[ICOIN 2026 Tutorial]

Toward 6G-Native Intelligence: Data-Driven Approaches for AI-RAN

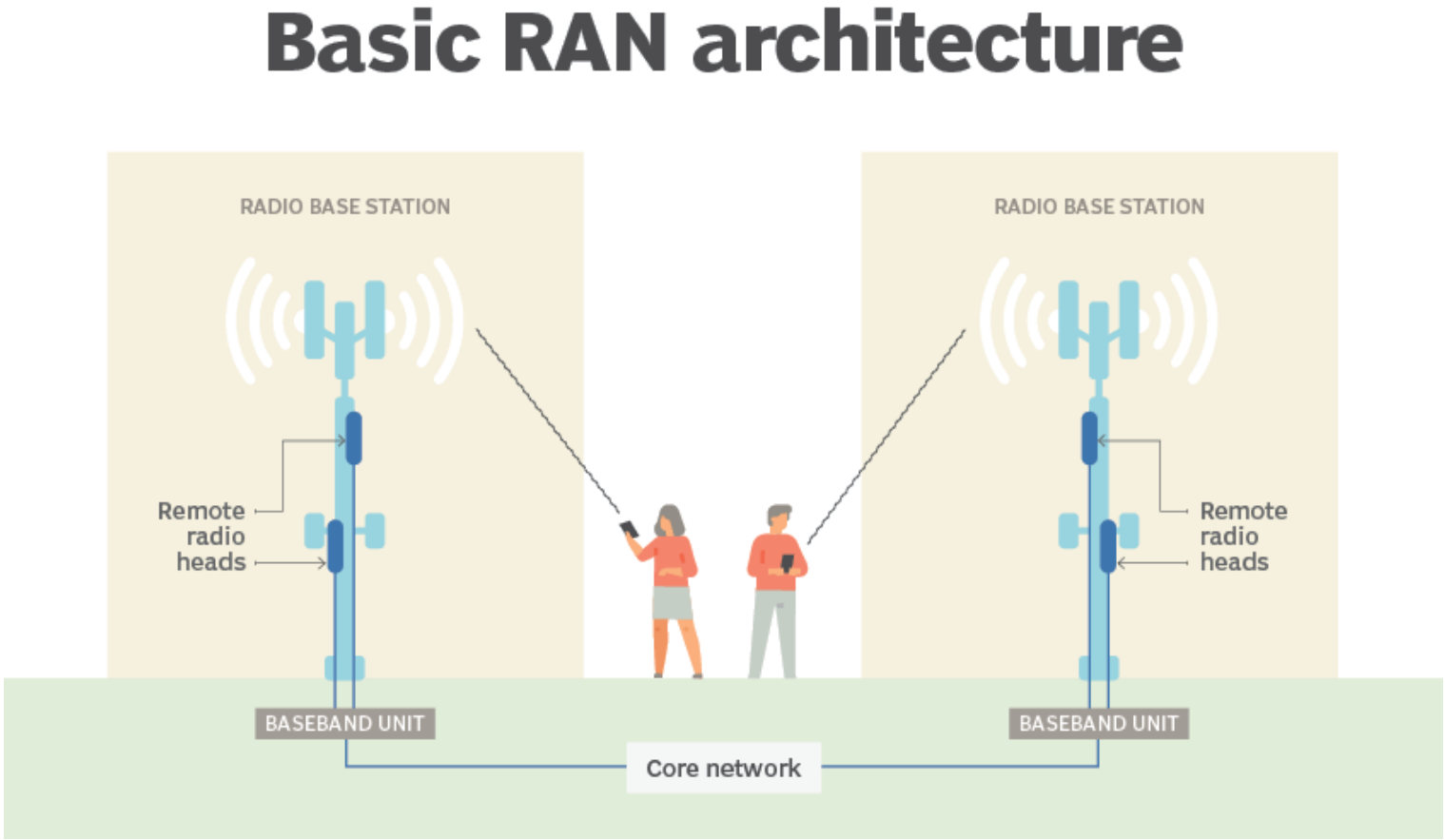
Prof. Hyunggon Park

Email: hyunggon.park@ewha.ac.kr

Multiagent Communications and Networking Laboratory
Department of Electronic and Electrical Engineering
Ewha Womans University, Seoul, Korea
https://mcnl.ewha.ac.kr

Motivation: Radio Access Network

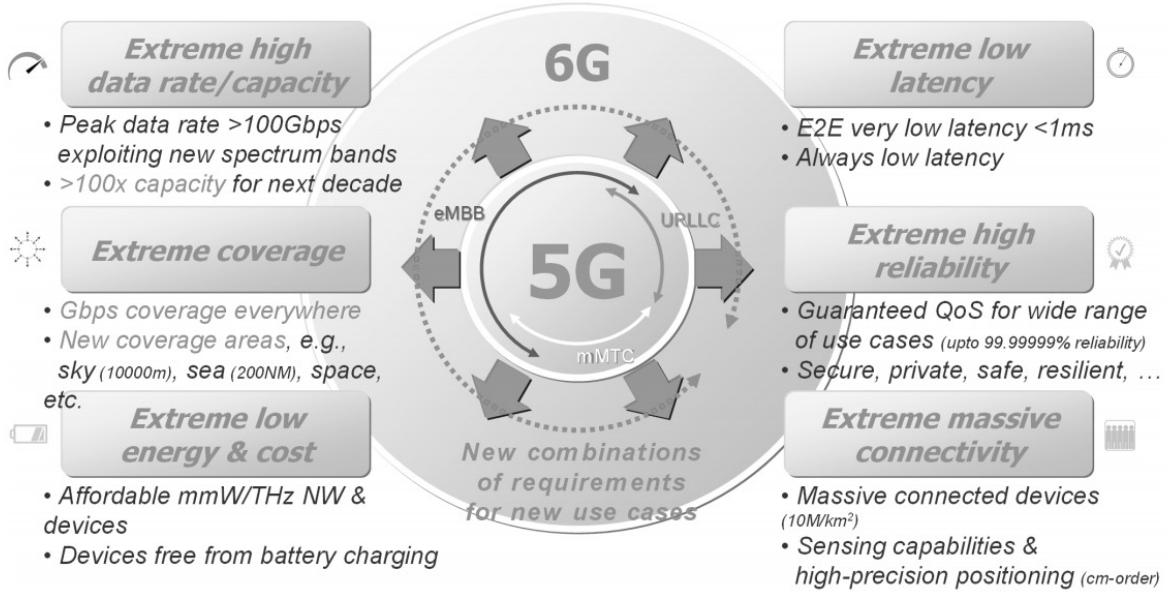
- RAN (Radio Access Network) connects user equipment (UE) to the core network via radio links and backhaul (fiber/wireless)
- Serves as the radio interface of a cellular network, enabling wireless access to network services
- RAN evolution from 1G to 5G has significantly increased system complexity and capability



[Source: TechTarget]

Motivation

- Why do we need AI for RAN?
  - Significantly increased complexity for 5G/5G-Adv and 6G networks
  - Challenging KPIs: data rate, coverage, energy efficiency, latency, reliability, connectivity, etc.
  - Limitations of typical rule-based / model-based control

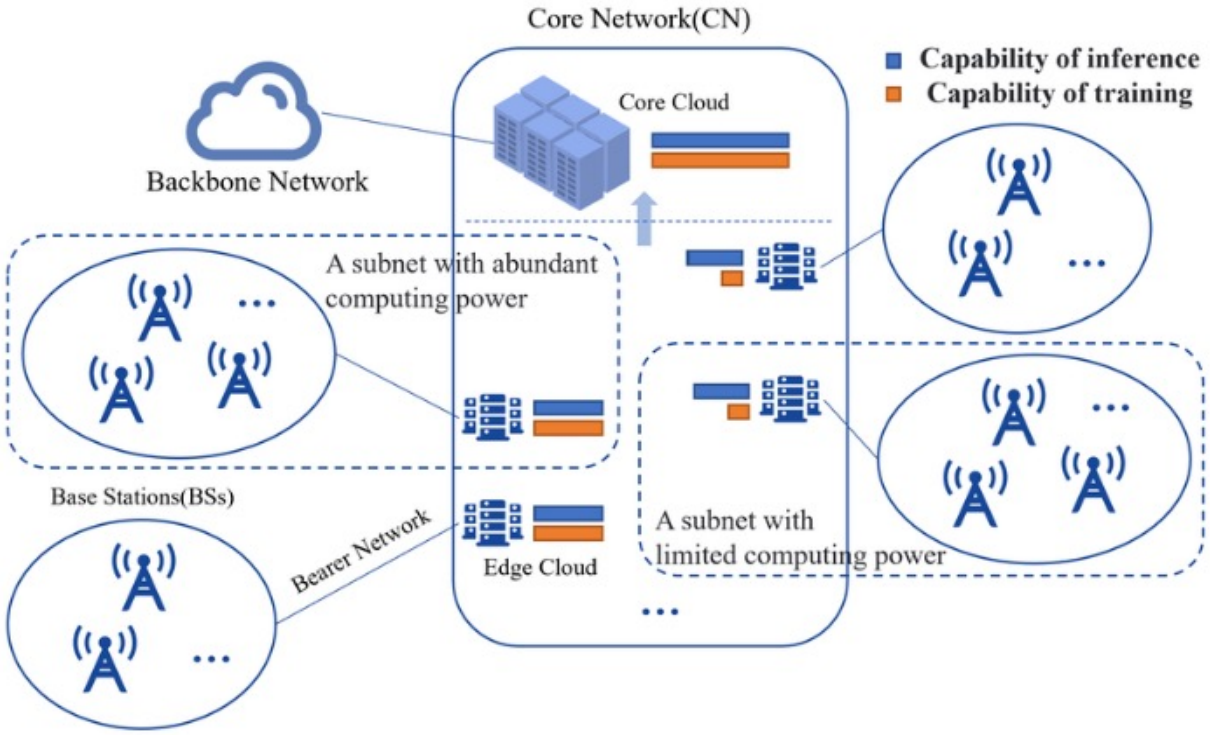


[Source: Non-Orthogonal Physical Layer (NOPHY) Design towards 5G Evolution and 6G, *IEICE Trans. Commun.*, 2022]

→ No longer controlled by rules designed by humans → AI may be a solution

Motivation

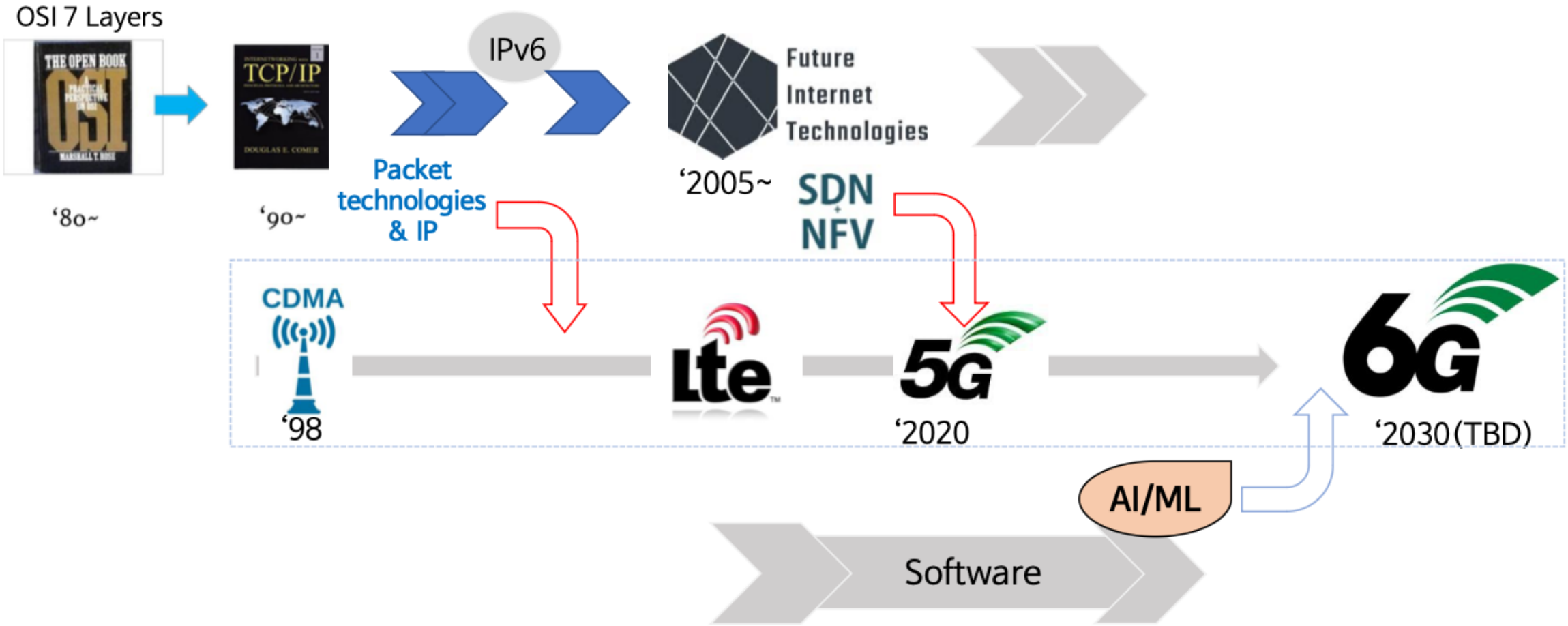
- What we need is network intelligence
  - Network intelligence (NI) is an enabling technology that allows communications service providers (CSPs) to capture subscriber-, service- and application-level awareness contained in network traffic.
  - This information is analyzed and exposed for integration with other applications in the back office, allowing CSPs to apply granular policies to influence customer experience and adapt to dynamic shifts in application and service usage.
  - The solution is based on nonproprietary hardware and software platforms and can be used by CSPs on any network. [Source: Gartner 2024]



<Core network and its connections>

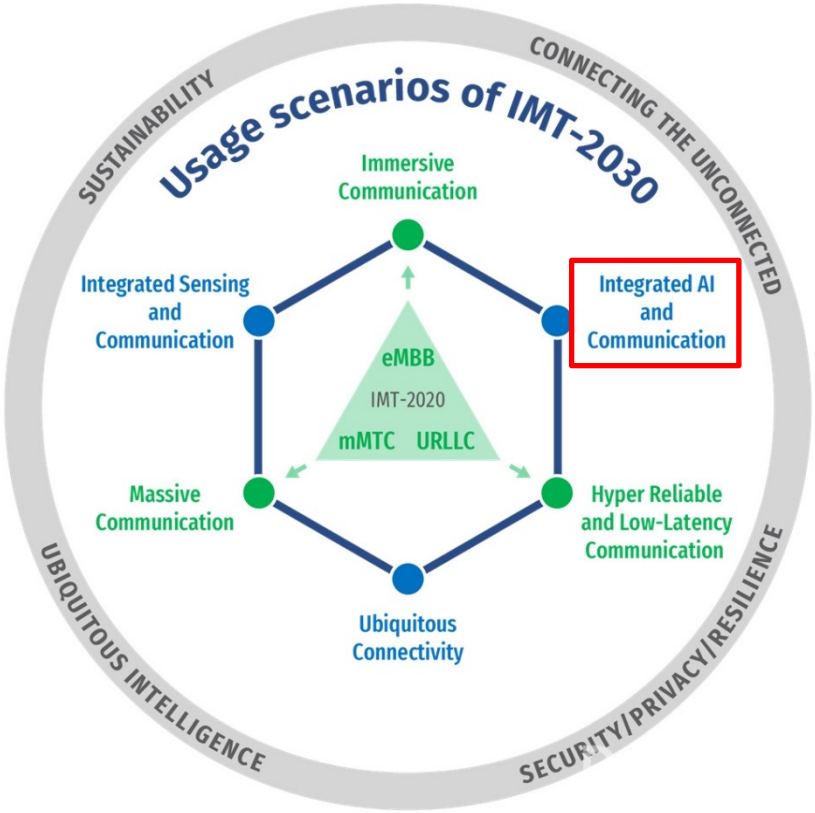
Motivation

- In the perspective of standards…

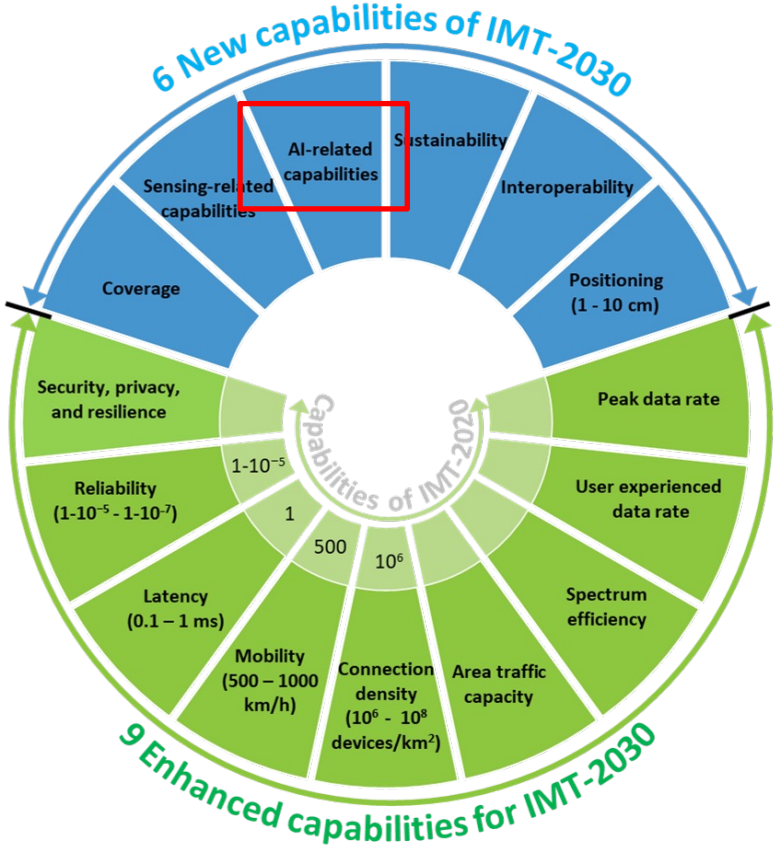


Motivation

- In the perspective of standards…



IMT 2030 – usage scenarios



IMT 2030 – 6G capabilities

[Source: Recommendation ITU-R M.IMT. Framework for 2030 and beyond]

Motivation

- 5G-Advanced vs. 6G
  - AI-native 6G System
    - AI is incorporated into major functionality from the very beginning of design and development of systems

