_1, \ldots, h_d$ over QoS flow keys $k = (\text{TEID}, \text{QFI})$. As shown in Figure 2, each bucket $j_i = h_i(k)$ stores four components: (a) *core counters* for packets and bytes, (b) *meter colors* tallying TrTCM outcomes (green, yellow, red), (c) a *latency histogram* updated from per-packet sojourn time, and (d) an *IAT histogram* updated from the bucket's timestamp register. These enriched buckets allow *Kestrel* to compactly summarize traffic distributions.

Packet updates. On each arriving packet with key $k = (\text{TEID}, \text{QFI})$ and latency t_s, *Kestrel* computes the bucket indices $j_i = h_i(k)$ for each row i. For each corresponding bucket (i, j_i), it then: 1) increments the packet and byte counters, 2) maps the latency t_s to a bin using QID-specific edges, 3) computes the Inter-Arrival Time (IAT) as the difference from the bucket's last-seen timestamp, updates the corresponding IAT bin, and overwrites the timestamp, and 4) increments the color counter corresponding to the packet's TrTCM marking.

Export and query model. At the end of each 1 s window, the control plane collects one compact record per bucket. To estimate the feature vector for a specific flow with key k, it queries all d buckets $h_i(k)$ to which the flow was mapped. The estimate for each feature (e.g., the count in latency bin #3) is the minimum across the d candidate buckets, following CMS query semantics. The control plane, which already maintains TEID–QFI mappings for bearer/session management, uses these mappings to reconstruct per-flow distributions suitable for ML-based anomaly detection. This model keeps the switch pipeline lightweight while ensuring anomaly-relevant signals are exposed at second-level cadence.

Illustrative Example (Figure 2). Consider a GTP-U packet arriving at the UPF with TEID=87 and QFI=2, experiencing a sojourn time of 25 μs and an inter-arrival gap of 18 μs. The UPF maps this flow to QID=3 according to its scheduling policies. *Kestrel* processes this packet as follows: First, it forms the sketch key k from TEID+QFI and applies d hash functions to map k to buckets (i, j_i) across all rows. Second, it retrieves the QID=3-specific bin edges; for latency ($\{0.5, 6.3, 82, \ldots, 4970\}$ μs) and IAT ($\{11.5, 16.2, 22.9, \ldots, 2.7{\times}10^6\}$ μs). Finally, for each bucket (i, j_i), it: increments packet/byte counters; assigns the 25 μs latency to bin 2 (as shown in Fig. 2); computes and records the 18 μs IAT (updating bin 2 and refreshing the timestamp); and updates the color counter based on meters.

IV. Parameterizing Kestrel

Kestrel's design raises three practical questions: (i) how to ensure detectability when anomalies only occupy a small region of the distribution, (ii) how to size the sketch parameters to meet detection targets, and (iii) how to place bin boundaries and handle distribution drift. We address each in turn.

A. Detectability Guarantees

We first establish when anomalies become visible to Kestrel. The key insight is that anomalies usually concentrate in specific distribution regions: we designate the latency tail and IAT head bins as *diagnostic region* T. The diagnostic region T is deliberately sparse (N_T packets in normal traffic), making any anomaly-induced increase Δ_T readily detectable.

However, not all anomaly packets land in T; some spill over to other bins. Let β be this spillover fraction ($\beta = 0$ means all anomaly packets fall in T). The sketch itself introduces collision noise εN_T, where $\varepsilon \approx e/w$ is the CMS error rate. Our detectability theorem accounts for both effects:

Theorem IV.1 (Detectability with spillover)**.** *For a flow k, suppose an anomaly adds Δ_T packets into its diagnostic bins T and $\beta\Delta_T$ outside T. Then, with probability at least $1-\delta$, the anomaly is detectable provided*

$$\Delta_T \;>\; \frac{\varepsilon N_T x_k + (x_{k,T} + \varepsilon N_T)\,\varepsilon N'}{x_{k,\bar{T}} - \varepsilon N_T - \beta(x_{k,T} + \varepsilon N_T)},$$

where x_k is the total mass of k in the window, $x_{k,T}$ the baseline mass of k in T, $x_{k,\bar{T}}$ the baseline mass outside T, and N' the queue's total mass in the window.

Proofs appear in Appendix A.

Intuition and example. Detection succeeds when the anomaly lift Δ_T in the diagnostic region T exceeds the CMS collision floor εN_T (plus small terms) *after accounting for spillover β outside T*. Consider $N = 10^6$ pkts/s. With diagnostic cap $\rho = 1\%$, at most $N_T = 10^4$ packets occupy T during baseline. At $w = 512$, $\varepsilon \approx 5.3 \times 10^{-3}$, so $\varepsilon N_T \approx 53$ packets. With 30% spillover ($\beta = 0.3$, i.e., 70% of anomaly mass lands in T), Theorem IV.1 requires $\Delta_T \gtrsim 76$ packets. A 50 ms microburst at 50 kpps injects ∼2,500 packets; with 70% in T,