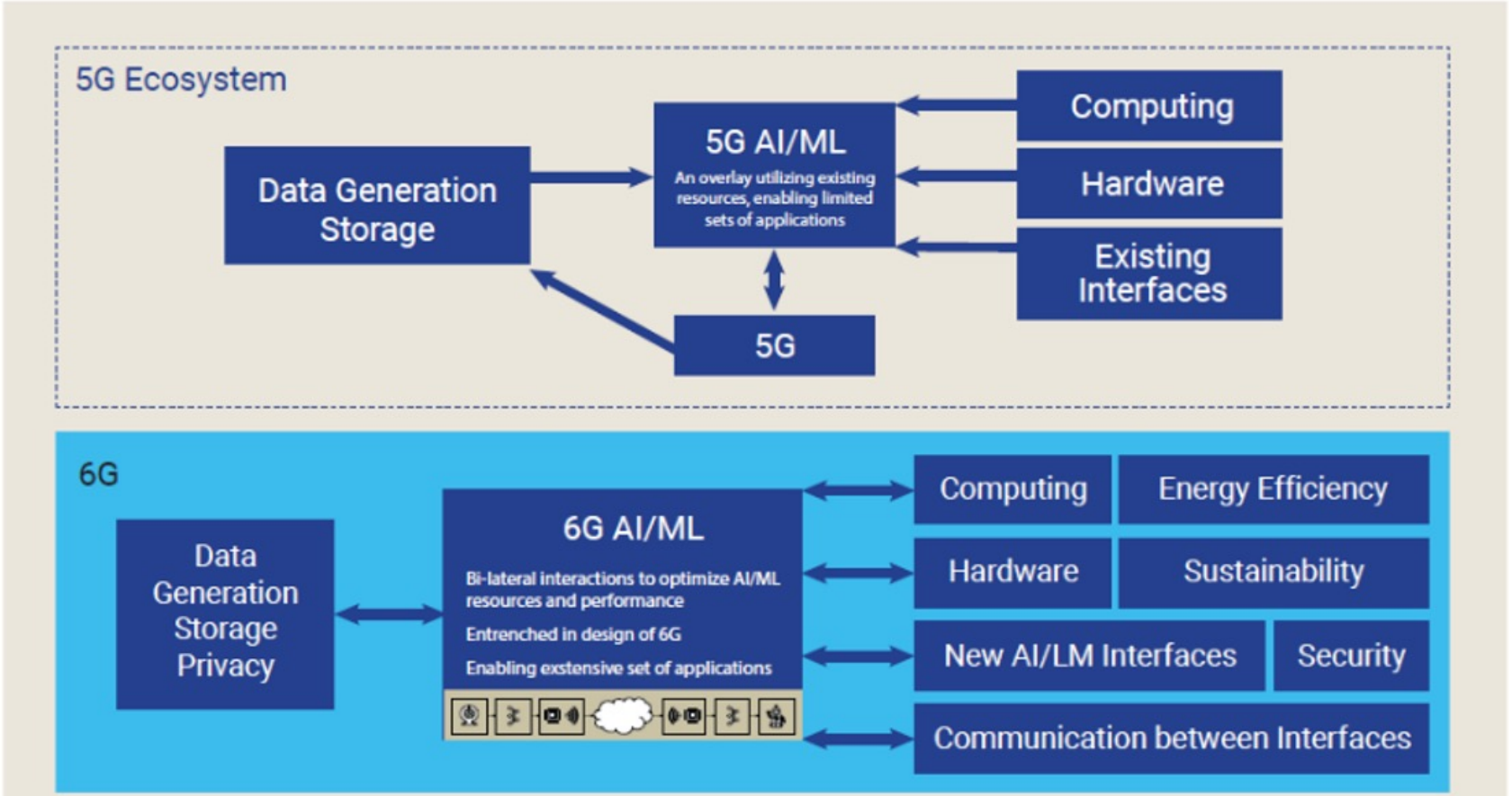
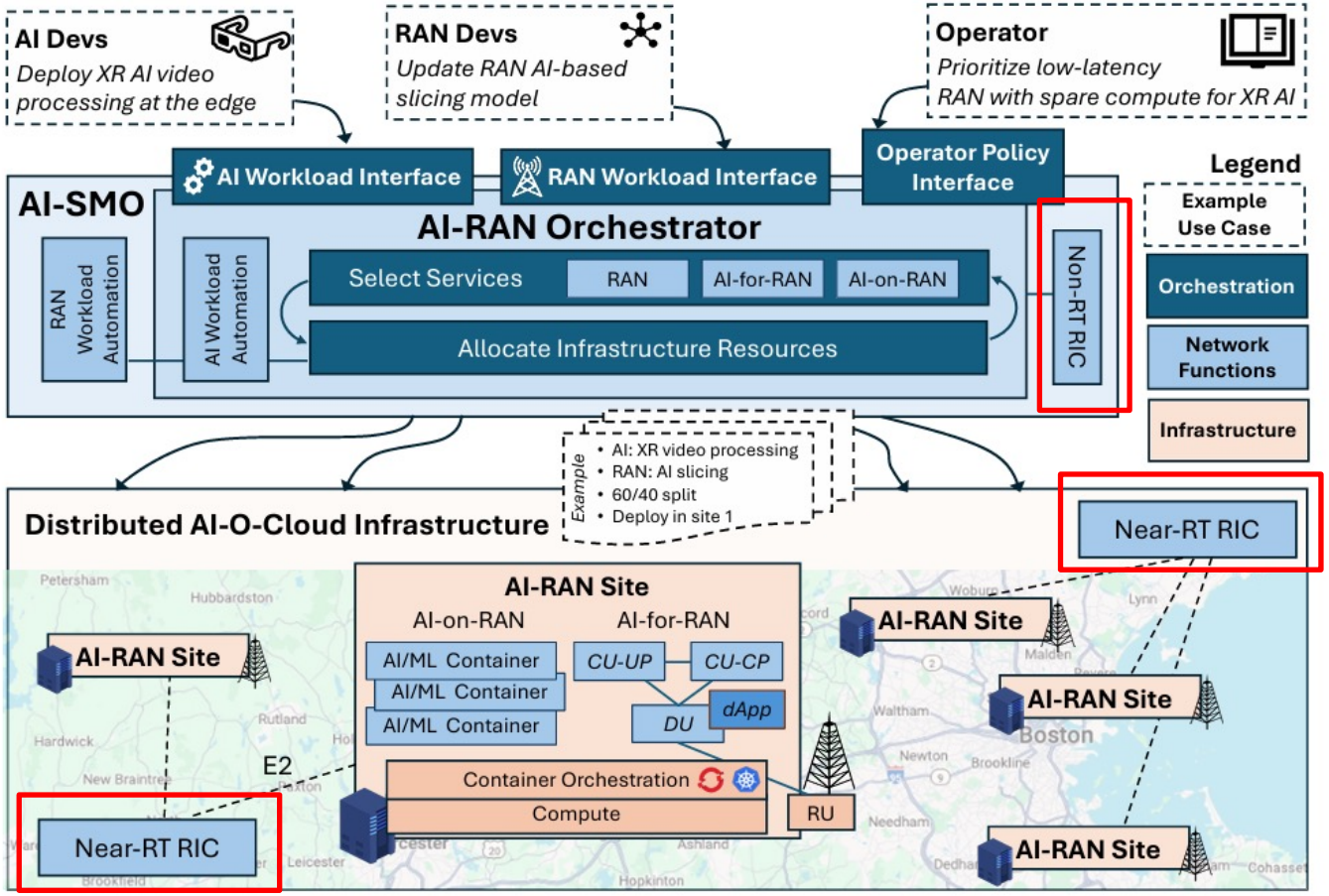


Figure 1: 5G AI/ML and 6G AI/ML [1]

AI-RAN Reference Architecture

- Non-RT RIC(RAN Intelligence Controller)/ Near-RT RIC/ RT
- Data collection → analytics → decisions → control



[Source: M. Polese, et. al, "Beyond Connectivity: An Open Architecture for AI-RAN Convergence in 6G," arXiv, 2025.

* SMO: Service Management and Orchestration

AI/ML for AI-RAN

- Representative Use Cases
  - Prediction (traffic, mobility)
  - Optimization (scheduling, power control)
  - Anomaly detection
  - Etc.

AI-RAN ALLIANCE

Integrating AI/ML in Open-RAN: Overcoming Challenges and Seizing Opportunities

Domain Adaptation for Effective AI in Open-RAN

1. Importance of Domain-Specific Training:

- **Description:** The effectiveness of AI models in telecom applications depends on their exposure to relevant domain-specific data during training.
- **Challenge:** Ensuring that AI models have been trained on telecom-specific data to accurately perform tasks such as network optimization, fault detection, and customer service automation.
- **Solution:** Collaborate with telecom operators to access domain-specific datasets for training. Develop partnerships with academic and research institutions to advance domain-specific AI research.

2. Approaches to Domain Adaptation:

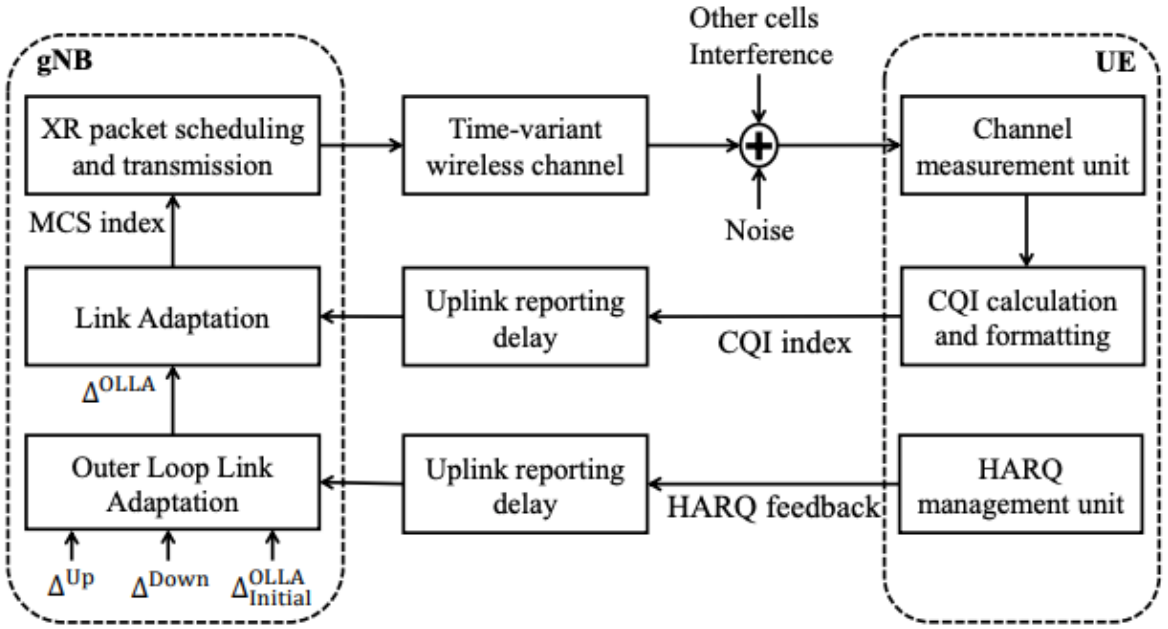
- **Transfer Learning:** Utilizing pre-trained models and fine-tuning them on telecom-specific data to enhance their performance in telecom applications.
- **Custom Training:** Developing AI models specifically trained on telecom datasets to ensure high accuracy and relevance.
- **Example:** Implement transfer learning techniques to adapt pre-trained AI models to handle telecom-specific tasks like network traffic prediction and anomaly detection.

[Source: Integrating AI/ML in Open-RAN: Overcoming Challenges and Seizing Opportunities, AI-RAN Alliance, 2024]

What about ingredients for AI/ML for networks?

- Data available in RAN
  - UE-side: CQI, RSRP, SINR, mobility events
  - gNB-side: scheduling, HARQ, buffer status
  - Network level: traffic load, handover stats
- Characteristics
  - High-dimensional
  - Spatio-temporal correlation
  - Noisy and partially observable
  - Non-stationary
- Challenges in Using RAN Data
  - Data sparsity and imbalance
  - Label scarcity
  - Privacy and real-time constraints

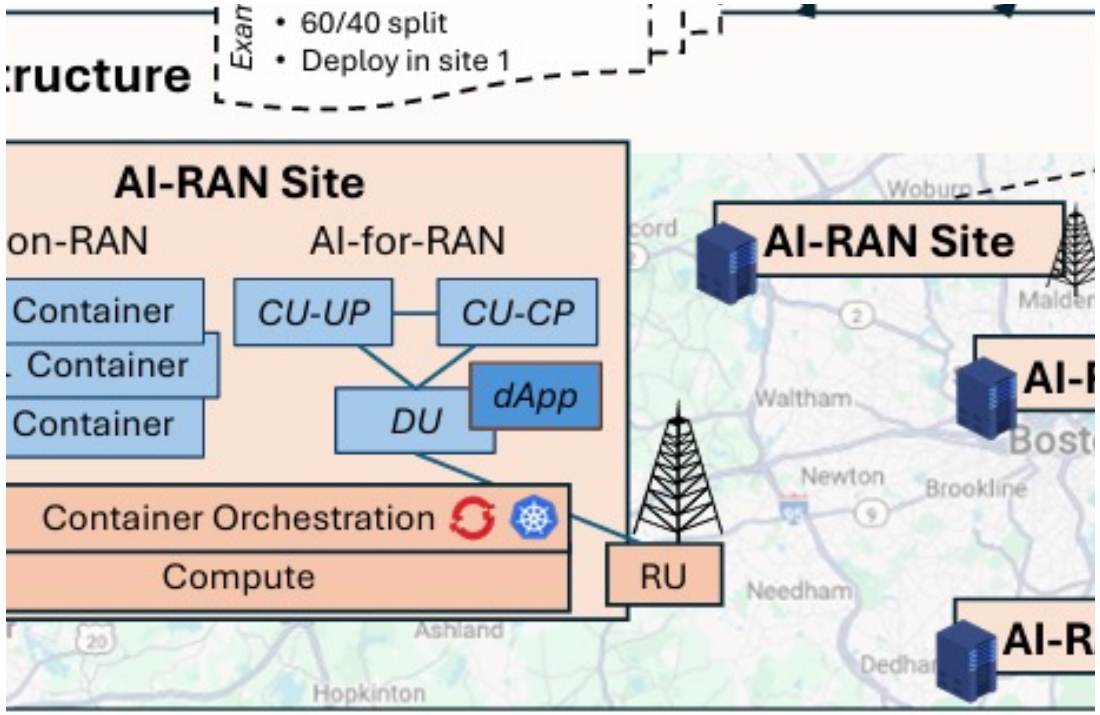
→ Abundant data, but cannot be used directly



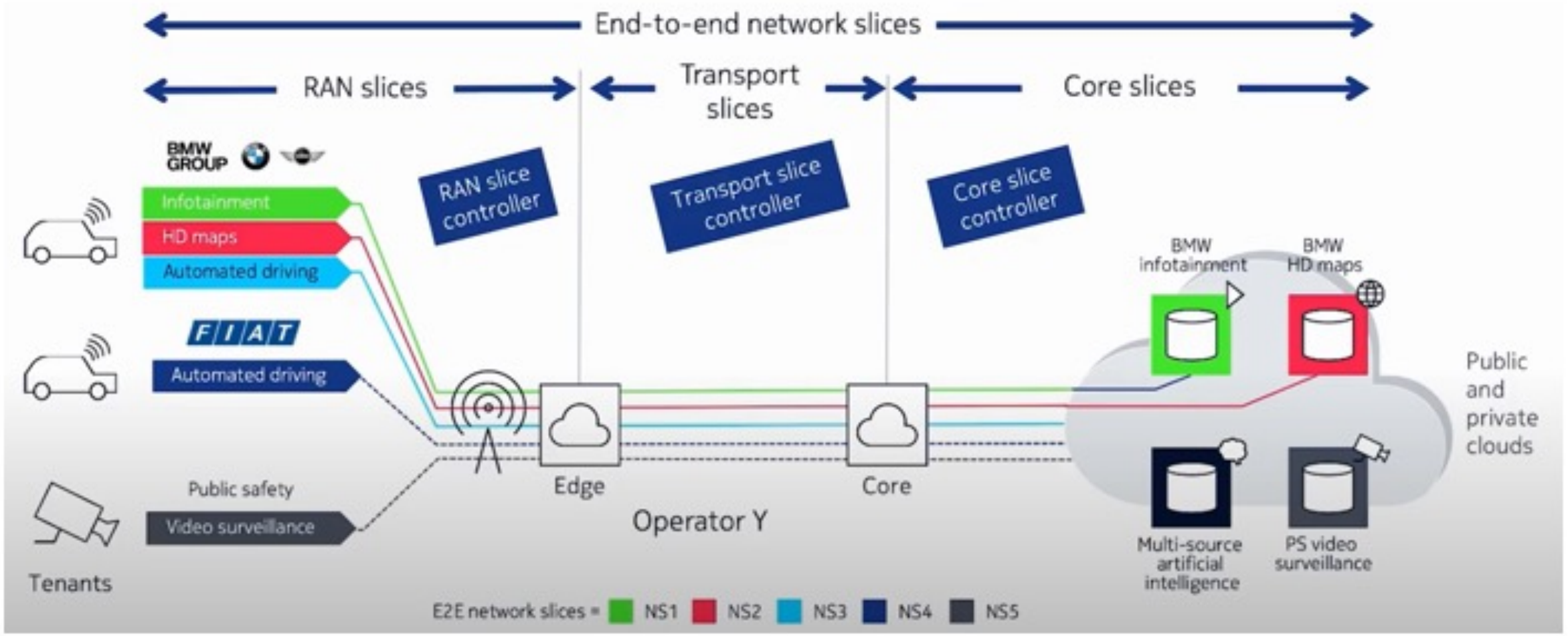
[P. Paymard, et. al., "Extended Reality over 3GPP 5G-Advanced New Radio: Link Adaptation Enhancements, arXiv, 2022]

In this tutorial,

- With two use cases
  - explicit consider communication and network constraints with AI/ML deployment
  - highlight tradeoffs between AI/ML algorithms
- Use cases
  - Slice classification
    - Feature extraction and explanation
    - Efficient AI/ML algorithms
    - Model selection
  - Network traffic anomaly detection
    - Feature selection algorithms and explanation
    - Extensive evaluation of anomaly detection algorithms



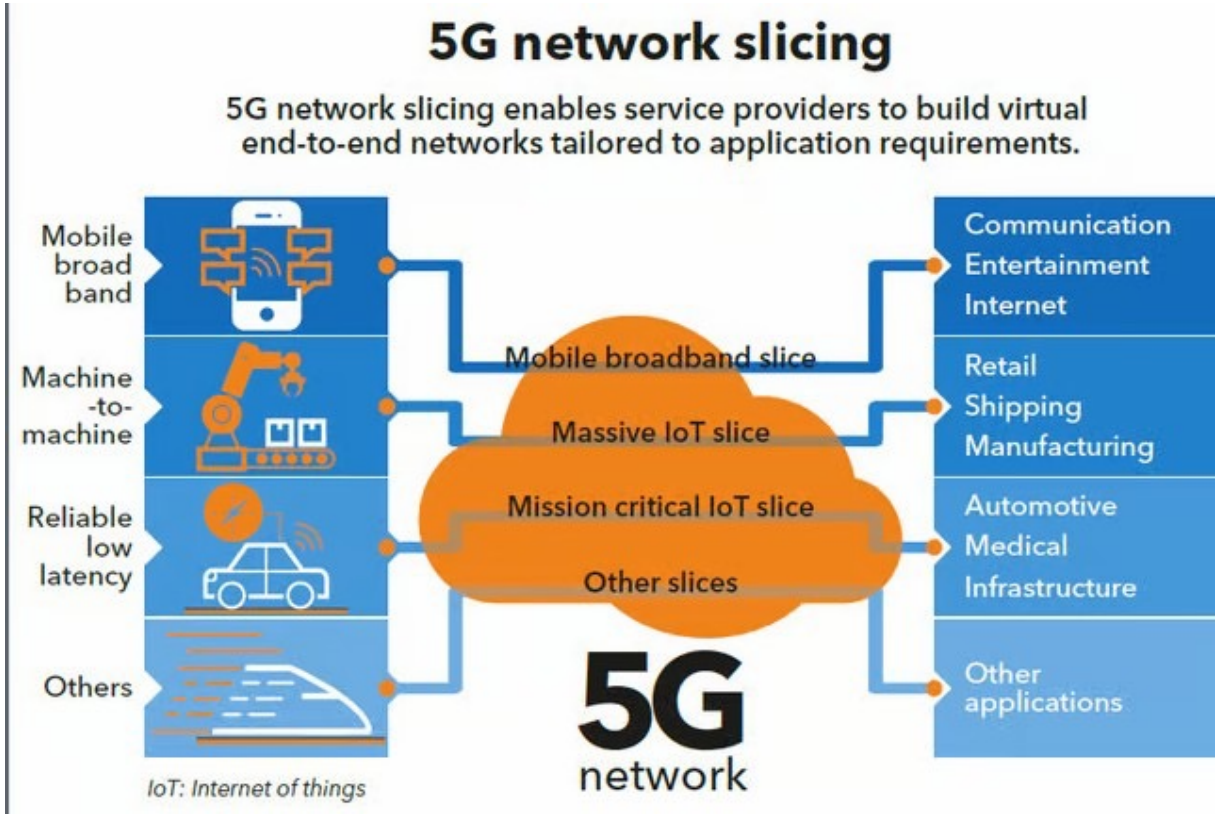
Use Case 1 – ML with Efficient Features for Slice Classification



[Source: 5G network slicing: automation, assurance and optimization of 5G transport slices, Nokia]

Introduction

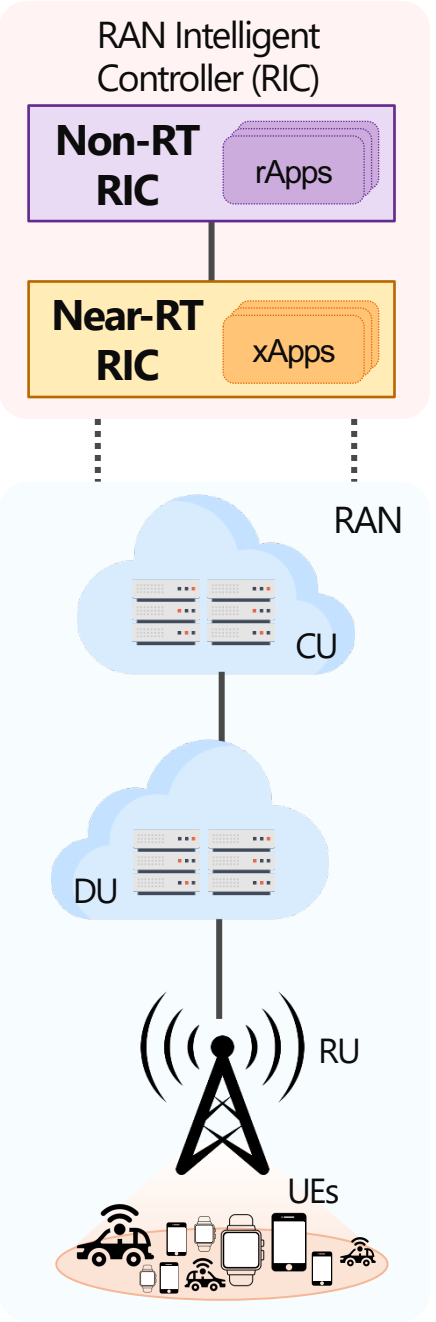
- Network slicing
  - Partitioning physical infrastructure into multiple logical slices
    - allows a single physical network to be partitioned into multiple virtual slices
    - tailored to specific SLAs and KPIs (e.g., latency, throughput, reliability, capacity)
    - enables service-specific optimization for diverse applications (public and private networks)
    - Improves resource efficiency and flexibility by matching network behavior to application demands
  - Representative service requirements
    - enhanced Mobile Broadband (eMBB)
    - Ultra-Reliable Low-Latency Communications (URLLC)
    - massive Machine-Type Communications (mMTC)



[Source: 5G Technology World, 2025]

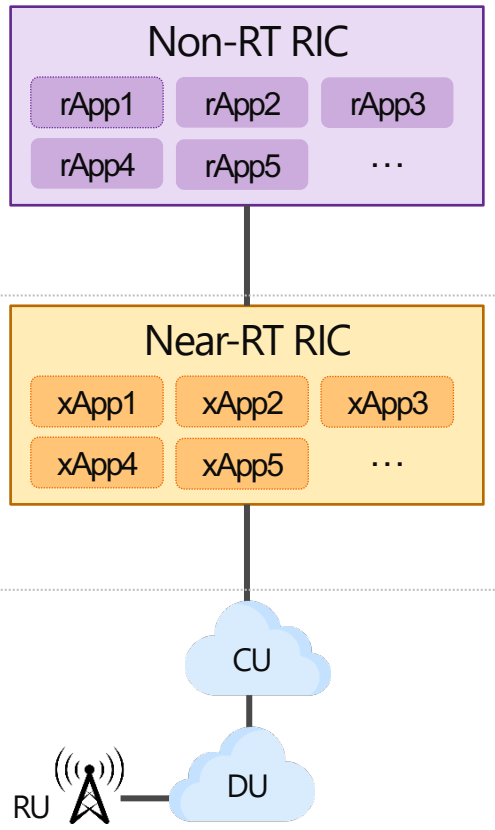
Introduction

- Why importance of network slice classification at RAN level?
  - Starting point of slice-aware orchestration
    - Guaranteeing Service Level Agreement (SLA) compliance and QoS management
    - Enabling resource allocation and scheduling
    - Ensuring isolation between slices
  - In RAN, RAN Intelligence Controller (RIC) can be responsible for slice classification



Introduction

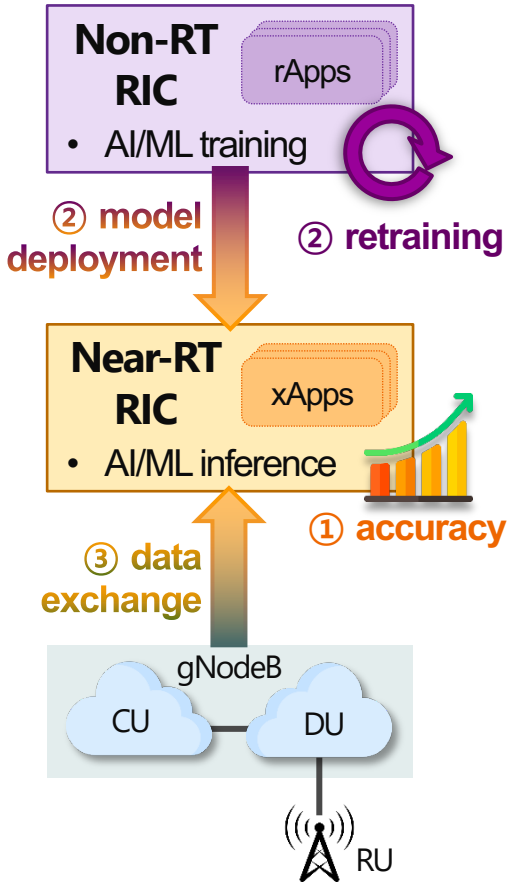
- RAN Intelligent Controller (RIC) in O-RAN architecture
  - Responsible for AI/ML-driven control and optimization



Latency	Intelligence layers
>> 1 sec	– Non-real-time RIC (Non-RT RIC) • Hosting rApps for policy and model management • Offline training and large-scale data analytics
1 msec~1sec	– Near-real-time RIC (Near-RT RIC) • Hosting xApps for real-time RAN optimization • Online inference and control with ~1sec latency
< 10msec	– Central Unit (CU) and Distributed Unit (DU) • Interfacing with RIC for real time actions

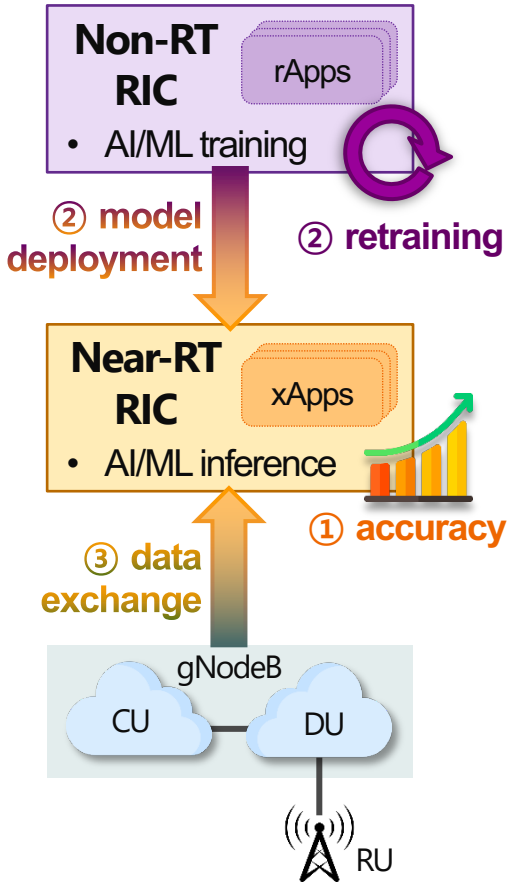
Challenges

- Requirements in O-RAN slice classification
  - ① High classification accuracy under strict latency constraints
  - ② Frequent retraining and fast deployment
  - ③ Efficient operation by avoiding high-dimensional data exchange



Challenges

- Requirements in O-RAN slice classification
  - ① High classification accuracy under strict latency constraints
  - ② Frequent retraining and fast deployment
  - ③ Efficient operation by avoiding high-dimensional data exchange
- Related work [1]-[5]
  - Mainly focusing on classification performance
  - Deploy deep learning (DL)-centric approaches
- Limitations of recent work
  - Rarely considering training cost, retraining feasibility, and update overhead
  - Lack of exploration of lightweight ML classifiers



[1] M. Belgiovine, et al., "MEGATRON: Machine Learning in 5G with Analysis of Traffic in Open Radio Access Networks," in *ICNC*, 2024, pp. 1054–1058.
[2] J. Groen, et al., "TRACTOR: Traffic Analysis and Classification Tool for Open RAN," in *IEEE ICC*, 2024, pp. 4894–4899.
[3] L. Bonati, et al., "Colosseum: Large-Scale Wireless Experimentation Through Hardware-in-the-Loop Network Emulation," in *IEEE DySPAN*, 2021, pp. 105–113.
[4] C. Tassie, et al., "Leveraging Explainable AI for Reducing Queries of Performance Indicators in Open RAN," in *IEEE ICC*, 2024, pp.5413–5418.
[5] M. Polese, et al., "ColO-RAN: Developing Machine Learning-Based xApps for Open RAN Closed-Loop Control on Programmable Experimental Platforms," *IEEE Transactions on Mobile Computing*, vol. 22, no. 10, pp. 5787–5800, 2023.

Goal

- Consideration of data characteristics and O-RAN deployment constraints in classification model selection
  - Comparison of ML and DL classifiers

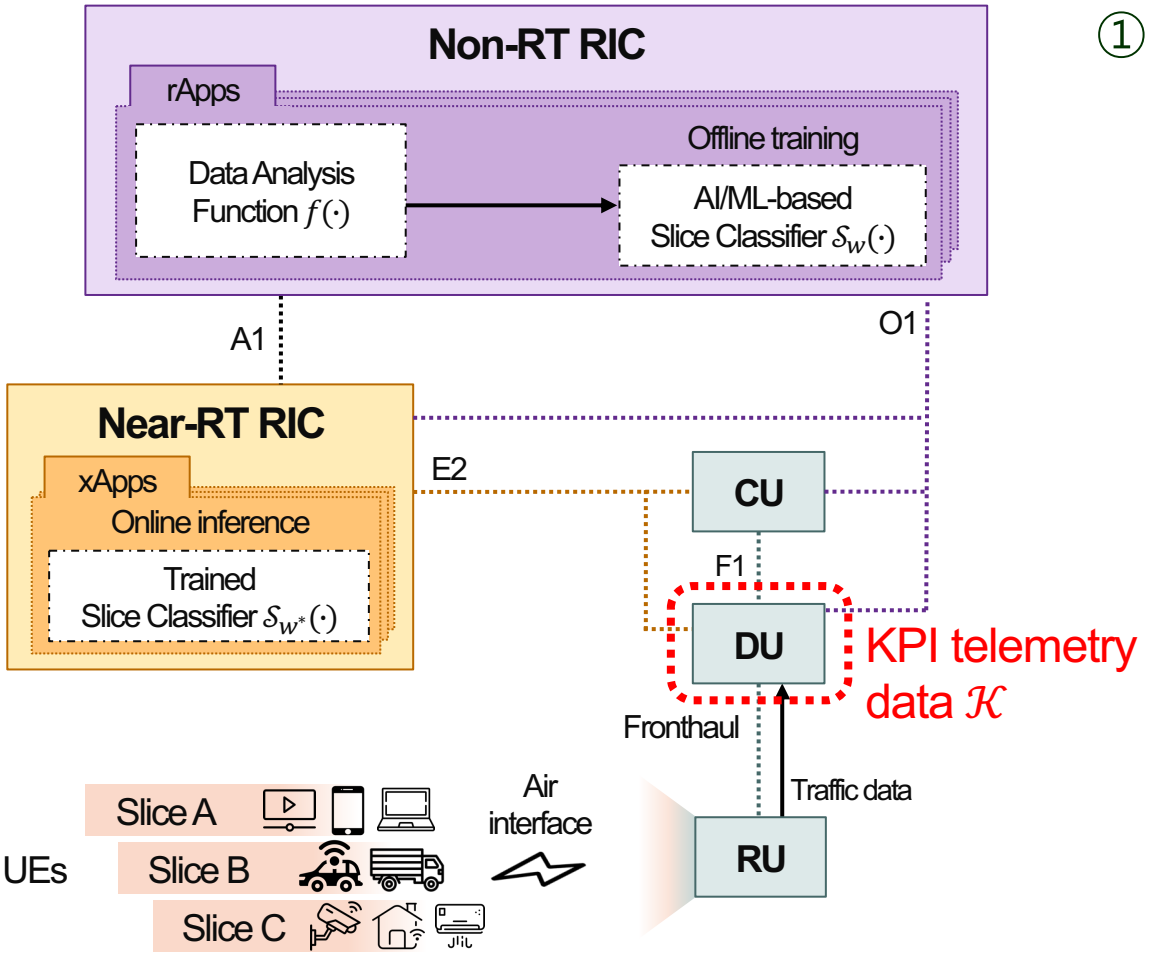
	Machine learning (ML)	Deep learning (DL)
Data processing ability	• Sufficient for simpler and structured data • Domain-driven feature engineering	• Strong for complex and non-linear structured data • Representation learning
Model complexity	• Low complexity • Lightweight and fast	• High complexity • Heavy computation

Systematic evaluation of ML and DL for O-RAN slice classification

[D. Choi, S. Park, J. Kwon and H. Park, "Few Features are Enough: Communication-Efficient AI-RAN," *Conference on Neural Information Processing Systems (NeurIPS 2025) (AI and ML for Next-Generation Wireless Communications and Networking (AI4NextG))*, 2025.]