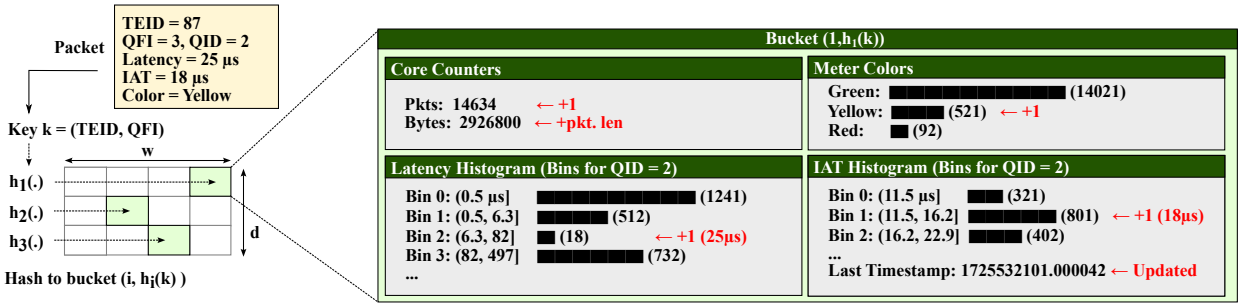


Fig. 2: Data structure and operations of *Kestrel*. Buckets extend CMS to record packet/byte totals, latency histograms, IAT histograms, and policer colors. Example shows an arriving packet (TEID=87, QFI=3, QID=2, latency=25 µs, IAT=18 µs, color=yellow) being hashed to a bucket: totals and color counters are incremented, latency/IAT bins are updated using QID-specific edges, and the last is timestamp refreshed (updates in red).

$\Delta_T \approx 0.7 \times 2{,}500 = 1{,}750\,(> 23 \times \text{threshold})$, so it is readily detectable.

B. Choosing Width and Depth

The detectability condition from Theorem IV.1 translates directly into concrete sizing rules for the sketch parameters w (width) and d (depth).

Theorem IV.2 (Width requirement)**.** *To detect all anomalies injecting at least $\Delta_T^{\min}$ packets into the diagnostic region, the sketch width must satisfy: $w \geq \frac{e\, N_T^{\max}}{(1-\beta_{\max})\,\Delta_T^{\min}}$, where $N_T^{\max}$ is the maximum diagnostic occupancy enforced by binning and $\beta_{\max}$ bounds the spillover fraction.*

Example. With $N_T^{\max} = 10^4$ and $\beta_{\max} = 0.3$, detecting anomalies that inject as few as 80 packets into T requires $w \geq 485$. Setting $w = 512$ provides a safety margin while remaining practical.

Theorem IV.3 (Depth requirement)**.** *Let $K = |T|$ be the number of diagnostic bins. If $d \geq \lceil \ln((K+1)/\zeta) \rceil$, then with probability at least $1-\zeta$ all CMS bounds needed for detection hold simultaneously in a window.*

In practice, K is small (typically 2–3 bins for latency tails and IAT heads). With $d = 3$, the probability that all bounds hold exceeds 99% per window. Since most anomalies persist across multiple windows, the effective miss probability becomes negligible over their duration.

C. Binning and Drift

Theorem IV.2 shows that detectability depends on both sketch width w and the background occupancy $N_T^{\max}$ of the diagnostic region. Since $N_T^{\max}$ is determined by the placement of the bin, careful binning is crucial: if too much baseline traffic lands in T, the collision noise εN_T increases, requiring larger widths for the same detection guarantee.

Target-occupancy binning. *Kestrel* uses *target-occupancy binning* to control $N_T^{\max}$: bin edges are placed to cap baseline occupancy in T at a target fraction ρ (typically 1–2% of total packets). Specifically, placing the latency-tail boundary at the $(1-\rho)$-quantile and the IAT-head boundary at the ρ-quantile ensures $N_T^{\max} = \rho N$ (see Appendix for derivation). This approach maximizes anomaly separability by making diagnostic bins nearly empty under normal conditions—as we validate empirically in §VI, target-occupancy outperforms alternative binning strategies by systematically improving detection accuracy.

Handling drift. Traffic distributions naturally drift over time, potentially raising N_T above ρN. Left uncorrected, this degradation scales the required width proportionally to the occupancy increase: if N_T doubles, the needed width also doubles. To maintain detection sensitivity, the control plane continuously monitors diagnostic occupancy from exported sketches and triggers *re-binning* when $\hat{N}_T/N > \rho$. Our P4 implementation supports this dynamically: bin edges are stored as runtime-programmable table entries keyed by latency range and queue ID, allowing the control plane to refresh boundaries via P4Runtime without pipeline disruption.

Key takeaways. Our analysis yields three practical configuration rules: (i) reserve a small diagnostic region (2–3 bins for latency tails and IAT heads), (ii) cap its baseline occupancy using target-occupancy binning ($\rho \leq 2\%$) with drift adaptation, and (iii) size sketches at $w \approx 512$, $d = 3$–4 under typical queue loads. These settings ensure $\Delta_T \gg \varepsilon N_T$ for anomalies of practical strength, providing robust detection at modest memory cost. With these parameters in place, *Kestrel* exports compact but anomaly-revealing summaries, lightweight enough to deploy in real UPFs. We next show how this design is realized on programmable hardware.