System Overview

- O-RAN architecture with data analysis function

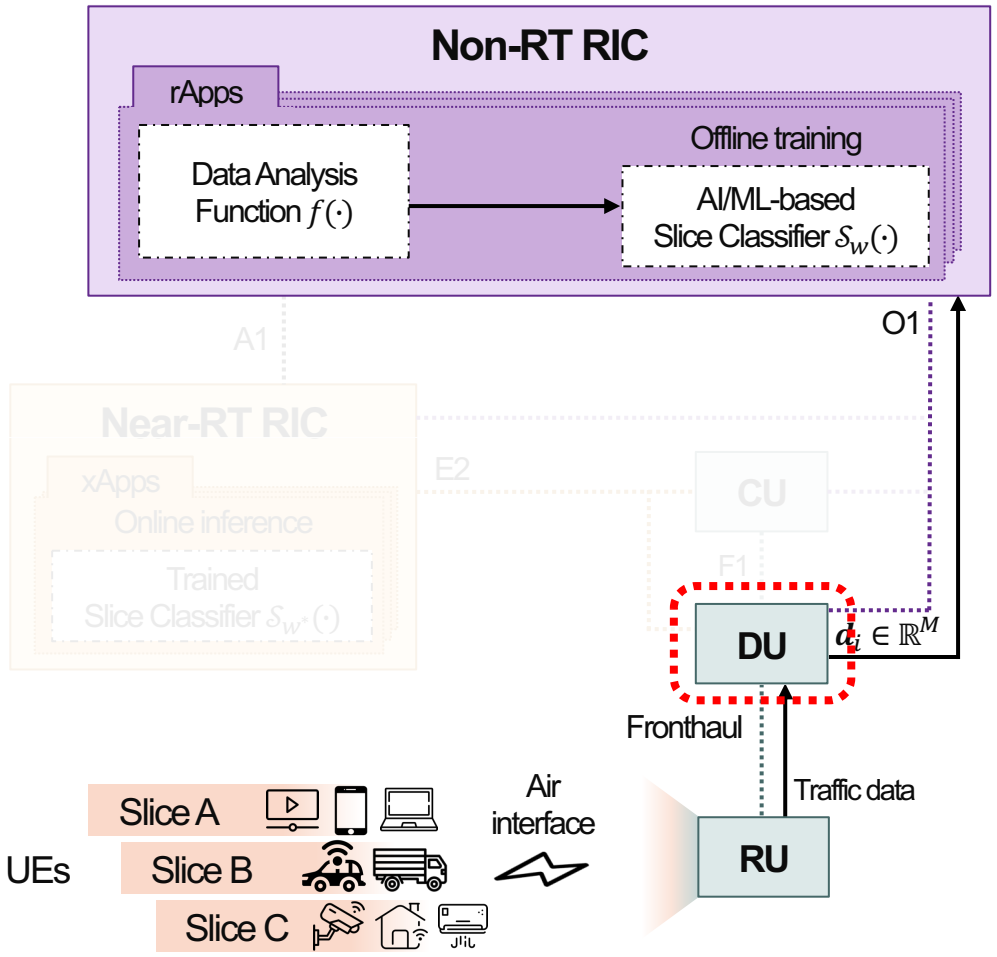


① Data collection at RAN

- UE → RU → DU → RIC
- Key Performance Indicators (KPIs) telemetry data $\mathcal{K} = \{\mathbf{k}_1, \ldots, \mathbf{k}_M\}$ extracted at DU

System Overview

- O-RAN architecture with data analysis function



① Data collection at RAN

- UE → RU → DU → RIC
- Key Performance Indicators (KPIs) telemetry data

  $\mathcal{K} = \{\mathbf{k}_1, \ldots, \mathbf{k}_M\}$ extracted at DU

<KPI telemetry data $\mathcal{K}$>

$\mathbf{k}_j \in \mathbb{R}^T$: KPI feature vector

$\boldsymbol{d}_i \in \mathbb{R}^M$
: traffic instances

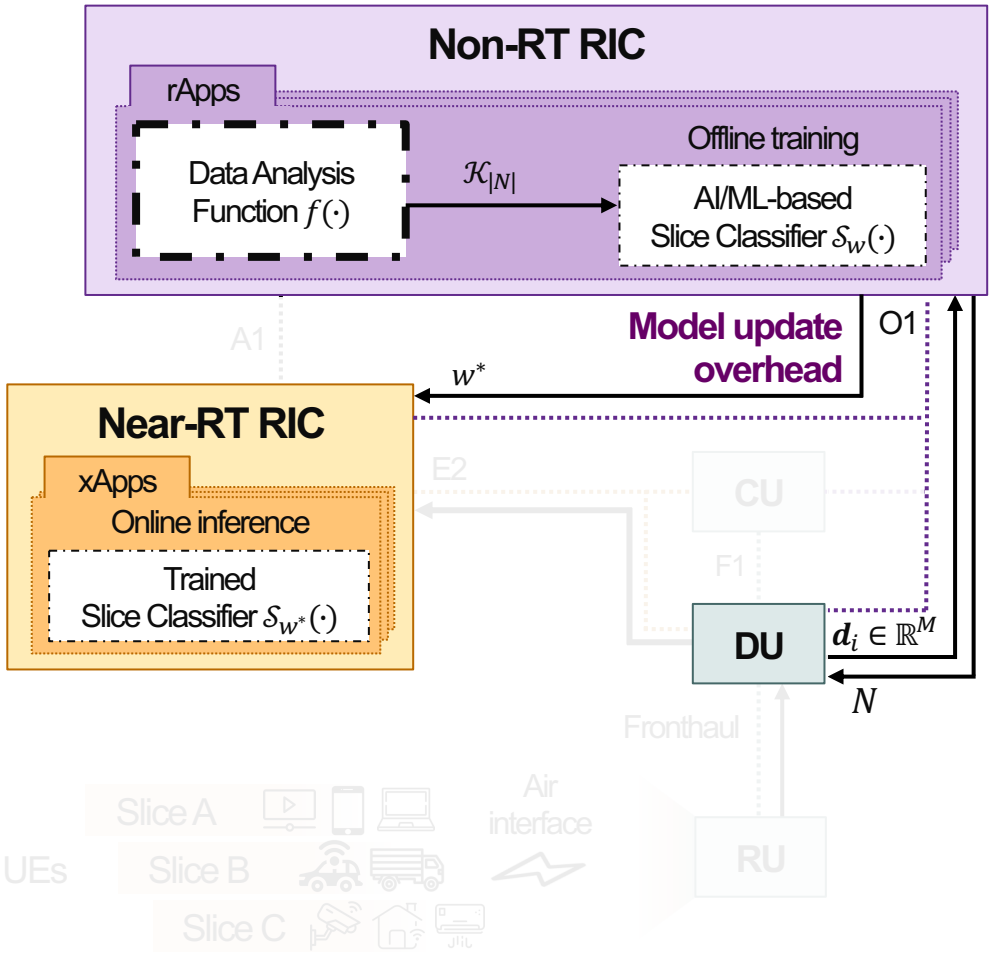
	$\mathbf{k}_1$	$\mathbf{k}_2$	⋯	$\mathbf{k}_M$	$\mathbf{y}$
$\boldsymbol{d}_1$					eMBB
$\boldsymbol{d}_2$					URLLC
⋮	⋮	⋮	⋮	⋮	⋮
$\boldsymbol{d}_T$					mTMC

T instances

M KPI features

System Overview

- O-RAN architecture with data analysis function

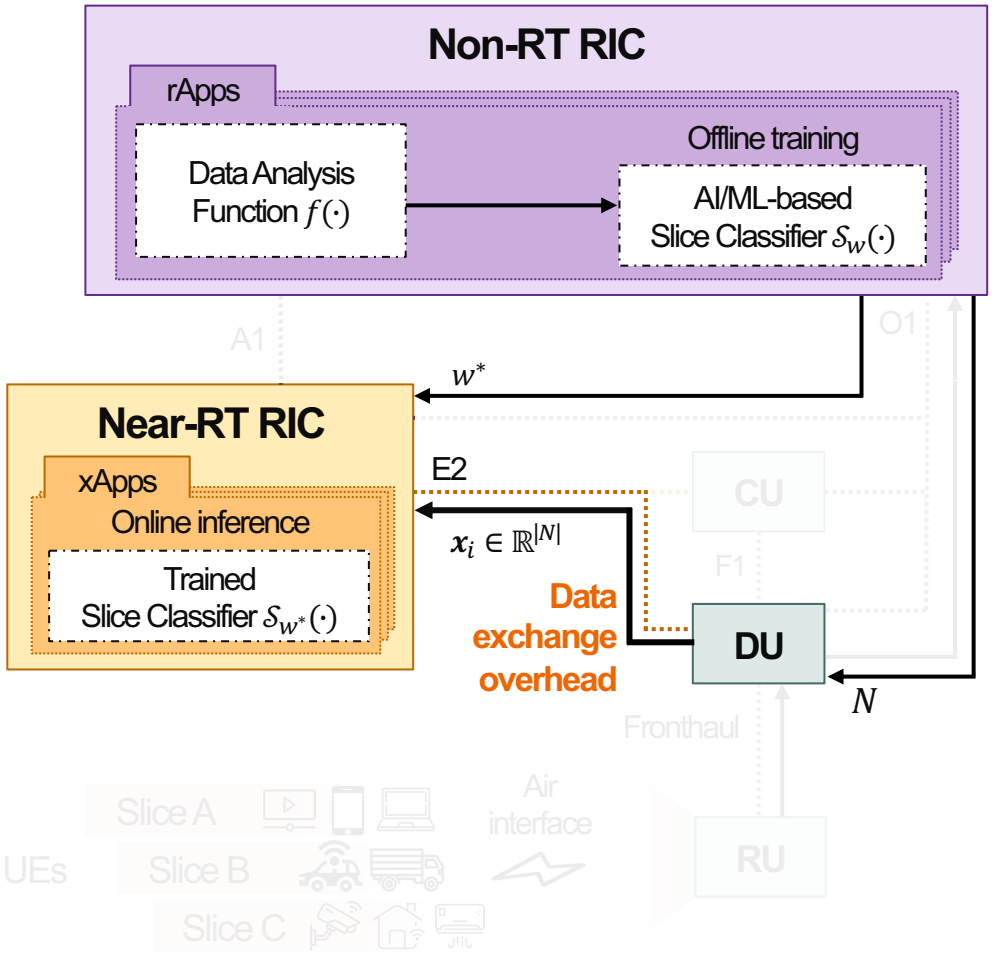


② Data analysis and model selection in Non-RT RIC

- Large-scale KPI telemetry data $\mathcal{K}$ analyzed via data analysis function $f(\cdot)$
  - N : index set of KPI features
  - $\mathcal{K}_{|N|} = \{\mathbf{k}_n | n \in N\}$: compact KPI feature subset
- Slice classifier $\mathcal{S}_w(\cdot)$ trained/updated to optimal parameter w^*
- Transmitting N to DU via O1
- Deploying w^* to Near-RT RIC via O1

System Overview

- O-RAN architecture with data analysis function



③ Online inference in Near-RT RIC

- Reduced KPI vectors $\boldsymbol{x}_i \in \mathbb{R}^{|N|}$ continuously transmitted from DU via E2
- Near-real-time slice classification using $\mathcal{S}_{w^*}(\cdot)$

Network Data Analysis

- Colosseum KPI dataset
  - Two datasets from Colosseum testbed
    - COMMAG [6] : 40 UEs per 4 base stations
    - ColO-RAN [5] : 42 UEs in 7 base stations
    - Slice labels: eMBB, URLLC, mMTC
  - 21 KPI measurements features
    - Collected every 250ms
    - Captured radio traffic characteristics

KPI features	Descriptions
dl_mcs	Downlink modulation and coding scheme
dl_n_samples	Number of downlink samples
dl_buffer	Downlink queue length in bytes
tx_brate_downlink	Downlink bitrate in Mbps
tx_pkts_downlink	Downlink number of packets transmitted
tx_errors_downlink	Downlink percent of packets with errors
dl_cqi	Downlink channel quality indicator
ul_mcs	Uplink modulation and coding scheme
ul_n_samples	Uplink number of samples
ul_buffer	Uplink queue length in bytes
rx_brate_uplink	Uplink bitrate in Mbps
rx_pkts_uplink	Uplink number of packets received
rx_errors_uplink	Uplink percent of packets with errors
ul_rssi	Uplink received signal strength indicator
ul_sinr	Uplink signal to interference plus noise ratio
phr	UE power head room
sum_requested_prbs	Total requested physical resource blocks
sum_granted_prbs	Total granted physical resource blocks
dl_pmi	Downlink precoding matrix indicator
dl_ri	Downlink rank indicator
ul_turbo_iters	Uplink turbo encoding iterations

< O-RAN compliant KPI features ($M = 21$) >

[5] M. Polese, et al., “ColO-RAN: Developing Machine Learning-Based xApps for Open RAN Closed-Loop Control on Programmable Experimental Platforms,” *IEEE Transactions on Mobile Computing*, vol. 22, no. 10, pp. 5787–5800, 2023.
[6] L. Bonati, et al., “Intelligence and Learning in O-RAN for Data-driven NextG Cellular Networks,” *IEEE Communications Magazine*, vol. 59,no. 10, pp. 21–27, 2021.

Network Data Analysis

	$\mathbf{k}_1$	$\mathbf{k}_2$	⋯	$\mathbf{k}_M$	$\mathbf{y}$
$\boldsymbol{d}_1$					eMBB
$\boldsymbol{d}_2$					URLLC
⋮	⋮	⋮	⋮	⋮	⋮
$\boldsymbol{d}_T$					mMTC

$\mathbf{k}_j$: j-th KPI feature vector

- KPI feature importance based on mutual information
  - Measure the impact of individual KPI features
  - Mutual information value I_j of feature j

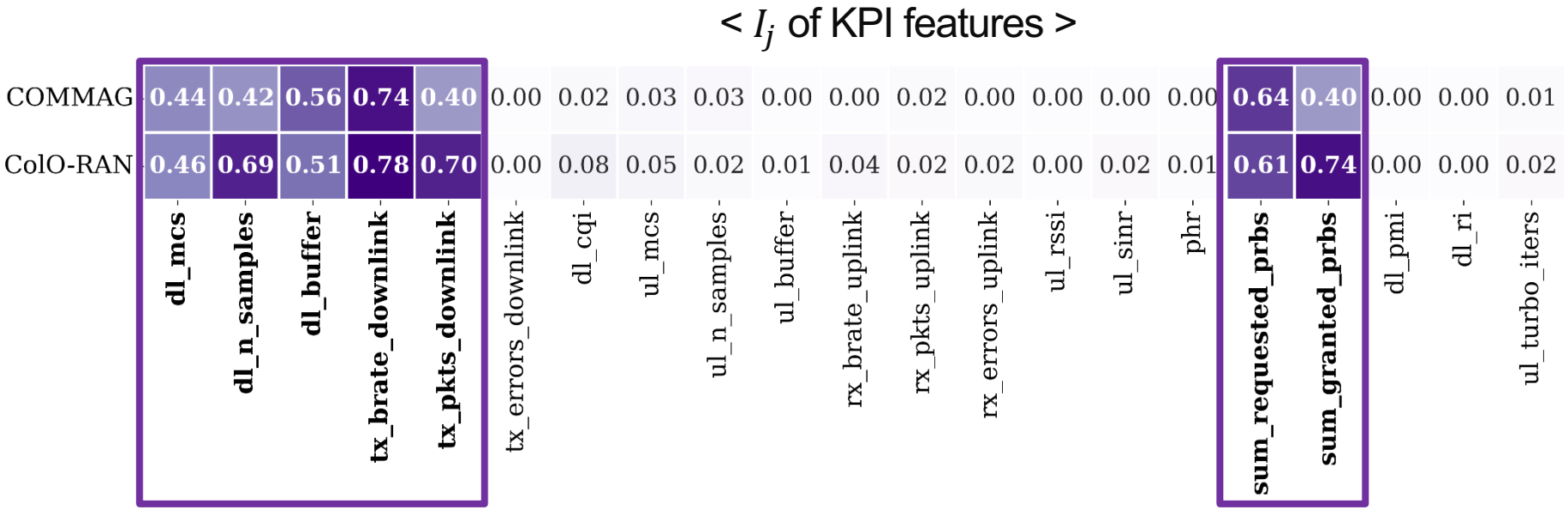
$$I_j = \sum_{y \in \mathcal{Y}} \sum_{k_j \in \mathcal{K}_j} P(k_j, y) \log \left(\frac{p(k_j, y)}{p(k_j) p(y)} \right)$$

- k_j : random variable of features
- y : random variable of label
- $\mathcal{K}_j$: discrete sets of possible values for k_j
- $\mathcal{Y}$: discrete sets of possible values for y

- Statistical dependency between random variable k_j and y
- Meaning: how much knowing one variable reduces the uncertainty of another
- Higher I_j → Feature j contributes more to slice classification

Network Data Analysis

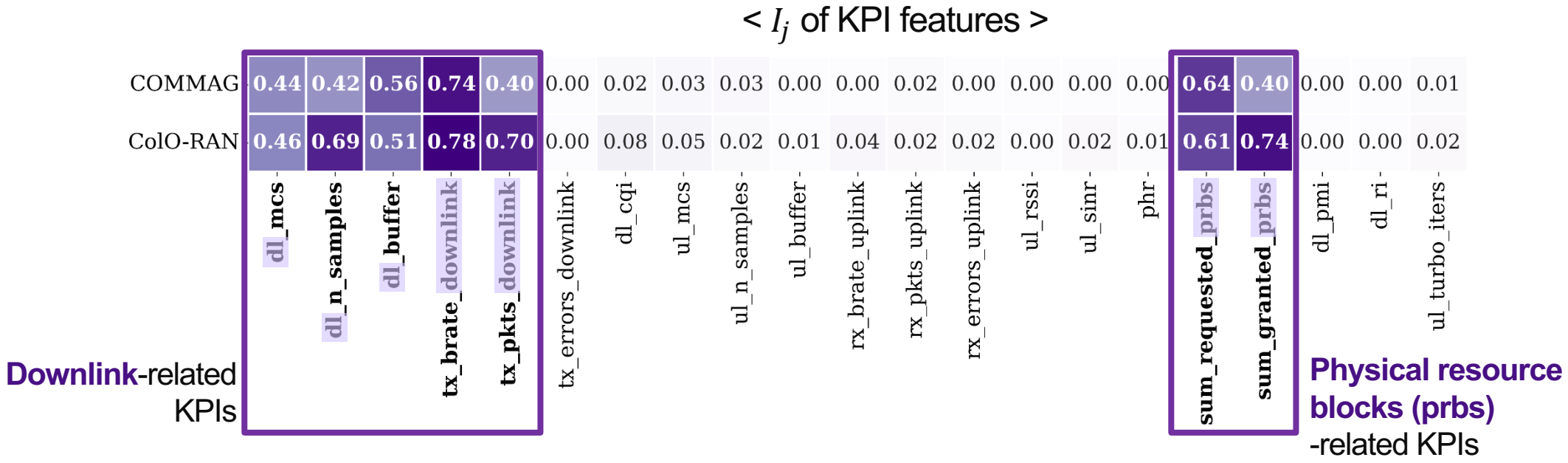
- KPI feature importance based on mutual information



- Highly sparse distribution
  - Only 7 KPI features showing strong dependencies
  - Consistent across two datasets collected in different experimental configurations

Network Data Analysis

- KPI feature importance based on mutual information



- Dominance of downlink/resource-related features
  - Inherent asymmetry of mobile traffic which is downlink-heavy

A few KPIs include (nearly) all information for classification

Network Data Analysis

- Slice class separability based on generalized Dunn's index
  - Goal: measuring how clearly slice classes can be distinguished in feature space
  - Generalized Dunn's index DI
    - Separability in m classes

$$DI = \frac{\min\limits_{1 \le g < h \le m} \delta(C_g, C_h)}{\max\limits_{1 \le l \le m} \Delta(C_l)}$$

- Inter-class distance

$$\delta(C_g, C_h) = \mathrm{dist}(\boldsymbol{\mu}_{C_g}, \boldsymbol{\mu}_{C_h})$$

- Intra-class distance

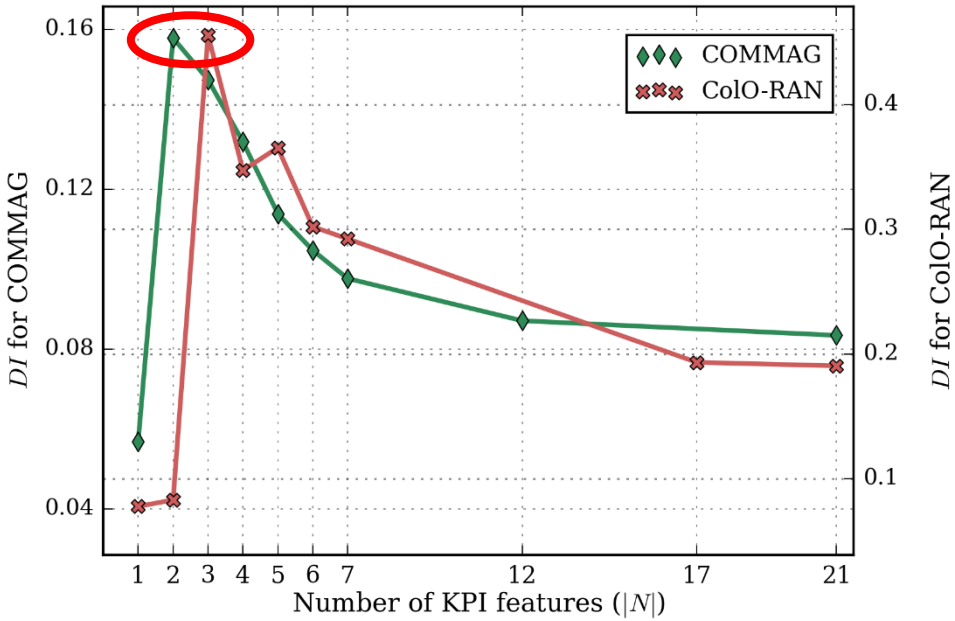
$$\Delta(C_l) = 2 \cdot \sum_{\mathbf{x}_i \in C_l} \frac{\mathrm{dist}(\mathbf{x}_i, \boldsymbol{\mu}_{C_l})}{|C_l|}$$

- C : cluster
- $\boldsymbol{\mu}_C$: centroids of cluster C
- $\mathrm{dist}(\cdot,\cdot)$: Mahalanobis distance

    - Centroid-based distance: robust against outliers
    - Mahalanobis distance: separation based on the covariance structure of the data
  - Meaning: ratio of minimum inter-class distance to maximum intra-class distance
  - Higher DI → Better slice class separability

Network Data Analysis

- Slice class separability based on generalized Dunn's index
  - Ranking features based on I_j
  - DI results with the number of KPI features ($|N|$)
    - Peak DI at small feature subset
      - A compact subset of KPIs yields the best class separability.
      - $|N| = 2$ in COMMAG, $|N| = 3$ in ColO-RAN
    - Decreasing DI as more features are added
      - Adding less-informative KPIs introduces noise and reduces separability.