V. Implementation

Testbed and Dataplane. We prototype *Kestrel* on a UfiSpace S9180-32X switch equipped with an Intel Tofino ASIC. The switch implements UPF functions including queuing, scheduling, and TrTCM metering, and connects over 10 GbE NICs to four Ubuntu 22.04 servers. These servers act as: (i) a GTP-U traffic generator that allows fine-grained control over TEID/QFI mappings, (ii) a control plane host that programs P4 tables and configures runtime telemetry, (iii) a collector for postcards and sketch registers, and (iv) an ML pipeline server for anomaly detection.

The dataplane implementation consists of ∼2K lines of P4 in the egress pipeline. The design supports all three telemetry modes: 3GPP-PM counters, postcards (via packet mirroring), and sketches. It fits within a single pipeline pass (∼7 stages) without recirculation. Sketches use d=3 rows and w=512 buckets (§IV); each bucket stores counters, 8-bin latency and

IAT histograms, and TrTCM color tallies. Per-QID bin tables are runtime-programmable, enabling dynamic reconfiguration. Queues implement strict-priority and WRR scheduling across QIDs to emulate realistic UPF service classes.

TABLE I: Anomalies injected in our testbed.

Scen.	Anomaly	Target	Dur.	Freq.
1	Microburst	One/more TEIDs	0.3–1.0 s	10–30 s
2	Congestion	One/more QFIs	25–40 s	60–120 s
3	Contention	Many QFIs/TEIDs	12–30 s	90–150 s
4	Policy abuse	Selected TEIDs	30–60 s	150–300 s
5	Mixed	Multiple	As above	As above

Workloads and Anomalies. Traffic workloads combine a 5G operator trace [19] (cloud gaming, live streaming, buffered streaming) with additional traffic types we collected for IoT, VoIP, and best-effort applications. The generator sustains 1–4 Gbps aggregate load across up to 100 UEs × 9 QFIs (≈900 flows)[2]. We inject four representative anomaly scenarios: *microbursts* (short high-rate bursts), *congestion* (sustained overload), *contention* (shared backhaul bottlenecks), and *policy abuse* (QFI remapping), as well as mixed cases. Table I summarizes their scope, durations, and injection frequencies. Each anomaly ultimately degrades QoS for affected UEs.

Telemetry export and processing. Sketches are exported once per second via the Barefoot gRPC runtime (∼6 Mbps), while postcards are captured by a dedicated collector using AF_PACKET v3 with a memory-mapped ring buffer for high-throughput, zero-copy capture. All telemetry modes are aligned to one-second windows for fair comparison. Feature extraction proceeds according to the intrinsic visibility of each mode. Sketches yield per-(TEID,QFI) distributions by querying the d rows for each key and applying CMS query semantics. Postcards provide per-(TEID,QFI) distributions directly from INT metadata (latency, IAT, color). Counters expose only QFI-level aggregates. From these, we construct feature vectors including traffic volume, latency percentiles and tail fractions, inter-arrival head fractions, policer color ratios, and contextual metrics such as TEID counts per QFI.

Anomaly detection pipeline. [yang: We implement an anomaly detection pipeline incorporating four dedicated detectors that target congestion, contention, microbursts, and policy abuse. Each detector is implemented as an XGBoost model trained on baseline and anomaly-injected traces, enabling robust classification between normal and abnormal traffic patterns. As a result, each detector is allowed to focus on anomaly-specific behaviors. For instance, the microburst detector leverages sub-second temporal spikes, while the contention detector uses distributional shifts across flows.]

VI. Evaluation

We evaluate *Kestrel* on our testbed along three axes: accuracy across anomaly types, telemetry cost, and responsiveness. We also report microbenchmarks to assess practicality.

Baselines. We evaluate three telemetry approaches: (i) **3GPP-PM**, the standards-defined per-QFI performance measurements [6, 7], which expose per-QFI packet and byte counts, average delay, and loss; (ii) Δ-*SMP*, selective postcard sampling that exports telemetry only when monitored metrics change beyond a configured Δ threshold [15, 16], thereby reducing postcard volume relative to per-packet export; and (iii) **Kestrel**, our sketch-based system that maintains per-TEID/QFI summaries of latency, inter-arrival times, and bandwidth.

[2]Our prototype generator sustains a few Gbps; higher loads are feasible with a DPDK-based design, which we leave to future work.

TABLE II: Telemetry baselines

Baseline	Switch	Collector	Overhead
3GPP-PM	Per-QFI counters	Poll every 1 s, construct per-QFI features	***Low***, $O(\#\text{QFIs})$
Kestrel	Per-QID sketches	Query registers every 1 s, reconstruct per-(TEID,QFI)	***Medium***, $O(wd)$
Δ-SMP	Postcard sampling	Collect samples, 1 s windowing into per-(TEID,QFI)	***High***, $O(\#\text{pkts})$

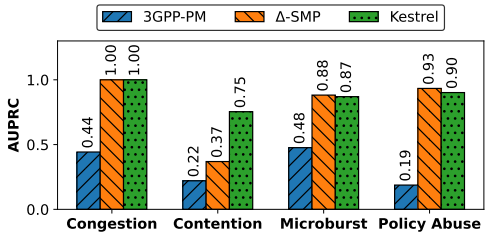
Table II summarizes the characteristics of these three telemetry approaches. It highlights their respective switch operations, collection mechanisms, exported features, and asymptotic overhead. These baselines capture the main trade-offs: (i) counters are coarse but widely deployed, (ii) postcards provide fine-grained visibility but at prohibitive overhead, and (iii) *Kestrel*'s sketches aim to balance accuracy and efficiency. Our evaluation proceeds in four parts: anomaly detection accuracy (§VI-A), cost-effectiveness (§VI-B), responsiveness (§VI-C, and microbenchmarks (§VI-D).