High-dimensional feature space is unnecessary to classify network slice!

Experiments

- Experimental setup
  - Goal: comparing performance and RIC operation efficiency of ML and DL slice classifiers
  - KPI telemetry dataset: COMMAG, ColO-RAN
    - 68,163 and 78,702 instances in COMMAG and ColO-RAN, respectively
    - Splitting into training, validation, and testing sets in a ratio of 60%, 15%, 25%
  - Slice classifier $\mathcal{S}_w(\cdot)$
    - ML models: eXtreme Gradient Boosting (XGBoost), Support Vector Machine (SVM), k-Nearest Neighbors (k-NN)
    - DL models: Gated Recurrent Unit (GRU), Transformer, Convolutional Neural Network (CNN)

Experiments

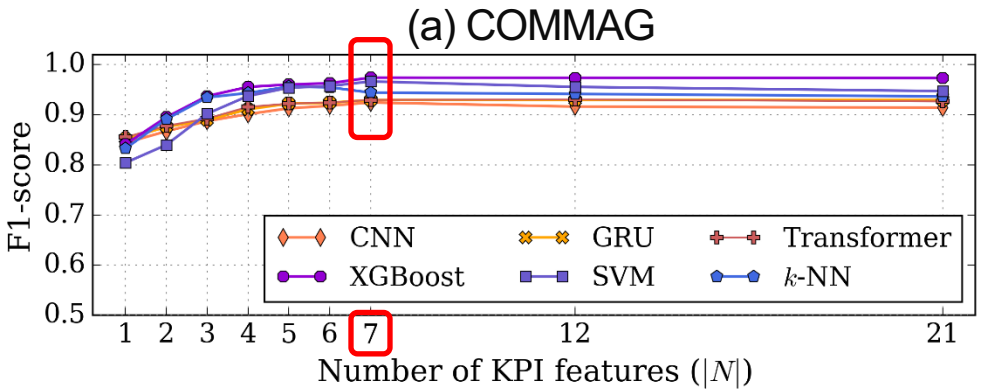
- Experimental setup
  - Classifier hyperparameter settings

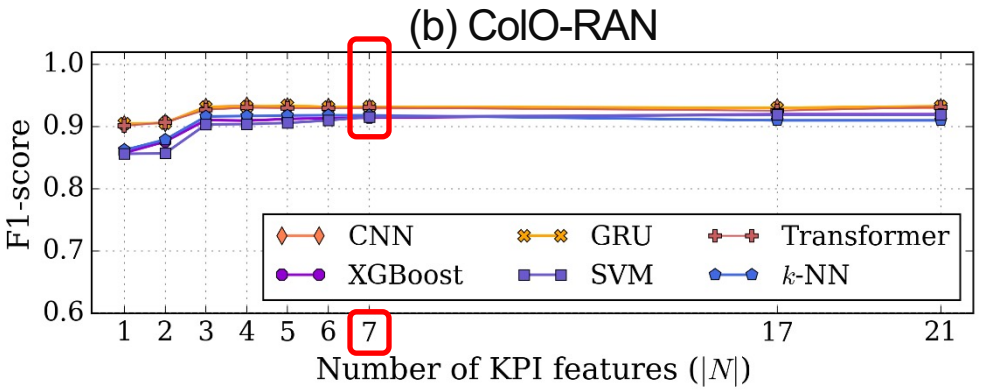
ML	XGBoost	Early stopping
	SVM	RBF kernel
	k-NN	k=5
DL	GRU	Early stopping, Sliding window (length 10 and stride 1), Learning rate 0.001, ReduceLROnPlateau learning rate scheduler, 3 layers, Hidden size 128, Adam optimizer, Batch size 64, 1 linear layer
	Transformer	Early stopping, Sliding window (length 10 and stride 1), Positional encoding, Learning rate 0.0001, LinearWarmupCosineAnnealing learning rate scheduler, Learning rate decay factor 0.01, 3 layers, Hidden size 65, Adam optimizer, Batch size 64, 1 linear layer,
	CNN	Early stopping, Sliding window (length 10 and stride 1), Conv2D layers with 4x1 kernel, Learning rate 0.001, ReduceLROnPlateau learning rate scheduler, 1 layer, Adam optimizer, Batch size 128, 2 linear layer

- ML models: minimal settings with default parameters
- DL models: extensive hyperparameter tuning

Experiments

- Slice classification performance





- Overall high performance: all classifiers achieves high F1-scores with a few KPI features.
- Impact of feature number $|N|$: performance improves rapidly up to 7 features, then saturates with marginal gains beyond that point.
- ML vs. DL classifiers: DLs show slightly higher performance on ColO-RAN, while ML models remain highly competitive on COMMAG.

Experiments

- Slice classification performance
  - Average F1-socres (mean±std) of MLs and DLs
    - Largest ML-DL performance gap
      - COMMAG: 0.034 (ML>DL)
      - ColO-RAN: 0.020 (ML<DL)
    - Smaller performance gap in ColO-RAN than in COMMAG

		ML classifiers	DL classifiers
COMMAG	$\|N\| = 21$	**0.952** ± 0.019	0.923 ± 0.008
	$\|N\| = 7$	**0.962** ± 0.016	0.928 ± 0.003
	$\|N\| = 2$	**0.875** ± 0.031	0.874 ± 0.006
ColO-RAN	$\|N\| = 21$	0.917 ± 0.005	**0.932** ± 0.001
	$\|N\| = 7$	0.916 ± 0.002	**0.931** ± 0.001
	$\|N\| = 3$	0.910 ± 0.006	**0.930** ± 0.003

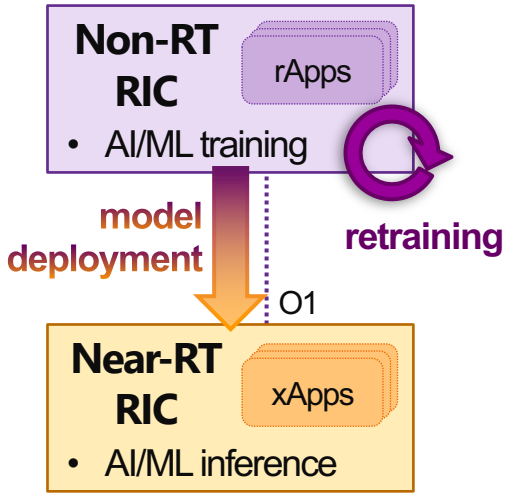
❖ **Key implication**

- DL classifiers are more sensitive to data structure, requiring careful tuning for each data.
- Due to concise parameter settings, ML classifiers can provide practicality when training time and retraining overhead are critical.

Experiments

- Efficiency in RIC operation
  - Training time (sec)
    - Significant gap between ML and DL classifiers

		COMMAG		ColO-RAN	
		$\|N\| = 7$	$\|N\| = 21$	$\|N\| = 7$	$\|N\| = 21$
ML	XGBoost	334ms	757ms	197ms	546ms
	SVM	7s	14s	11s	15s
	k-NN	**28ms**	**2ms**	**33ms**	**2ms**
DL	GRU	12m 53s	15m 21s	8m 50s	16m 18s
	Transformer	32m 34s	31m 4s	14m 59s	18m 50s
	CNN	20m 3s	18m 33s	7m 54s	7m 50s



❖ **Implication for RIC operation**

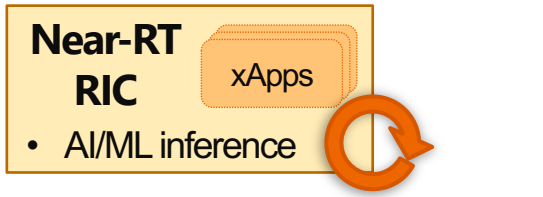
- Frequent retraining in non-RT RIC favors lightweight MLs.
- Reduced retraining overhead causes faster model updates and deployment.

Experiments

- Efficiency in RIC operation
  - Inference time (mean $\pm$ std) (μsec)

		COMMAG		ColO-RAN	
		$\|N\| = 7$	$\|N\| = 21$	$\|N\| = 7$	$\|N\| = 21$
ML	XGBoost	1447 ± 18	3175 ± 403	1472 ± 24	3223 ± 421
	SVM	287 ± 4	441 ± 6	313 ± 5	378 ± 4
	k-NN	592 ± 14	385 ± 402	594 ± 24	364 ± 218
DL	GRU	620 ± 6	657 ± 78	632 ± 20	615 ± 38
	Transformer	867 ± 99	793 ± 56	702 ± 8	723 ± 51
	CNN	79 ± 2	139 ± 13	81 ± 3	142 ± 14

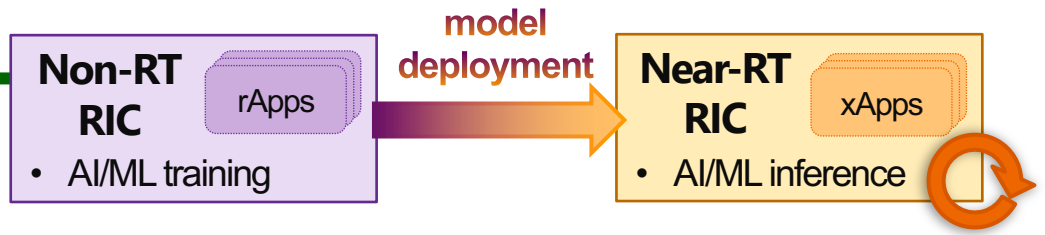
- All classifiers achieve similar inference latency.
- Compared to near-RT RIC maximum time constraint of 1 second, all operate less than 0.2% of the limit.



❖ Implication for near-RT RIC

- Low-latency requirement is satisfied by both ML and DL.
- Inference speed is not the main differentiator in deployment.
  - Near-RT RIC can provide sufficient computational resources.

Experiments



- Efficiency in RIC operation
  - Memory usage and trained model file size

(a) Memory usage (mean ± std) (MB)

		COMMAG		ColO-RAN	
		$\|N\| = 7$	$\|N\| = 21$	$\|N\| = 7$	$\|N\| = 21$
ML	XGBoost	277 ± 0.04	269 ± 0.09	361 ± 0.09	351 ± 0.08
	SVM	299 ± 0.11	295 ± 0.00	372 ± 0.01	384 ± 0.06
	k-NN	294 ± 0.03	315 ± 0.12	414 ± 0.04	398 ± 0.05
DL	GRU	300 ± 0.02	242 ± 1.12	316 ± 0.14	250 ± 0.05
	Transformer	323 ± 0.06	298 ± 0.09	337 ± 0.05	307 ± 0.09
	CNN	329 ± 0.01	317 ± 0.00	326 ± 0.14	339 ± 0.02

(b) Trained model file size (mean ± std) (MB)

		COMMAG		ColO-RAN	
		$\|N\| = 7$	$\|N\| = 21$	$\|N\| = 7$	$\|N\| = 21$
ML	XGBoost	2.517	3.554	0.561	2.073
	SVM	0.655	2.363	0.749	1.936
	k-NN	6.107	8.009	7.007	9.248
DL	GRU	0.962	0.982	0.962	0.982
	Transformer	15.687	15.690	15.687	15.690
	CNN	1.925	5.753	1.925	5.753

- Both ML and DL classifiers yield small memory usage.
- Trained model sizes are trivial compared to RIC server capacity.

❖ **Implication for deployment**

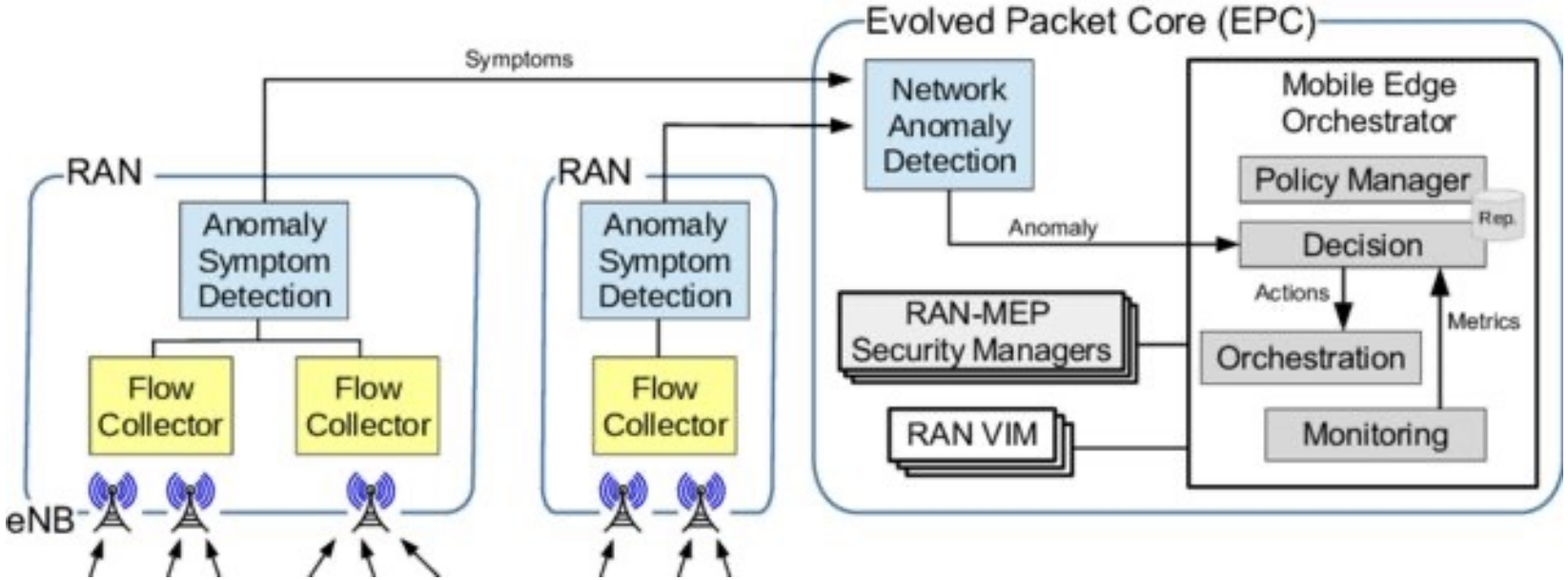
- Key point of RIC operation is based on training time of classifiers!

Takeaway from use case 1

- Study data analytics approach and reevaluation of ML algorithms for efficient operation of slice classification in O-RAN architectures.
- A small subset of KPIs can suffice for discriminative capability in slice classification.
- ML classifiers can achieve competitive performance while substantially reducing training time.

Use Case 2 – Network Traffic Anomaly Detection

- Goal: ML/DL-based network traffic anomaly detection
  - As efficient architecture as possible
- Challenges
  - High computational complexity
  - Often tabular (structured) data
- Solutions
  - Network data feature grouping or engineering via feature analysis



[L. Fernández-Maimó, et. al., "Dynamic management of a deep learning-based anomaly detection system for 5G networks," Journal of Ambient Intelligence and Humanized Computing, 2019]

Challenges

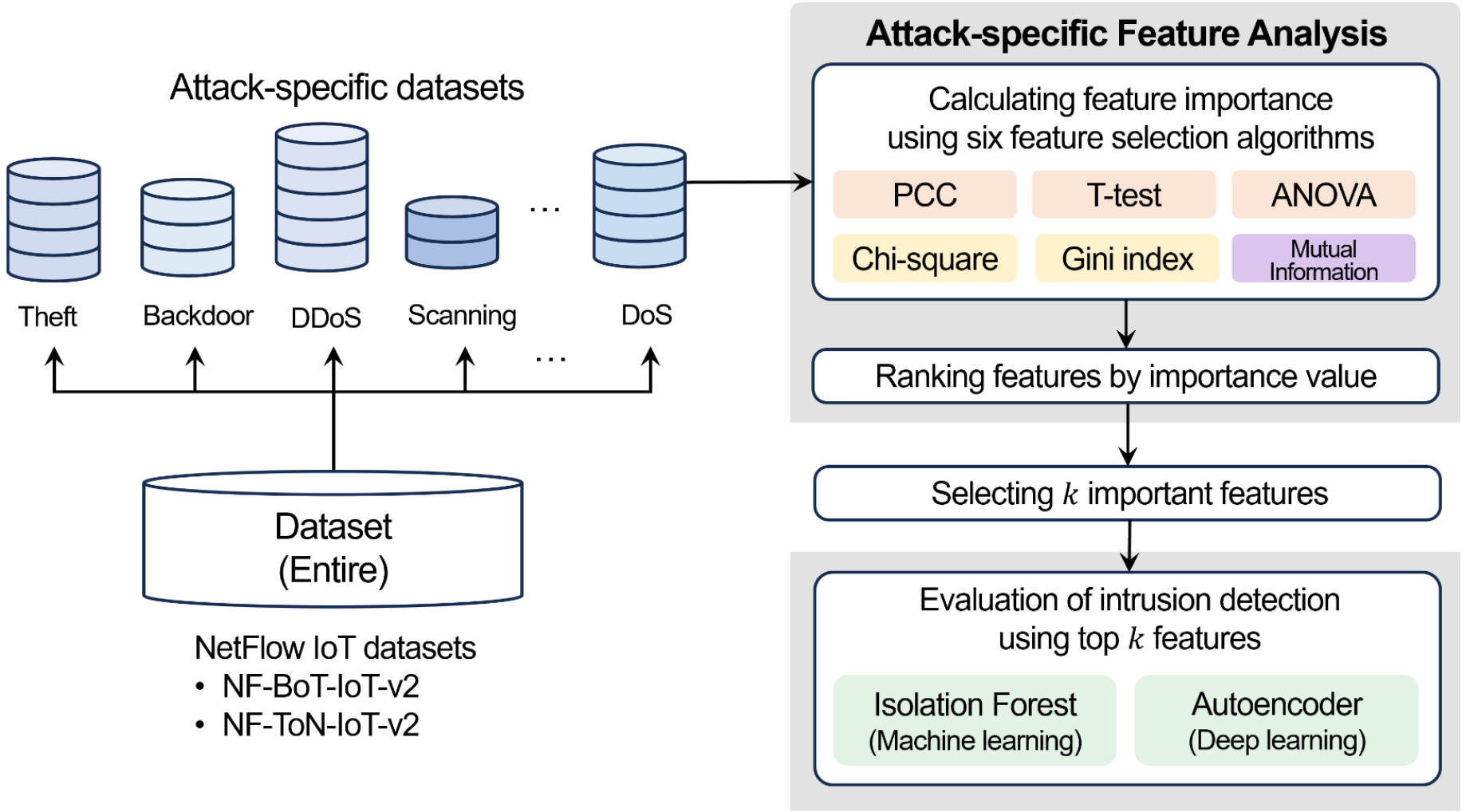
- Challenges of dimensionality reduction on network traffic data
  - Considering network traffic data is class-imbalanced tabular (structured) data
  - Removing features with high redundancy or lack or relevance among them
  - Maintaining performance equivalent to that of using all data before dimensionality reduction

➔ Analyzing network traffic data and features based on the types of attacks

➔ Grouping and selecting important features via attack-specific feature analysis

Attack-specific Feature Analysis Framework

- Overview of attack-specific feature analysis framework



[D. Choi, J. Rheey and H. Park, "Attack-Specific Feature Analysis Framework for NetFlow IoT Datasets," *Computers & Security*, 2025]

Attack-specific Feature Analysis Framework

- IoT network traffic dataset
  - NetFlow[1],[2]-based standard feature set (39 features)
    - Most widely used standard feature set with a variety of data features and bi-directional flow support
    - An industry-standard protocol for network traffic collection and network information analysis

	Feature	Description
f1	PROTOCOL	IP protocol identifier byte
f2	L7_PROTO	Layer 7 protocol (numeric)
f3	IN_BYTES	Incoming number of bytes
f4	OUT_BYTES	Outgoing number of bytes
f5	IN_PKTS	Incoming number of packets
f6	OUT_PKTS	Outgoing number of packets
f7	FLOW_DURATION_MILLISECONDS	Flow duration in milliseconds
f8	TCP_FLAGS	Cumulative of all TCP flags
f9	CLIENT_TCP_FLAGS	Cumulative of all client TCP flags
f10	SERVER_TCP_FLAGS	Cumulative of all server TCP flags
f11	DURATION_IN	Client to Server stream duration (msec)
f12	DURATION_OUT	Client to Server stream duration (msec)
f13	MIN_TTL	Min flow TTL
f14	MAX_TTL	Max flow TTL
f15	LONGEST_FLOW_PKT	Longest packet (bytes) of the flow
f16	SHORTEST_FLOW_PKT	Shortest packet (bytes) of the flow
f17	MIN_IP_PKT_LEN	Len of the smallest flow IP packet observed
f18	MAX_IP_PKT_LEN	Len of the largest flow IP packet observed
f19	SRC_TO_DST_SECOND_BYTES	Src to dst Bytes/sec
f20	DST_TO_SRC_SECOND_BYTES	Dst to src Bytes/sec

f21	RETRANSMITTED_IN_BYTES	Number of retransmitted TCP flow bytes (src->dst)
f22	RETRANSMITTED_IN_PKTS	Number of retransmitted TCP flow packets (src->dst)
f23	RETRANSMITTED_OUT_BYTES	Number of retransmitted TCP flow bytes (dst->src)
f24	RETRANSMITTED_OUT_PKTS	Number of retransmitted TCP flow packets (dst->src)
f25	SRC_TO_DST_AVG_THROUGHPUT	Src to dst average thpt (bps)
f26	DST_TO_SRC_AVG_THROUGHPUT	Dst to src average thpt (bps)
f27	NUM_PKTS_UP_TO_128_BYTES	Packets whose IP size <= 128
f28	NUM_PKTS_128_TO_256_BYTES	Packets whose IP size > 128 and <= 256
f29	NUM_PKTS_256_TO_512_BYTES	Packets whose IP size > 256 and <= 512
f30	NUM_PKTS_512_TO_1024_BYTES	Packets whose IP size > 512 and <= 1024
f31	NUM_PKTS_1024_TO_1514_BYTES	Packets whose IP size >1024 and <= 1514
f32	TCP_WIN_MAX_IN	Max TCP Window (src->dst)
f33	TCP_WIN_MAX_OUT	Max TCP Window (dst->src)
f34	ICMP_TYPE	ICMP Type * 256 + ICMP code
f35	ICMP_IPV4_TYPE	ICMP Type
f36	DNS_QUERY_ID	DNS query transaction Id
f37	DNS_QUERY_TYPE	DNS query type (e.g. 1=A, 2=NS..)
f38	DNS_TTL_ANSWER	TTL of the first A record (if any)
f39	FTP_COMMAND_RET_CODE	FTP client command return code

[1] I. Cisco, NetFlow Version 9 Flow-Record Format, White Paper, Feb (2007).
[2] B. Claise, Cisco Systems NetFlow Services Export Version 9, RFC 3954,2004. URL: https://www.rfc-editor.org/info/rfc3954.

Attack-specific Feature Analysis Framework

- IoT network traffic dataset : *NF-BoT-IoT-v2, NF-ToN- IoT-v2*
  - All existing IoT datasets provided in NetFlow for IDS [M. Sarhan, et al., "Towards a Standard Feature Set for Network Intrusion Detection System Datasets," *Mobile Networks and Applications*, 2022]
    a. NF-BoT-IoT-v2 : 4 attack types (DoS, DDoS, Reconnaissance, Theft)
    b. NF-ToN-IoT-v2 : 9 attack types (Backdoor, DoS, DDoS, Injection, MITM (Man-in-the-Middle), Password, Ransomware, Scanning, XSS(Cross-Site Scripting))
  - Generated from the publicly available packet capture files (pcap) of the widely used IoT datasets
  - Contains network traffic flows in IoT collected over realistic network environments and attack scenarios

Attack-specific Feature Analysis Framework

- IoT network traffic dataset : *NF-BoT-IoT-v2, NF-ToN- IoT-v2*
  - *NF-BoT-IoT-v2*
    - Generated from *Bot-IoT* dataset [N. Koroniotis, et al., "Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-IoT dataset," *Future Generation Computer Systems*, 2019.]
    - Total number of traffic flows: 37,763,497 (37,628,460 (99.64%) attack traffic samples and 135,037 (0.36%) benign traffic samples)

Class	Number of samples	Descriptions
Benign	13,859	Normal unmalicious flows
DoS	16,673,183	An attempt to overload a computer system's resources to prevent access to or availability of its data
DDoS	18,331,847	An attempt similar to DoS but having multiple different distributed sources
Reconnaissance	2,620,999	A technique also known as a probe, for gathering information about a network host
Theft	2,431	A group of attacks that aims to obtain sensitive data such as data theft and keylogging

Attack-specific Feature Analysis Framework

- IoT network traffic dataset : *NF-BoT-IoT-v2, NF-ToN- IoT-v2*
  - *NF-ToN-IoT-v2*
    - Generated from *TON-IoT* dataset. [N. Moustafa, "A new distributed architecture for evaluating AI-based security systems at the edge: Network TON IoT datasets," *Sustainable Cities and Society*, 2021]
    - Total number of data flows: 16,940,496 (10,841,027 (63.99%) attack traffic samples and 6,099,469 (36.01%) benign samples)

Class	Number of samples	Descriptions
Benign	6,099,469	Normal unmalicious flows
Scanning	3,781,419	Techniques that aim to discover information about networks and hosts, also known as probing
XSS	2,455,020	A type of injection in which an attacker uses web applications to send malicious scripts to end-users
Password	1,153,323	A variety of attacks aimed at retrieving passwords through brute force or sniffing
Injection	684,465	A variety of attacks that supply untrusted inputs, aiming to alter the course of execution, with SQL and code injections being two of the main ones
DDoS	2,026,234	An attempt similar to DoS but having multiple different distributed sources
DoS	712,609	An attempt to overload a computer system's resources to prevent access to or availability of its data
Ransomware	3,425	An attack that encrypts the files stored on a host and demands for compensation in exchange for the decryption key
Backdoor	16,809	A technique that aims to attack remote-access computers by replying to specific constructed client applications

Attack-specific Feature Analysis Framework

- Feature selection algorithms for measuring the feature importance
  - Feature selection algorithms based on filter approaches
    - Univariate methods can measure the correlation between each feature and label(normal/abnormal) feature.
    - These methods solely consider feature values without the intervention of a model.

Pearson Correlation Coefficient (PCC)

- **PCC measures linear correlation between two random variables**

$$\rho = \frac{\mathbb{E}[(\boldsymbol{X} - \mu_X)(\boldsymbol{Y} - \mu_Y)]}{\sigma_X \sigma_Y}$$

Analysis of Variance (ANOVA)

- **ANOVA is used to test the difference between two or more means**

$$F = \frac{\sum_{k=1}^{\mathcal{C}} n_k (\mu_{i,k} - \mu_i)^2}{\sum_{k=1}^{\mathcal{C}} n_k \ \mathrm{var}_{i,k}{}^2}$$

T-test

- **T-test can be utilized to decide whether the means for two sets are not the same**

$$T = \frac{|\mu_1 - \mu_2|}{\sqrt{\left(\frac{\mathrm{var}_{i,1}{}^2}{n_1}\right) + \left(\frac{\mathrm{var}_{i,2}{}^2}{n_2}\right)}}$$

Chi-Square

- **Chi-square is used to evaluate the independence between two events**

$$\mathcal{X}^2 = \sum_{i=1}^{\mathcal{I}} \sum_{j=1}^{\mathcal{C}} \frac{(O_{i,j} - E_{i,j})^2}{E_{i,j}}$$

Mutual information

- **Mutual information measures the mutual dependence between two random variables**

$$I = \sum_{y \in \boldsymbol{Y}} \sum_{x \in \boldsymbol{X}} p_{XY}(x,y) \, log\left(\frac{p_{XY}(x,y)}{p_X(x) p_Y(y)}\right)$$

Gini-index

- **Gini-index quantifies if the feature is able to separate instances from different class**

$$G = \min_{W} p(W)\left(1 - \sum_{s=1}^{c} p(C_s|W)^2 + P(\bar{W})\left(1 - \sum_{s=1}^{c} p(C_s|W)^2\right)\right)$$

[D. Choi, J. Rheey and H. Park, "A Study on Performance Improvement of Network Traffic Anomaly Detection via Progressive Feature Addition," *2023 Winter Conference on KICS*, Jan. 2023.] (Best Paper Award)

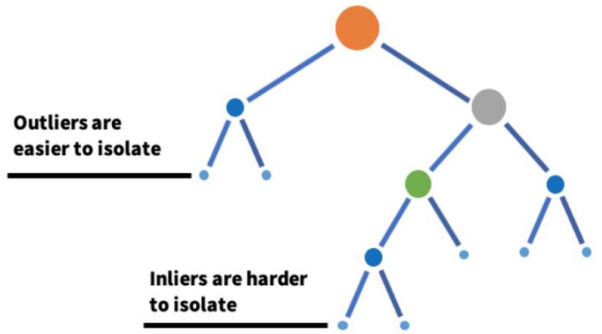
Attack-specific Feature Analysis Framework

- Evaluation of IDS
  - Reduce the dimension of the dataset including only k important features
  - Design an attack-specific anomaly detector with the reduced dataset using unsupervised learning-based IDS
    - Isolation Forest: an ML algorithm for anomaly detection
    - Autoencoder: commonly used DL architecture as an anomaly detector
  - Compare anomaly detection performance via progressive feature addition

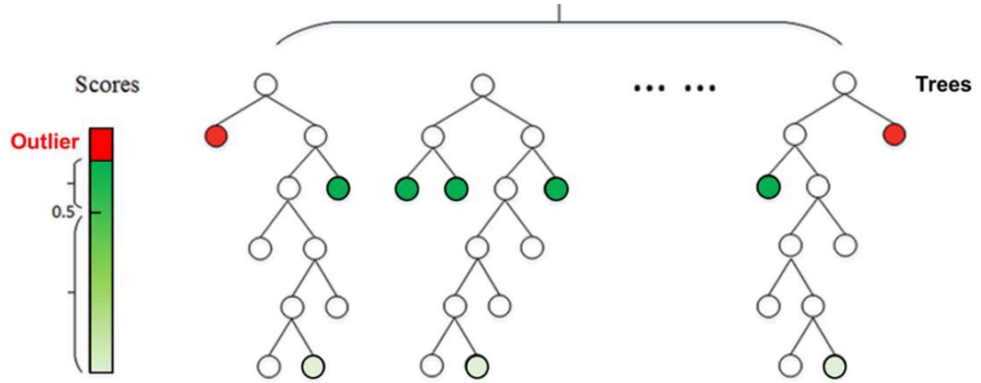
Recall: Isolation Forest

- Concept of Isolation and Tree
  - Anomalies are ‘few and different’, which make them more susceptible to isolation than normal points.
  - Tree structure can be constructed effectively to isolate every single instance.
- Isolation Tree
  - Anomalies are isolated closer to the root of the tree.
  - Normal points are isolated at the deeper end of the tree.
- Isolation Forest (iForest)
  - iForest builds and Ensemble of iTrees.
  - Anomalies have short average path lengths on the iTrees.

<Isolation Tree>



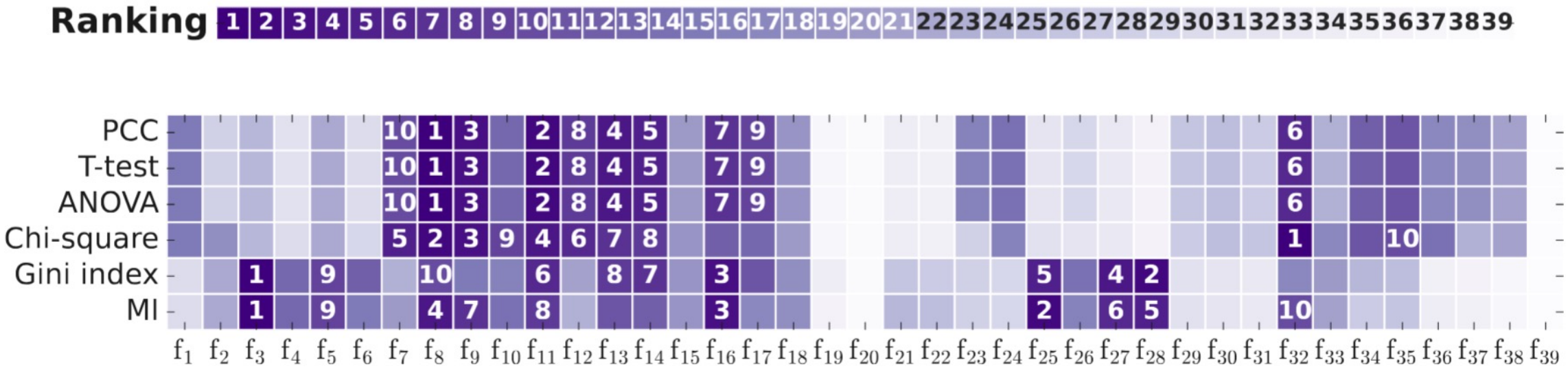
<Isolation Forest>



[F. T. Liu, K. M. Ting, Z.-H Zhou,, “Isolation Forest,” *The 8th International Conference on Data Mining*, 2008]

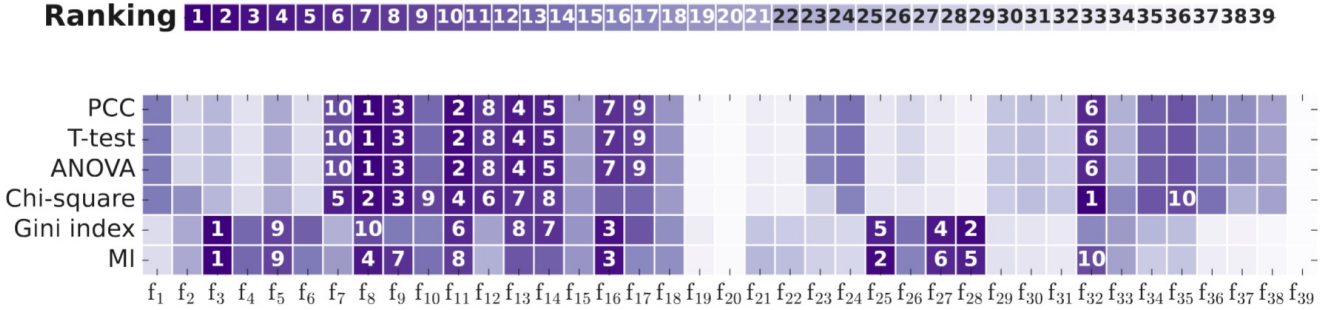
Feature Analysis Results

- Attack-specific feature analysis
  - Heatmaps present the attack-specific feature rankings for individual feature and feature selection algorithms.
    - The darker the color, the higher the rank

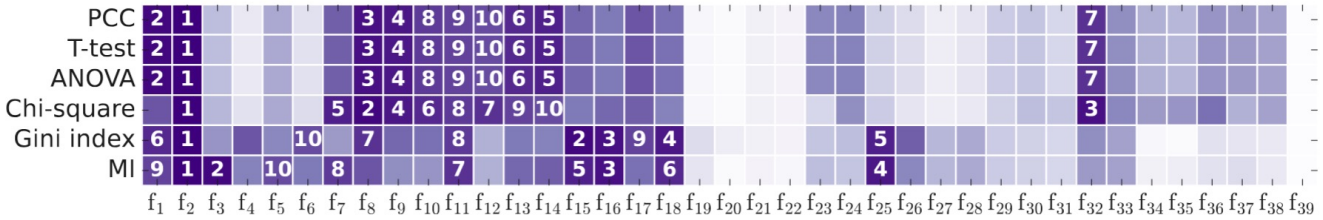


Feature Analysis Results

- Attack-specific feature analysis
  - *NF-BoT-IoT-v2*
    - DoS
      - eight of top-10 feature combinations in PCC, T-test, ANOVA, and Chi-square share a common set of features
      - relate to statistics of network flow duration, TTL (Time To Live), and TCP
    - DDoS: The top-ranked feature, *L7 PROTO* (f_2), is consistent across all feature selection algorithms.