A. Anomaly Detection Accuracy

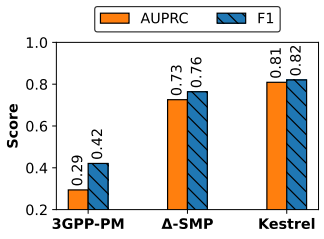
Single anomaly scenarios. We first evaluate scenarios containing one anomaly type at a time (Scenarios 1–4). Figure 3a shows the area under the precision–recall curve (AUPRC) values revealing a fundamental insight: *detection fidelity is constrained by telemetry granularity*. 3GPP-PM counters, limited to per-QFI aggregates, consistently underperform with AUPRC as low as 0.19 (policy abuse) and rarely exceeding 0.5. Both Δ-SMP and Kestrel achieve higher accuracy by exposing distributional features, but differ in coverage: Δ-SMP exports only when metrics exceed thresholds, while Kestrel summarizes every window.

Kestrel consistently matches or surpasses Δ-SMP across scenarios. For congestion and policy abuse, *Kestrel* and Δ-SMP both achieve near-perfect accuracy (AUPRC 1.0 and 0.9, respectively), vastly outperforming counters. For microbursts, *Kestrel* achieves 0.87, slightly below Δ-SMP (0.88) but still more than 80% higher than counters (0.48). Contention is the most telling case: counters collapse to AUPRC 0.22, Δ-SMP improves to 0.37, while *Kestrel* achieves 0.75; over 3× higher than counters and more than 100% better than Δ-SMP. Here, the advantage of continuous *Kestrel* over selective sampling is clearest. Contention manifests as small but widespread distortions across many flows rather than sharp per-flow spikes; Δ-SMP, which only exports when a single metric exceeds a threshold, systematically under-exports during such conditions.

Mixed anomalies. We next consider Scenario 5, where multiple anomaly types may overlap in the same window. This is a strictly harder setting: signatures can dilute one another, and



(a) Single anomaly type (Scenarios 1–4).



(b) Mixed anomalies (Scenario 5).

Fig. 3: Detection accuracy across telemetry approaches. (a) Single-anomaly scenarios show Kestrel matching or exceeding alternatives; (b) Mixed scenarios demonstrate *Kestrel*'s robustness.

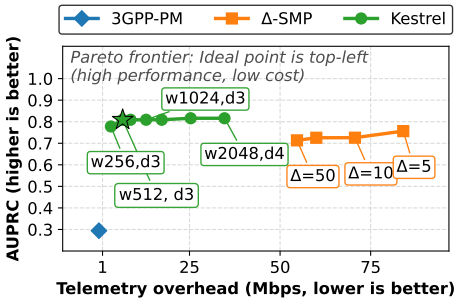


Fig. 4: Pareto of AUPRC vs telemetry cost. Kestrel's best (w=512, d=3) reaches AUPRC 0.81 at ∼6 Mbps—3× higher than 3GPP-PM and 10× cheaper than Δ-SMP.

overlap increases the likelihood of false positives. Figure 3b shows that counters collapse (AUPRC 0.29, F1 0.42), making them unusable for detecting overlapping anomalies. Δ-SMP remains reasonably strong (0.73/0.76), but Kestrel surpasses it with 0.81/0.82, a **10% improvement in AUPRC** and more than **2.5× higher than counters**. The advantage stems from *Kestrel* capturing the full distribution in every window, while Δ-SMP may miss sub-threshold events when anomalies coincide.

These results demonstrate that counters offer little diagnostic value for anomaly detection, while *Kestrel* delivers accuracy on par with or better than Δ-SMP across all scenarios.

B. Cost-Effectiveness

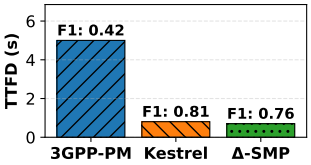
We next turn from accuracy to cost, quantifying the telemetry overhead required to achieve the results above. Figure 4 plots the accuracy-cost trade-off, with ideal operating points in the top-left region. 3GPP-PM occupies the lower-left: minimal overhead (∼1 Mbps) but unusable accuracy (AUPRC ∼0.3). Δ-SMP achieves moderate accuracy (0.7–0.76) at high cost (60–80 Mbps), substantially more expensive than *Kestrel* for inferior performance. *Kestrel* defines the Pareto frontier, achieving AUPRC 0.75–0.82 with 5–10 Mbps overhead. The sweet spot at $w = 512, d = 3$ delivers AUPRC 0.81 which is **3× higher than 3GPP-PM and 10× cheaper than Δ-SMP**. Larger configurations show diminishing returns, confirming our parameter analysis from §IV.

At the tested loads of 1–4 Gbps (§V), *Kestrel* exports ∼6 Mbps (0.15–0.6% of throughput). Its export volume is primarily determined by the sketch configuration (number of QIDs, w, d, and bins) rather than instantaneous packet rate; it remains stable for a fixed configuration. At higher loads or with denser traffic in diagnostic regions, operators may provision modestly larger w (cf. §IV-B), but this scaling is explicit and operator-controlled.

In contrast, Δ-SMP is event-driven: export rises with the frequency and severity of metric excursions. While not strictly linear in rate, postcard volume grows with burstiness and anomaly intensity and can spike during busy periods. In our tests, Δ-SMP incurs 60–80 Mbps (1.5–8%), an order of magnitude above Kestrel despite lower accuracy. The bounded, configuration-driven nature of *Kestrel*'s cost makes it predictable and easier to provision at carrier scale.

C. Responsiveness

We next evaluate responsiveness, measured as the *time-to-first-detection* (TTFD). At the tuned thresholds from our earlier F1 analysis, we record the median delay between anomaly onset and the first alarm. Figure 5 (left) summarizes results across schemes, with F1 scores indicated above each bar. Kestrel and Δ-SMP both detect anomalies within roughly one second, while 3GPP-PM counters require several seconds and achieve much lower F1. Figure 5 (right) breaks down results by anomaly type.



Anomaly	3GPP-PM	Δ-SMP	Kest.
Congestion	6.00	1.20	1.05
Contention	6.00	0.20	1.05
Microburst	0.50	0.20	0.05
Policy abuse	4.00	1.20	0.55

Fig. 5: Median time to first detection (TTFD). Right: bar chart across schemes with F1 scores indicated above bars. Left: breakdown per anomaly type and scheme.

For microbursts, Kestrel detects essentially immediately (0.05 s median) while sustaining high accuracy (F1 0.92). Δ-SMP is also fast (0.20 s) but yields lower precision, reflecting missed or partial bursts when export is not triggered. Counters detect only the largest bursts (0.50 s) and otherwise miss episodes. For congestion and contention, Δ-SMP produces short delays (0.20–1.20 s), but *Kestrel* achieves comparable responsiveness (0.20–1.05 s) with consistently higher F1. Policy abuse is detected within sub-second delay by sketches (0.55 s) compared to several seconds for counters (4.00 s).

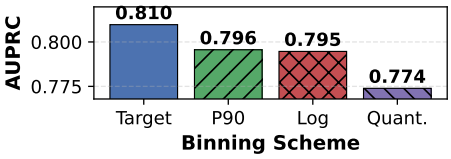
These results show that *Kestrel* maintains responsiveness on par with postcards, while counters respond too slowly to be useful for timely anomaly detection.

D. Microbenchmarks and Design Insights

We complement the accuracy and cost evaluation with microbenchmarks that validate *Kestrel*'s practicality. Figure 6 summarizes three aspects: (a) per-window processing latency, (b) sensitivity to binning strategies, and (c) hardware resource usage on Tofino.

Pipeline latency. We measure end-to-end processing per 1 s window, broken down into ingestion (I), feature extraction (F),

Scheme	p50 (ms)	IQR (± ms)	I/F/ML (%)
3GPP-PM	1.83	0.04	37/29/33
Kestrel	53.3	0.45	27/72/1
Δ-SMP	200.4	7.3	33/66/1



Resource	Usage
Memory (SRAM/TCAM)	<20%
Hashing (Hash Units)	≈25%
Computation (ALU)	≈35%

Fig. 6: Microbenchmarks of *Kestrel*. (a) Per-window pipeline latency, showing bounded cost. (b) Accuracy across binning strategies, target-occupancy performs best. (c) Switch resource footprint, showing modest usage.

and ML detection (ML). Table 6(a) shows that all methods finish well within the deadline: counters in ∼2 ms, Kestrel in 53 ms, and Δ-SMP in 200 ms (median). Counters are trivially cheap but uninformative. *Kestrel* spends most of its budget (72%) on feature extraction, as sketch registers must be queried and aggregated, yet the absolute cost remains modest: tens of ms per window, and independent of packet rate or number of flows. By contrast, Δ-SMP spends two-thirds of its budget on feature extraction because each sampled packet is processed individually, causing latency to grow with traffic volume. *Kestrel* also exhibits low variance (IQR ±0.45 ms), confirming predictable performance. Therefore, it strikes a practical middle ground: richer than counters, yet an order of magnitude faster and more stable than postcard sampling.

Binning strategies. A key design choice is how to discretize latency and IAT into bins. Our *target-occupancy* rule places latency-tail and IAT-head boundaries at quantiles that ensure at most ρN baseline samples fall into diagnostic bins (§IV-C). This keeps extreme bins nearly empty under normal conditions, making anomalies produce clear spikes.

We compare against three alternatives (Figure 6(b)): logarithmic spacing (*Log*), fixed 90th percentile thresholds (*P90*), and equal-mass binning (*Quantile*). These alternatives have reasonable motivations: Log accounts for heavy-tailed distributions, P90 captures most normal behavior while isolating outliers, and Quantile ensures balanced coverage. Target-occupancy achieves the highest AUPRC (0.810) versus P90 (0.796), Log (0.795), and Quantile (0.774). We find consistent gains of 1–4% across all QIDs and datasets. The results are statistically significant, showing standard deviations below 0.5% over five independent experiments. In practice, these improvements reduce false alarms at high recall operating points (e.g., target-occupancy lowers contention false positives by ∼20% at 90% recall compared to quantile binning).