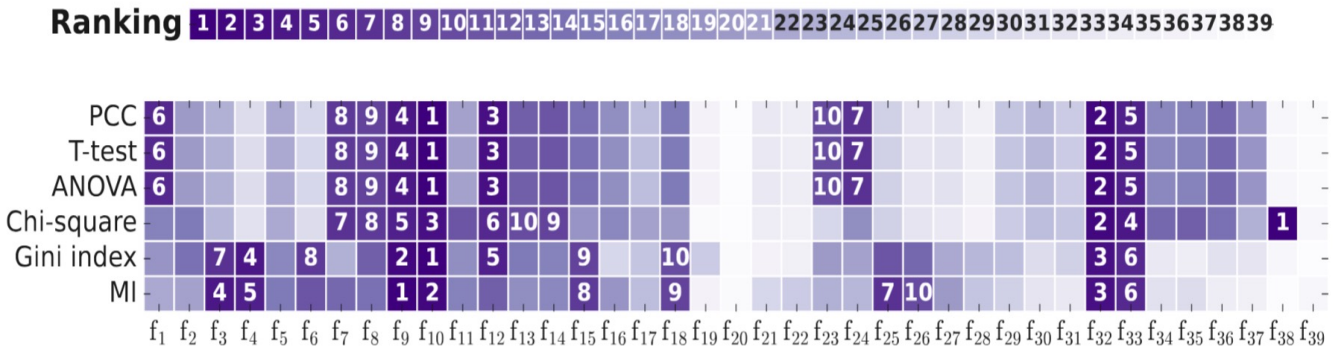
(a) DoS



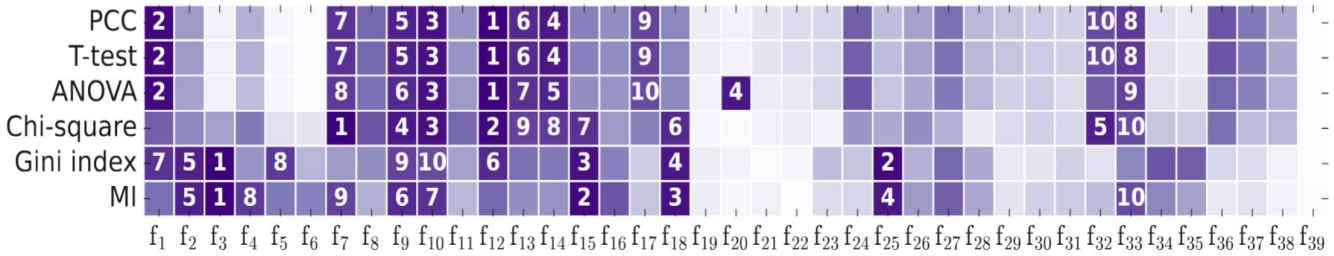
(b) DDoS

Feature Analysis Results

- Attack-specific feature analysis
  - *NF-BoT-IoT-v2*
    - Reconnaissance:
      - Top-5 feature combinations for all feature selection algorithms include *CLIENT TCP FLAGS* (f_9), SERVER TCP FLAGS(f_{10}), and TCP WIN MAX IN (f_{32}), which are oriented from TCP.
    - Theft:
      - PCC, T-test, and ANOVA are also of equivalent importance order across all features (except ANOVA ranks DST TO SRC SECOND BYTES (f_{20}) as a fourth important feature)



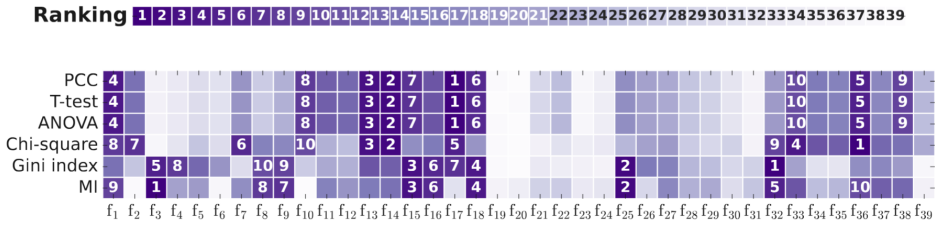
(c) Reconnaissance



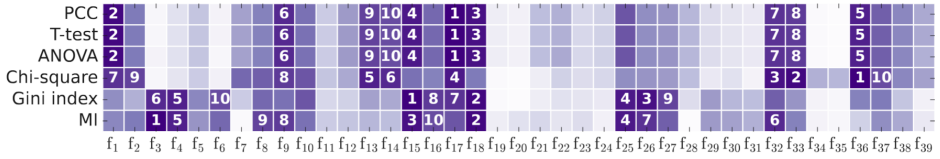
(d) Theft

Feature Analysis Results

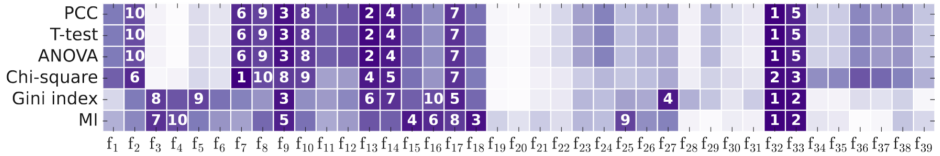
- Attack-specific feature analysis
  - *NF-ToN-IoT-v2*



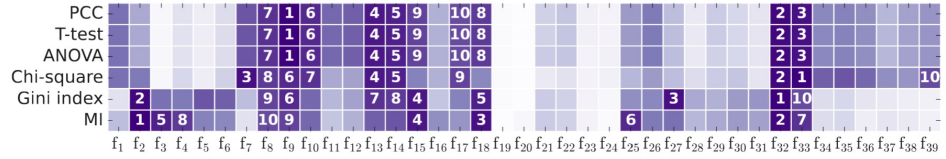
(a) Scanning



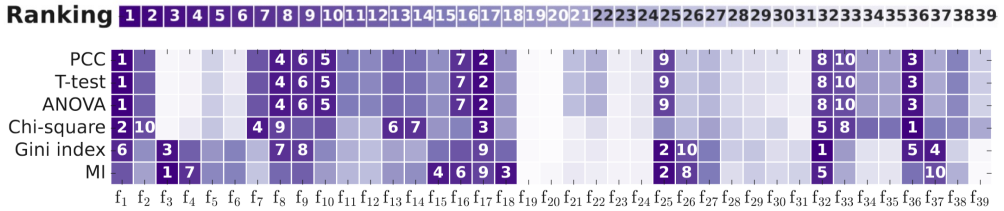
(b) XSS



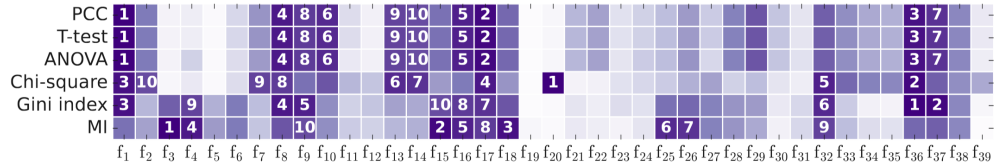
(c) DDoS



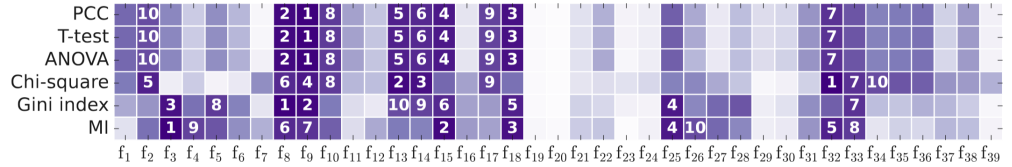
(d) Password



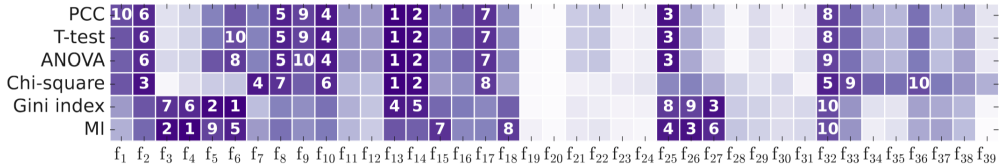
(a) DoS



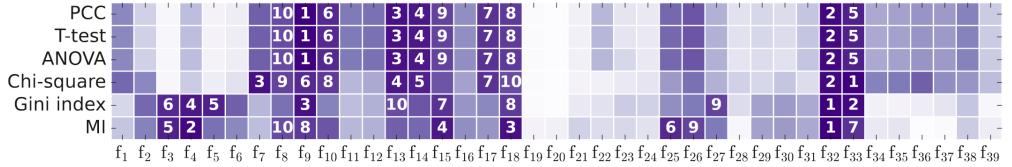
(b) MITM



(c) Ransomware



(d) Backdoor

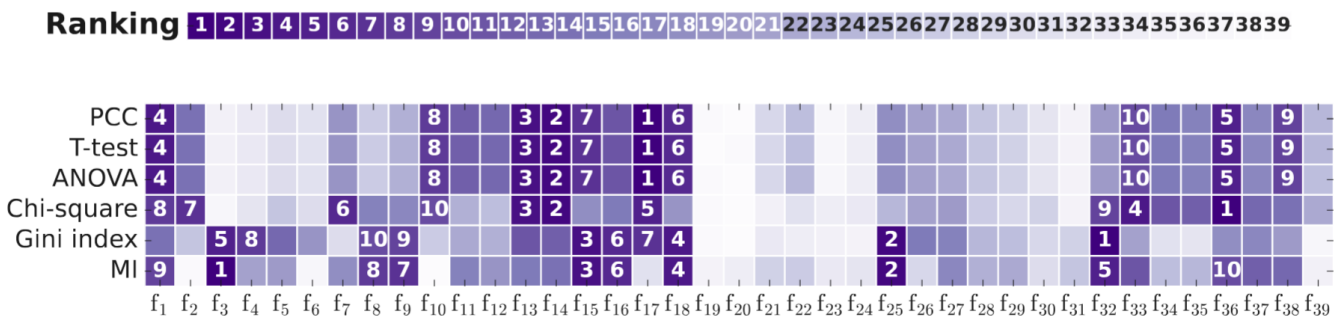


(e) Injection

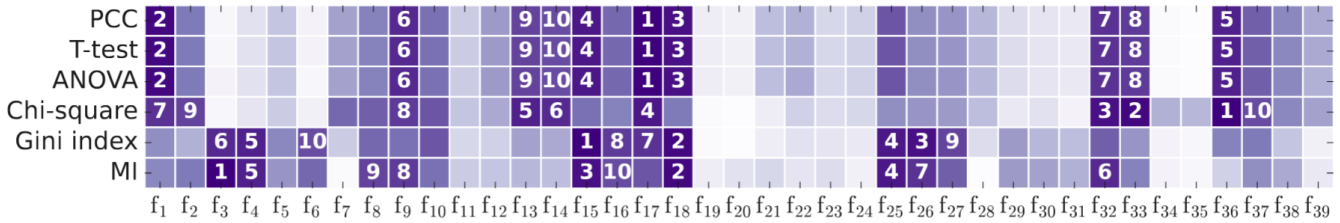
Feature Analysis Results

- Attack-specific feature analysis
  - *NF-ToN-IoT-v2*
    - Similar ranking is observed among feature selection algorithms as well as attack types.
    - Scanning and XSS attacks
      - highest-ranked features in all feature selection algorithms are exactly the same as each other (except Gini index).

		PCC	T-test	ANOVA	Chi-square	Gini index	MI
Top-1	Scanning XSS	MIN_IP_PKT_LEN (f_{17})			DNS_ID_QUERY (f_{36})	TCP_WIN_MAX_IN (f_{32}) LONGEST_FLOW_PKT (f_{15})	IN_BYTES (f_3)



(a) Scanning

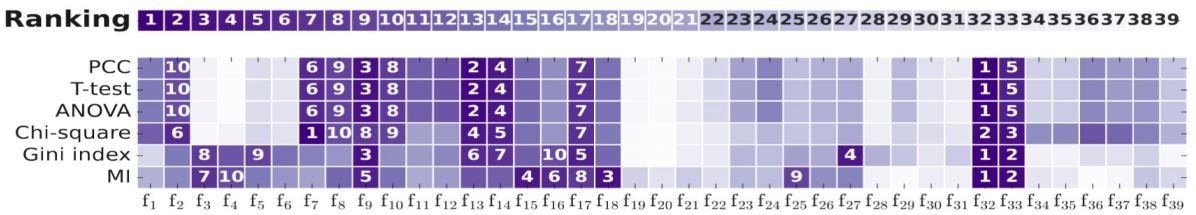


(b) XSS

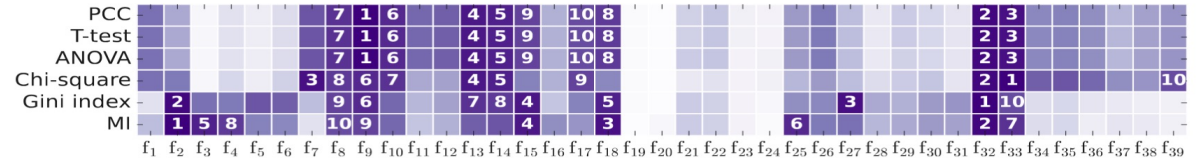
Feature Analysis Results

- Attack-specific feature analysis
  - *NF-ToN-IoT-v2*
    - DDoS, Password, Injection attacks
      - TCP-related features such as *CLIENT TCP FLAGS* (f9), *TCP WIN MAX IN* (f32), and *TCP MAX OUT* (f33), are of considerable impact.

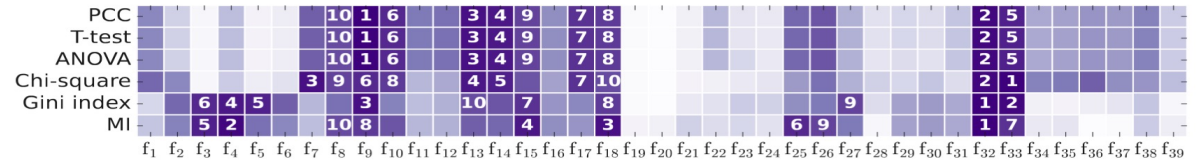
		PCC	T-test	ANOVA	Chi-square	Gini index	MI
DDoS	Top-1	TCP_WIN_MAX_IN (f_{32})			FLOW_DURATION_MILLISECONDS (f_7)	TCP_WIN_MAX_IN (f_{32})	
	Top-5	f_{13}, f_{14}, f_{32}, f_{33}				f_9, f_{32}, f_{33}	
	Top-10	f_2, f_7, f_8, f_9, f_{10}, f_{13}, f_{14}, f_{17}, f_{32}, f_{33}				f_3, f_9, f_{16}, f_{17}, f_{32}, f_{33}	
Password	Top-1	CLIENT_TCP_FLAGS (f_9)		TCP_WIN_MAX_OUT (f_{33})		TCP_WIN_MAX_IN (f_{32})	L7_PROTO (f_2)
	Top-5	f_{13}, f_{14}, f_{32}, f_{33}				f_2, f_3, f_{15}, f_{18}	
	Top-10	f_8, f_9, f_{10}, f_{13}, f_{14}, f_{17}, f_{32}, f_{33}				f_2, f_8, f_9, f_{15}, f_{18}, f_{32}, f_{33}	
Injection	Top-1	CLIENT_TCP_FLAGS (f_9)		TCP_WIN_MAX_OUT (f_{33})		TCP_WIN_MAX_IN (f_{32})	
	Top-5	f_{13}, f_{14}, f_{32}, f_{33}				f_4, f_{32}	
	Top-10	f_8, f_9, f_{10}, f_{13}, f_{14}, f_{17}, f_{18}, f_{32}, f_{33}				f_3, f_4, f_8, f_9, f_{15}, f_{18}, f_{32}, f_{33}	



(c) DDoS



(d) Password



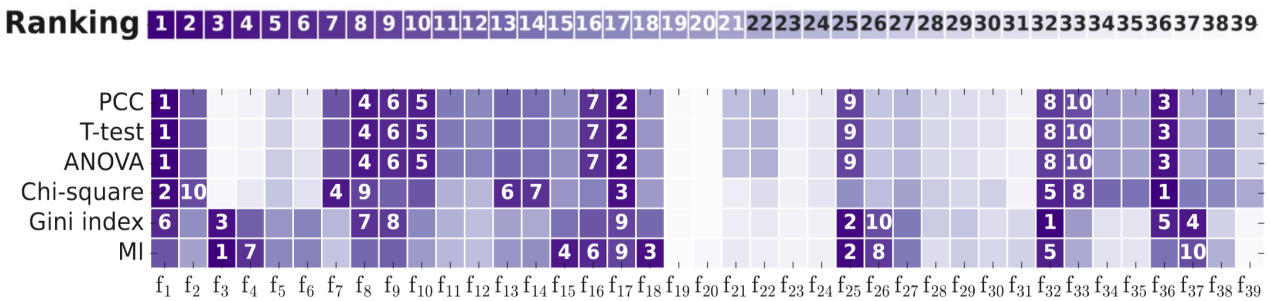
(e) Injection

Feature Analysis Results

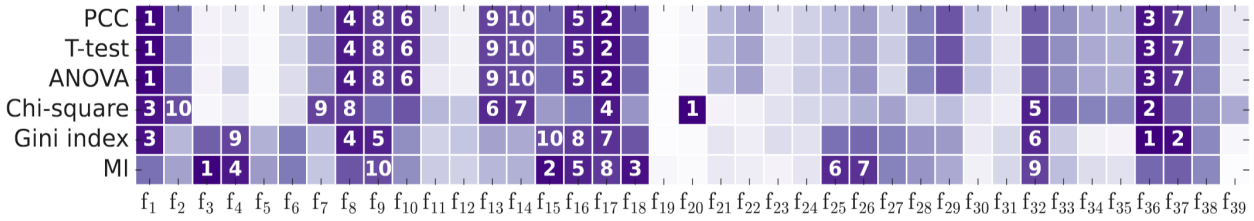
- Attack-specific feature analysis
  - *NF-ToN-IoT-v2*

	PCC T-test ANOVA	Chi-square	Gini index	MI
Top-10	$f_1, f_8, f_9, f_{10}, f_{16}, f_{17}, f_{36}$	$f_1, f_2, f_7, f_8, f_{13}, f_{14}, f_{17}, f_{32}, f_{36}$	$f_1, f_8, f_9, f_{17}, f_{32}, f_{36}, f_{37}$	$f_3, f_4, f_{15}, f_{16}, f_{17}, f_{18}, f_{25}, f_{26}, f_{32}$

- DoS and MITM attacks
  - Similarities in top-10 combinations are found between the two attacks when the same feature selection algorithms are employed.