Hardware usage. Figure 6(c) shows Kestrel's resource footprint on Tofino. The design consumes less than 20% of SRAM/TCAM, about 25% of hash units, and 35% of ALUs, leaving substantial headroom for concurrent functions. These results demonstrate that Kestrel fits comfortably within the constraints of a production programmable switch.

VII. Related Work

Telemetry paradigms. *In-band telemetry (INT)* encodes metadata directly in packets [15, 20]–[23]. While widely used in datacenters, traditional INT is challenging to deploy in mobile networks due to GTP-U encapsulation, middlebox handling, and lack of end-to-end programmability. *Event-driven approaches* report telemetry only when certain predicates are met. For example, Marple [24], BurstRadar [25], and NetSeer [26] trigger telemetry on queue thresholds or on drops, reducing cost but limiting visibility to predefined events. *Sketch-based approaches* maintain compact summaries [17, 18, 27], and have been previously applied to scenarios such as heavy hitter and volumetric attack detection. However, prior works, typically do not focus on *per-flow distributions* of latency, IAT, or policing colors, features that are critical for QoS anomaly detection at the UPF. Kestrel addresses precisely this gap.

Anomaly detection. Control-plane anomaly detection [28]–[30] addresses abnormal behavior in 5G core network functions (NFs) introduced by the Service-Based Architecture (SBA). Approaches include deep sequence models to identify abnormal NF interactions [28], AI/ML for detecting anomalous traffic events [29], and detection of service degradation in cloudified NF deployments [30]. In the RAN context, SpotLight [31] applies distributed generative models across edge and cloud to detect performance anomalies, while other studies survey signaling storms and their mitigation [32]. Slice-level anomaly detection [33, 34] addresses security and QoS threats at the granularity of network slices. For example, [33] applies deep learning to detect DDoS attacks targeting slices and to dynamically isolate attackers, while [34] proposes a graph-based framework that correlates multivariate slice KPIs to proactively detect and explain anomalies. These approaches operate either on packet captures, incurring very high cost, or on aggregate counters and KPIs which cannot capture transient per-TEID QoS violations within a slice.

VIII. Conclusion

We presented *Kestrel*, a sketch-based telemetry system for detecting QoS anomalies in the 5G user plane. By summarizing per-packet behavior into compact per-QFI histograms, *Kestrel* retains the signals needed for anomaly detection while keeping export overhead low. Our evaluation on a Tofino testbed shows that *Kestrel* achieves high detection accuracy, sub-second responsiveness, and predictable performance, all within modest hardware resource budgets. These results demonstrate that sketch-based telemetry, when carefully designed, is both practical and effective for next-generation mobile networks.

While this paper focused on hardware-accelerated UPFs, the same sketching principles could extend to software dataplanes (e.g., eBPF or VPP), offering lightweight, anomaly-aware telemetry across diverse UPF deployments.

References

[1] NGMN Alliance. (2016) Description of network slicing concept. [Online]. Available: https://ngmn.org/wp-content/uploads/160113_NGMN_Network_Slicing_v1_0.pdf

[2] 3GPP, "System architecture for the 5g system; stage 2," 3GPP, Technical Specification (TS) 23.501, 09 2020, version 16.5.1.

[3] ——, "Service requirements for the 5g system," 3GPP, Technical Specification (TS) 22.261, 07 2024, version 18.14.0.

[4] M. Cohn. (2023) The case for private network slas: A real-world view. [Online]. Available: https://www.spirent.com/blogs/the-case-for-private-network-slas-a-real-world-view

[5] K. Qin and M. Zarri, "Network slicing usecase requirements," GSMA, White Paper, 2018, https://www.gsma.com/solutions-and-impact/technologies/networks/wp-content/uploads/2018/07/Network-Slicing-Use-Case-Requirements-fixed.pdf.

[6] 3GPP, "Management and orchestration; 5G performance measurements," 3GPP, Technical Specification (TS) 28.552, 09 2020, version 17.0.0.

[7] Open5GS. (2024) Open5GS github. [Online]. Available: https://github.com/open5gs/open5gs

[8] R. MacDavid, C. Cascone, P. Lin, B. Padmanabhan, A. ThakuR, L. Peterson, J. Rexford, and O. Sunay, "A p4-based 5g user plane function," in *Proceedings of the ACM SIGCOMM Symposium on SDN Research (SOSR)*, ser. SOSR '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 162–168. [Online]. Available: https://doi.org/10.1145/3482898.3483358

[9] Z. Wen and G. Yan, "HiP4-UPF: Towards High-Performance comprehensive 5g user plane function on p4 programmable switches," in *2024 USENIX Annual Technical Conference (USENIX ATC 24)*. Santa Clara, CA: USENIX Association, Jul. 2024, pp. 303–320. [Online]. Available: https://www.usenix.org/conference/atc24/presentation/wen

[10] S. K. Singh, C. E. Rothenberg, J. Langlet, A. Kassler, P. Vörös, S. Laki, and G. Pongrácz, "Hybrid p4 programmable pipelines for 5g gnodeb and user plane functions," *IEEE Transactions on Mobile Computing*, vol. 22, no. 12, pp. 6921–6937, 2023.

[11] M. Rouili and R. Boutaba, "Blink: A p4-based 5g centralized unit," in *IEEE/IFIP Network Operations and Management Symposium (NOMS)*, 2024, pp. 1–9.

[12] N. Handigol, B. Heller, V. Jeyakumar, D. Mazières, and N. McKeown, "I know what your packet did last hop: Using packet histories to troubleshoot networks," in *Proceedings of the USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*, 2014, pp. 71–85.

[13] ONF, "In-band network telemetry (int)," https://docs.sd-fabric.org/sdfabric-1.0/advanced/int.html, 2022, accessed: 2025-06-07.

[14] Intel Corporation, "In-band network telemetry detects network performance issues," Intel, Tech. Rep., 2020, https://builders.intel.com/docs/networkbuilders/in-band-network-telemetry-detects-network-performance-issues.pdf.

[15] S. Sheng, Q. Huang, and P. P. C. Lee, "DeltaINT: Toward General In-band Network Telemetry with Extremely Low Bandwidth Overhead," in *Proceedings of the 29th IEEE International Conference on Network Protocols (ICNP)*, Nov. 2021, pp. 1–11.

[16] S. R. Chowdhury, R. Boutaba, and J. François, "Lint: Accuracy-adaptive and lightweight in-band network telemetry," in *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2021, pp. 349–357.

[17] T. Yang, J. Jiang, P. Liu, Q. Huang, J. Gong, Y. Zhou, R. Miao, X. Li, and S. Uhlig, "Elastic sketch: adaptive and fast network-wide measurements," in *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, ser. SIGCOMM '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 561–575. [Online]. Available: https://doi.org/10.1145/3230543.3230544

[18] Z. Liu, R. Ben-Basat, G. Einziger, Y. Kassner, V. Braverman, R. Friedman, and V. Sekar, "Nitrosketch: robust and general sketch-based monitoring in software switches," in *Proceedings of the ACM Special Interest Group on Data Communication*, ser. SIGCOMM '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 334–350. [Online]. Available: https://doi.org/10.1145/3341302.3342076

[19] Y.-H. Choi, D. Kim, and M. Ko. (2023) 5g traffic datasets. [Online]. Available: https://dx.doi.org/10.21227/ewhk-n061

[20] The P4.org Application Working Group, "In-band Network Telemetry (INT) Dataplane Specification," 2024, https://p4.org/p4-spec/docs/INT_v2_1.pdf.

[21] Y. Li, R. Miao, H. H. Liu, Y. Zhuang, F. Feng, L. Tang, Z. Cao, M. Zhang, F. Kelly, M. Alizadeh, and M. Yu, "Hpcc: high precision congestion control," in *Proceedings of the ACM Special Interest Group on Data Communication*, ser. SIGCOMM '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 44–58. [Online]. Available: https://doi.org/10.1145/3341302.3342085

[22] R. Ben Basat, S. Ramanathan, Y. Li, G. Antichi, M. Yu, and M. Mitzenmacher, "Pint: Probabilistic in-band network telemetry," in *Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication*, ser. SIGCOMM '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 662–680. [Online]. Available: https://doi.org/10.1145/3387514.3405894

[23] S. Tang, S. Zhao, X. Pan, and Z. Zhu, "How to Use In-Band Network Telemetry Wisely: Network-Wise Orchestration of Sel-INT," *IEEE/ACM Transactions on Networking*, pp. 1–15, 2022.

[24] S. Narayana, A. Sivaraman, V. Nathan, P. Goyal, V. Arun, M. Alizadeh, V. Jeyakumar, and C. Kim, "Language-directed hardware design for network performance monitoring," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, ser. SIGCOMM '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 85–98. [Online]. Available: https://doi.org/10.1145/3098822.3098829

[25] R. Joshi, T. Qu, M. C. Chan, B. Leong, and B. T. Loo, "Burstradar: Practical real-time microburst monitoring for datacenter networks," in *Proceedings of the 9th Asia-Pacific Workshop on Systems*, ser. APSys '18. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: https://doi.org/10.1145/3265723.3265731

[26] Y. Zhou, C. Sun, H. H. Liu, R. Miao, S. Bai, B. Li, Z. Zheng, L. Zhu, Z. Shen, Y. Xi, P. Zhang, D. Cai, M. Zhang, and M. Xu, "Flow event telemetry on programmable data plane," in *Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication*, ser. SIGCOMM '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 76–89. [Online]. Available: https://doi.org/10.1145/3387514.3406214

[27] M. Yu, L. Jose, and R. Miao, "Software defined traffic measurement with opensketch," in *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation*, ser. nsdi'13. USA: USENIX Association, 2013, p. 29–42.

[28] Y. Tan, J. Liu, Y. Li, and J. Wang, "Deep learning based proactive anomaly detection for 5g core control plane network function interactions," *IEEE Transactions on Cognitive Communications and Networking*, pp. 1–1, 2025.

[29] A. Mekrache, K. Boutiba, and A. Ksentini, "Combining network data analytics function and machine learning for abnormal traffic detection in beyond 5g," in *Proceedings of the IEEE Global Communications Conference*, 2023, pp. 1204–1209.