(a) DoS

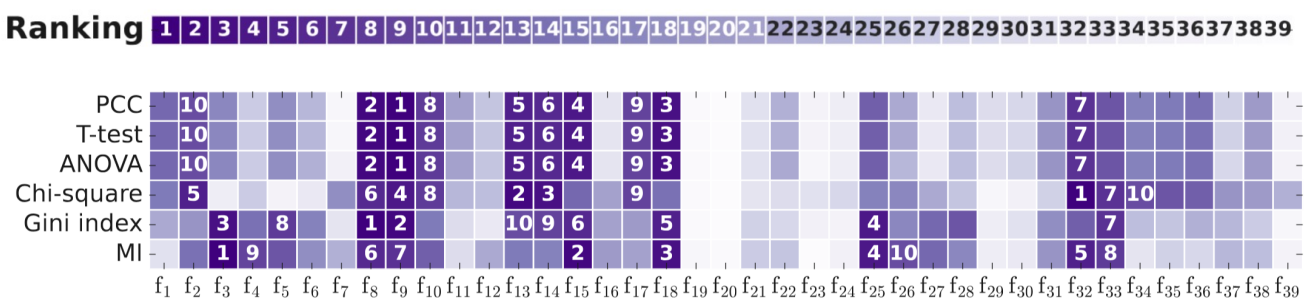


(b) MITM

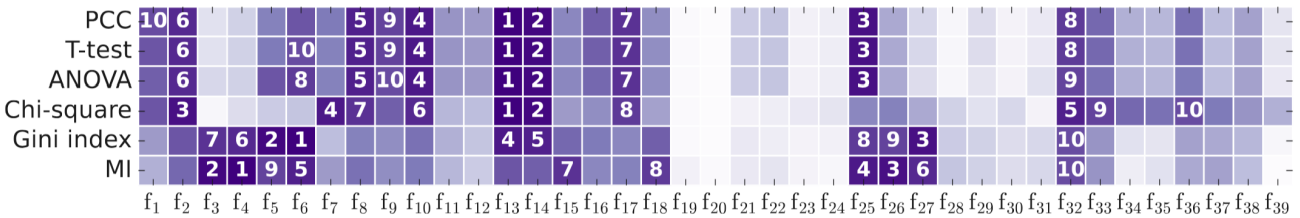
Feature Analysis Results

- Attack-specific feature analysis
  - *NF-ToN-IoT-v2*
    - Ransomware attack
      - TCP-related features tend to be the top-ranked features
    - Backdoor attack
      - feature rankings are inconsistent in PCC, T-test, and ANOVA, though PCC and T-test rankings remain consistent and similar to ANOVA.

		PCC	T-test	ANOVA	Chi-square	Gini index	MI
Ransomware	Top-1	CLIENT_TCP_FLAGS (f_9)			TCP_WIN_MAX_IN (f_{32})	IN_BYTES (f_3)	TCP_FLAGS (f_8)
	Top-5	f_9, f_{13}				f_3, f_{18}, f_{25}	
	Top-10	f_2, f_8, f_9, f_{10}, f_{13}, f_{14}, f_{17}, f_{32}				f_3, f_8, f_9, f_{15}, f_{18}, f_{25}, f_{33}	
Backdoor	Top-1	MIN_TTL (f_{13})				OUT_BYTES (f_4)	OUT_PKTS (f_6)
	Top-5	f_{13}, f_{14}				f_6	
	Top-10	f_2, f_8, f_{10}, f_{13}, f_{14}, f_{32}				f_3, f_4, f_5, f_6, f_{25}, f_{26}, f_{32}	



(c) Ransomware



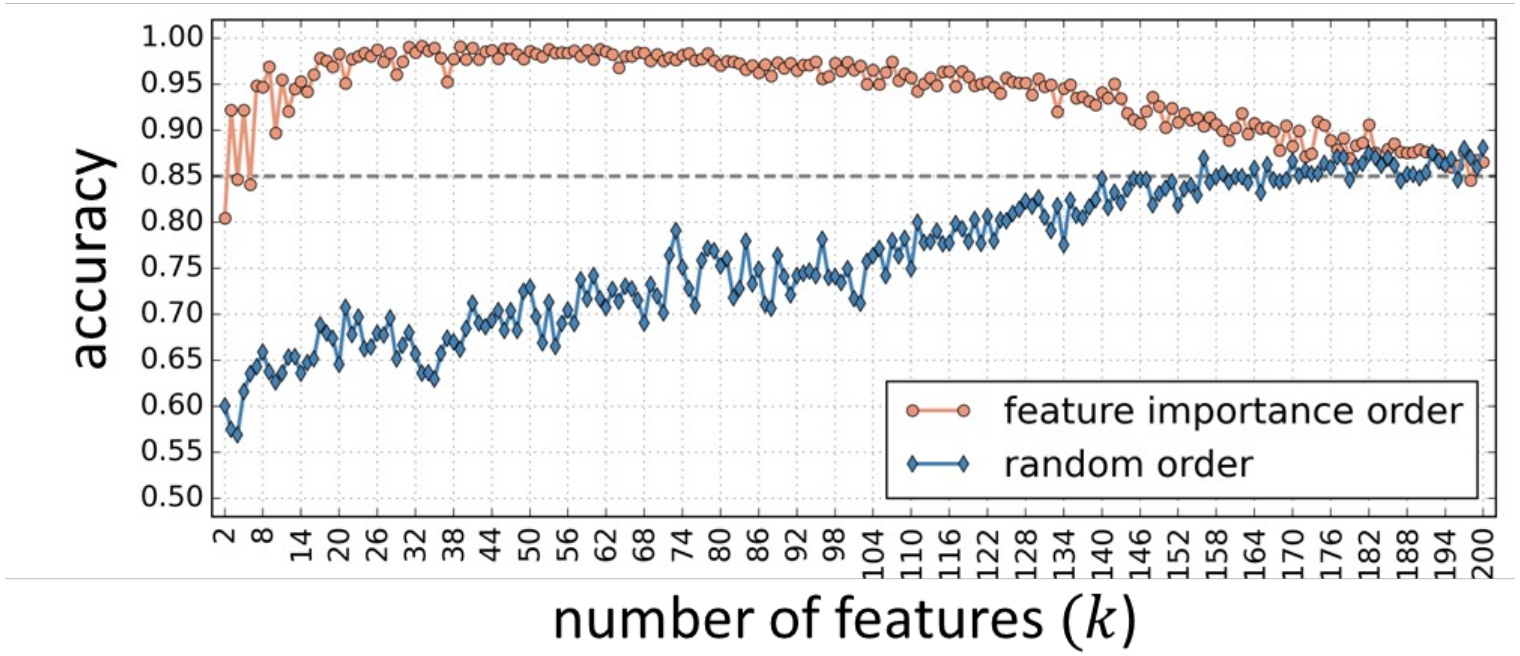
(d) Backdoor

Results and discussion – Feature analysis

- Attack-specific feature analysis
  - A discernible pattern in the feature selection algorithms across different types of attacks.
    - PCC, T-test, and ANOVA demonstrate similar patterns regardless of the type of attack.
    - MI and Gini index also exhibit a tendency to produce similar rankings.
    - In the case of Chi-square, some attacks have more similar rankings to PCC, T-test, and ANOVA than MI and Gini index.

Progressive Feature Addition

- Autoencoder-based anomaly detection using important features
- Results with synthetic dataset

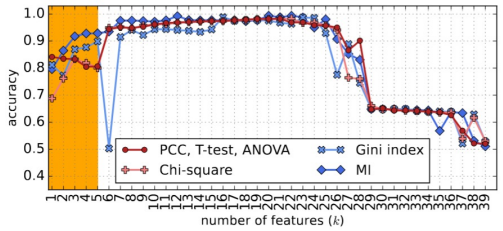


[D. Choi, J. Rheey and H. Park, "Autoencoder-based Anomaly Detection using Network Traffic Feature Grouping," *The 3rd Korea Artificial Intelligence Conference*, Sep. 2022.] (Best Paper Award)

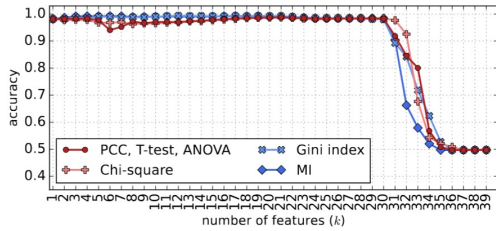
Progressive Feature Addition for IDS

- Attack-specific IDS based on Isolation forest and autoencoder
  - *NF-BoT-IoT-v2*

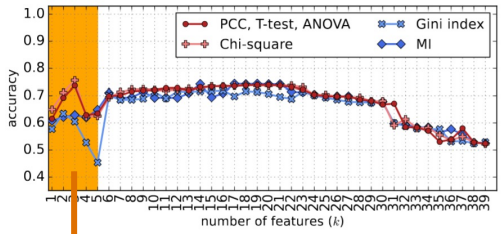
< Isolation Forest-based attack-specific IDS >



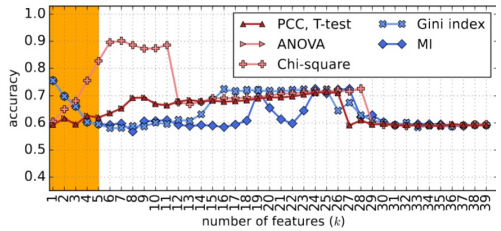
(a) DoS



(b) DDoS



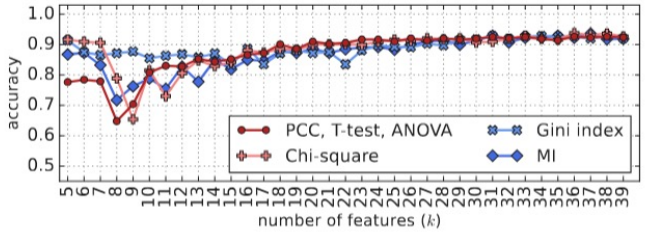
(c) Reconnaissance



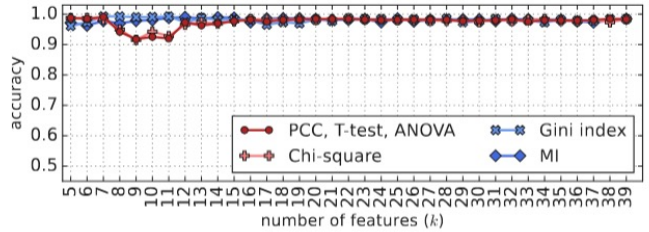
(d) Theft

average accuracy over permutations

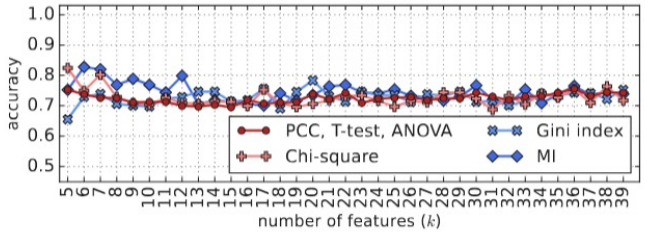
< Autoencoder-based attack-specific IDS >



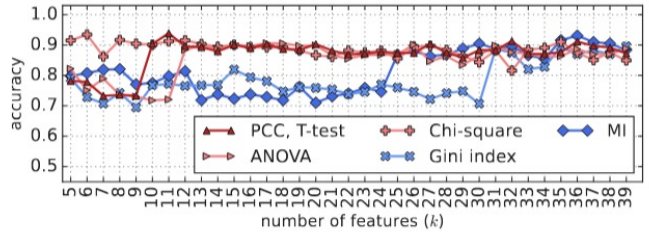
(a) DoS



(b) DDoS



(c) Reconnaissance

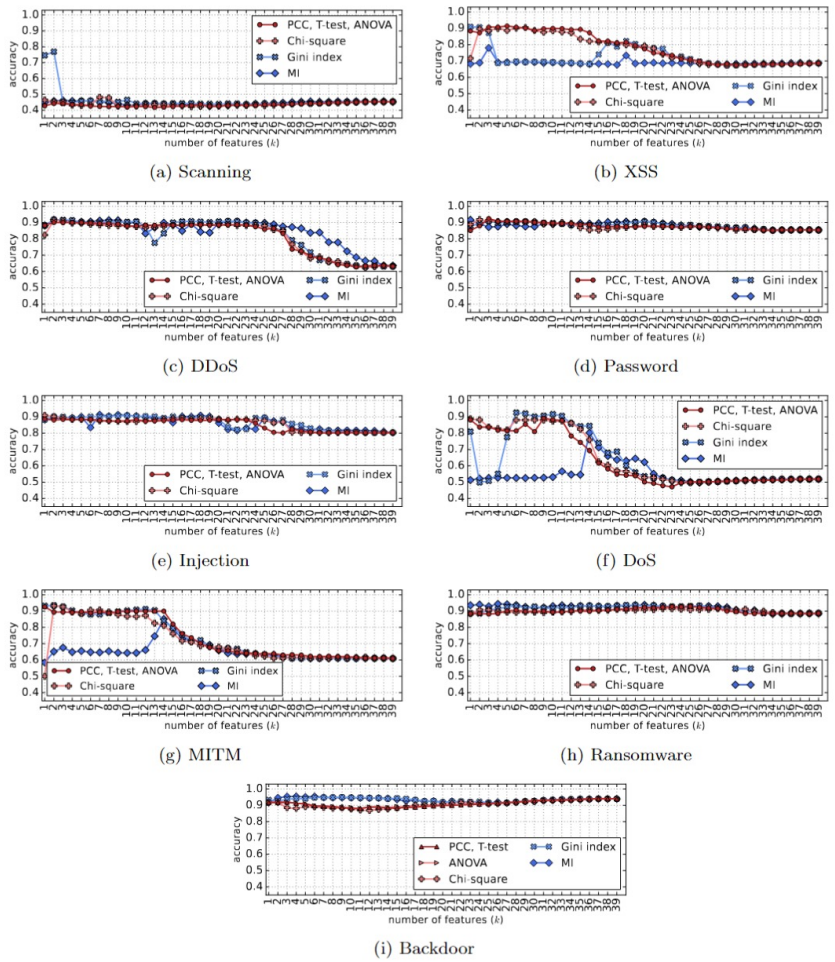


(d) Theft

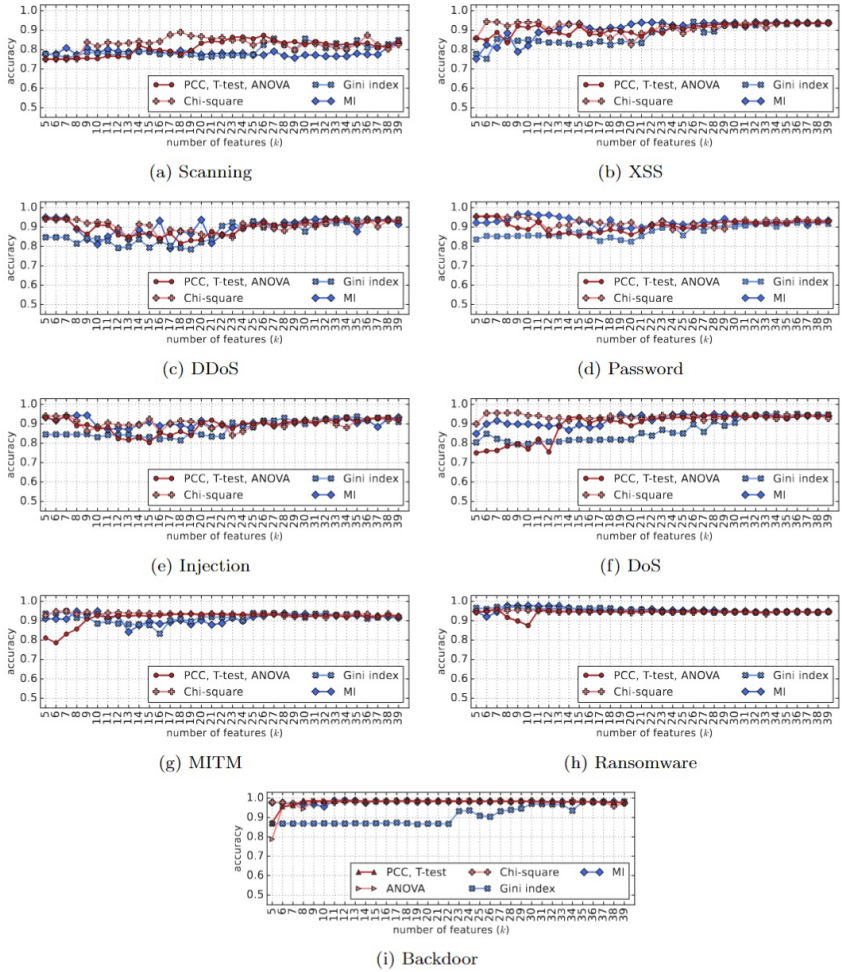
Progressive Feature Addition for IDS

- Attack-specific IDS based on Isolation forest and autoencoder
  - *NF-ToN-IoT-v2*

< Isolation Forest-based attack-specific IDS >



< Autoencoder-based attack-specific IDS >



Results and Discussion - IDS

- Attack-specific IDS based on Isolation Forest and autoencoder
  - Features are added incrementally in accordance with the common feature ranking in the case of feature selection algorithms with an identical ranking for 39 features.
    - Autoencoder-based attack-specific IDS
      - Pros: can achieve the highest performance with a few numbers of k
      - Cons: generally reach peak performance by using the full feature set
    - Isolation Forest-based attack-specific IDS
      - Pros: can detect attacks with high accuracy even with a small number of k for most attacks
      - Cons: may have capped performance
    - Important observation: a notable performance degradation when the size of k is increased excessively, regardless of the feature selection algorithm

Takeaway from use case 2

- To design ML/DL-based lightweight network traffic anomaly/intrusion detection system (IDS), we focus on efficient data manipulation approaches
  - Semantic feature grouping for network traffic data
  - Dimensionality reduction via network traffic data feature analysis
- Propose an attack-specific feature analysis framework with NetFlow IoT datasets, *NF-BoT-IoT-v2* and *NF-ToN-IoT-v2*
- We observe from extensive experiment results that
  - Some features are commonly important across different types of attacks/feature selection algorithms
  - For better improved and robust IDS, attack specific feature selection and design are still required

Conclusions

- For data driven AI-RAN,
  - Reference architectures for Open RAN and potential AI/ML deployment
  - Network data available for analysis
  - Communication and network constraints that need to be explicitly considered
  - Tradeoffs between AI/ML algorithms
  - Importance of feature extraction (feature reduction) for lightweight model design

Thank you

For more information, please visit our homepage at
http://mcnl.ewha.ac.kr