[30] F. Michelinakis, J. S. Pujol-Roig, S. Malacarne, M. Xie, T. Dreibholz, S. Majumdar, W. Y. Poe, G. Patounas, C. Guerrero, A. Elmokashfi, and V. Theodorou, "Ai anomaly detection for cloudified mobile core architectures," *IEEE Transactions on Network and Service Management*, vol. 20, no. 2, pp. 1976–1992, 2023.

[31] C. Sun, U. Pawar, M. Khoja, X. Foukas, M. Marina, and B. Radunovic, "Spotlight: Accurate, explainable and efficient anomaly detection for open ran," in *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*. ACM, Nov. 2024, the 30th Annual International Conference On Mobile Computing And Networking, MobiCom 2024 ; Conference date: 18-11-2024 Through 22-11-2024.

[32] A. Tabiban, H. A. Alameddine, M. A. Salahuddin, and R. Boutaba, "Signaling storm in o-ran: Challenges and research opportunities," *IEEE Communications Magazine*, vol. 62, no. 6, pp. 58–64, 2024.

[33] B. Bousalem, V. F. Silva, R. Langar, and S. Cherrier, "Ddos attacks detection and mitigation in 5g and beyond networks: A deep learning-based approach," in *Proceedings of the IEEE Global Communications Conference*, 2022, pp. 1259–1264.

[34] A. Chawla, A.-M. Bosneag, and A. Dalgkitsis, "Graph-based interpretable anomaly detection framework for network slice management in beyond 5g networks," in *Proceedings of the IEEE/IFIP Network Operations and Management Symposium*, 2023, pp. 1–6.

Appendix

Appendix A Proofs and Supplementary Results

This appendix provides proofs for the results in §IV.

a) Proof of Theorem IV.1 (Detectability with spillover).

By the CMS guarantee, each bin estimate satisfies $x_{k,j} \leq \hat{x}_{k,j} \leq x_{k,j} + \varepsilon N_j$ with probability at least $1 - \delta$. Summing over the diagnostic bins T gives:

$$x_{k,T} \leq \hat{x}_{k,T} \leq x_{k,T} + \varepsilon N_T,$$

and summing over all bins gives:

$$x_k \leq \hat{x}_k \leq x_k + \varepsilon N'.$$

In baseline conditions, the maximum estimated diagnostic ratio is:

$$\frac{\hat{x}_{k,T}}{\hat{x}_k} \leq \frac{x_{k,T} + \varepsilon N_T}{x_k}.$$

During an anomaly adding Δ_T packets to T and $\beta\Delta_T$ elsewhere, the diagnostic mass increases while the total mass grows more modestly. The minimum estimated ratio becomes:

$$\frac{\hat{x}_{k,T}}{\hat{x}_k} \geq \frac{x_{k,T} + \Delta_T}{x_k + \Delta_T + \beta\Delta_T + \varepsilon N'}.$$

Detection succeeds when the anomaly-window ratio exceeds the baseline maximum. Rearranging this inequality yields the theorem's condition. □

b) Proof of Theorem IV.2 (Width requirement).

In the sparse diagnostic regime ($x_{k,T} \ll x_k$, $N_T \ll N'$), Theorem IV.1 simplifies to:

$$(1 - \beta)\Delta_T \gtrsim \varepsilon N_T, \quad \text{where } \varepsilon \approx e/w.$$

Enforcing the design constraints $N_T \leq N_T^{\max}$, $\Delta_T \geq \Delta_T^{\min}$, and $\beta \leq \beta_{\max}$ gives:

$$(1 - \beta_{\max})\Delta_T^{\min} \geq \frac{e}{w} N_T^{\max}.$$

Solving for w yields the required width. □

c) Proof of Depth Requirement Theorem.

Each CMS estimate violates its error bound with probability $\leq e^{-d}$. Detection involves K diagnostic-bin estimates and one total-count estimate ($K + 1$ events total). By union bound, all bounds hold simultaneously with probability $\geq 1-(K+1)e^{-d}$. Setting this $\geq 1 - \zeta$ gives:

$$d \geq \left\lceil \ln\left(\frac{K+1}{\zeta}\right) \right\rceil.$$

□

d) Proof of Target-Occupancy Proposition.

Let F be the baseline CDF. Placing the latency-tail boundary at the $(1 - \rho)$-quantile ensures exactly ρN packets have latency above the boundary. Similarly, placing the IAT-head boundary at the ρ-quantile ensures exactly ρN packets have IAT below the boundary. Thus, $N_T^{\max} = \rho N$. □

e) Proof of Drift Effect Proposition.

From Theorem IV.2, required width scales as $w \propto N_T^{\max}$. If diagnostic occupancy increases from ρN to $\rho' N$, then $w_{\text{new}}/w_{\text{old}} = \rho'/\rho$. □

f) Practical Implications.

These results explain the parameter choices in §IV-B: width w scales linearly with $N_T^{\max}$ but inversely with $\Delta_T^{\min}$, while depth d grows only logarithmically with K. This explains why moderate parameters ($w = 512$, $d = 3$–4) suffice in practice, as illustrated in the examples of §IV-B.