MACHINE LEARNING-ASSISTED PREDICTION IN PACKET-SWITCHED 5G XHAUL NETWORKS

Master Thesis Defense

Thesis supervisor: DAVIDE CAREGLIO (Department of Computer Architecture)

Thesis co-supervisor: JORDI PERELLÓ MUNTAN (Department of Computer Architecture)

Presented by Raul-Ioan Ferent

Universitat Politècnica de Catalunya — July 2, 2025

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

PRESENTATION ROADMAP

1. Background & Motivation
2. Dataset & Feature Engineering
3. First Results (Baseline Setup)
4. Other Tests (Stress + Generalization)
5. Conclusions

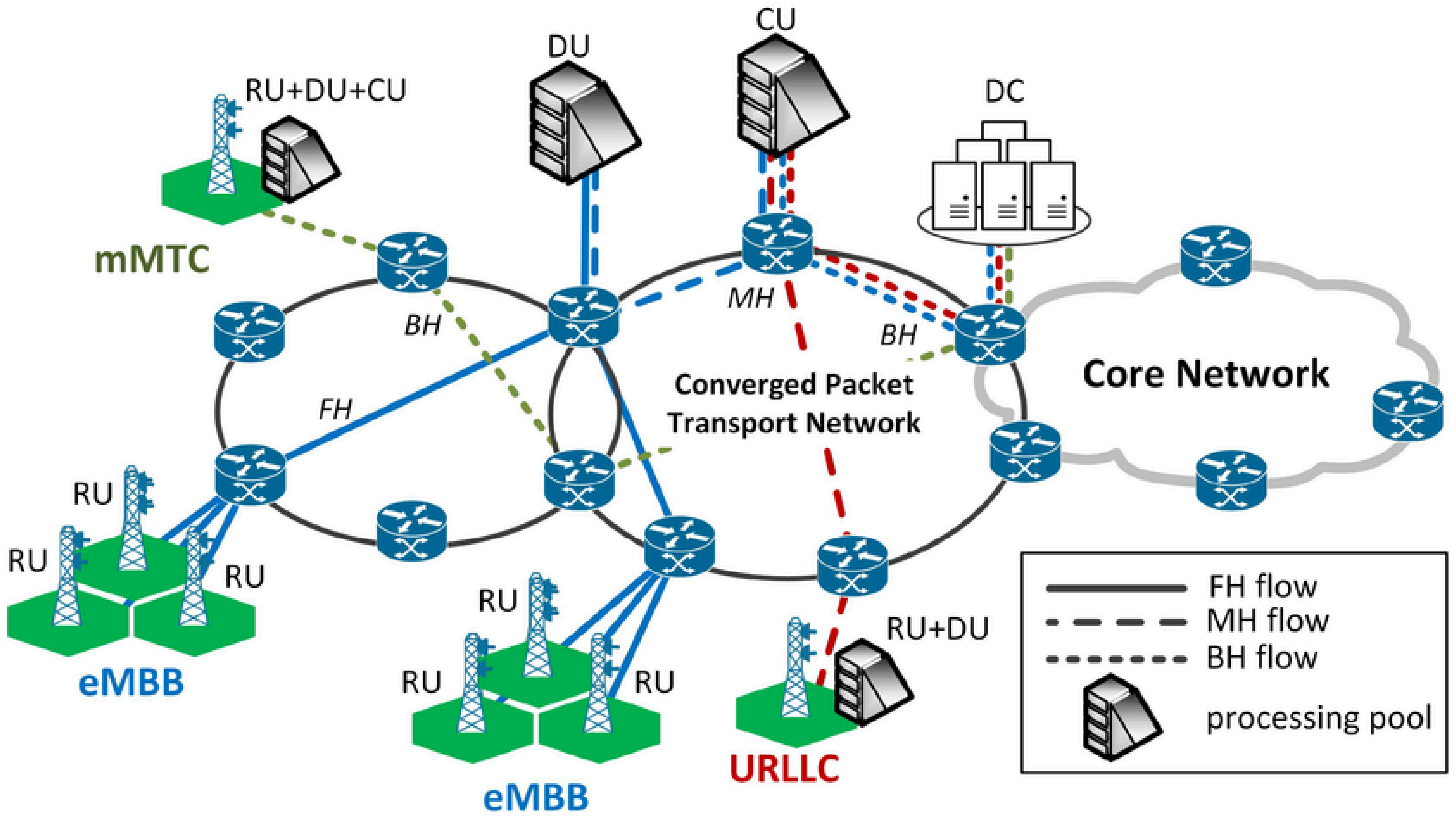
Why 5G Needs Smarter Decision-Making

- In 5G networks, ensuring low-latency communication is critical, especially for time-sensitive flows like URLLC
- Traditional solutions use worst-case estimators with rigid rules to ensure SLA compliance, but they are often too conservative, wasting resources and rejecting safe packets
- In this project, we explore how Machine Learning can provide smarter, adaptive mechanisms that make real-time, flow-aware decisions to improve efficiency and accuracy

BACKGROUND & MOTIVATION

What is 5G xHaul and Why It Matters

- 5G xHaul is the transport layer of the Radio Access Network (RAN), connecting Radio, Distributed, and Central Units
- It's split into three parts:
  - **Fronthaul (FH)**: RU → DU (very low-latency needed)
  - **Midhaul (MH)**: DU → CU
  - **Backhaul (BH)**: CU → Core Network
- Fronthaul is the most delay-sensitive part (critical for real-time flows)
- Ensuring SLA compliance across the full xHaul chain is essential for 5G reliability and performance



Inside the Radio Access Network (RAN)

- RAN connects user devices to the core network
- It’s split into 3 functional blocks:
  - **RU (Radio Unit)**: Handles wireless transmission, placed close to users
  - **DU (Distributed Unit)**: Executes real-time processing (PHY/MAC)
  - **CU (Central Unit)**: Handles control and non-real-time layers (PDCP, RRC)
- This split:
  - Reduces latency (DU closer to RU)
  - Enables centralization and cost savings
  - Supports scalable, flexible deployments

Understanding 5G Traffic & Network Flows

5G supports diverse use cases, each with different requirements in terms of bandwidth, latency, and reliability. These are typically grouped into three main traffic categories:

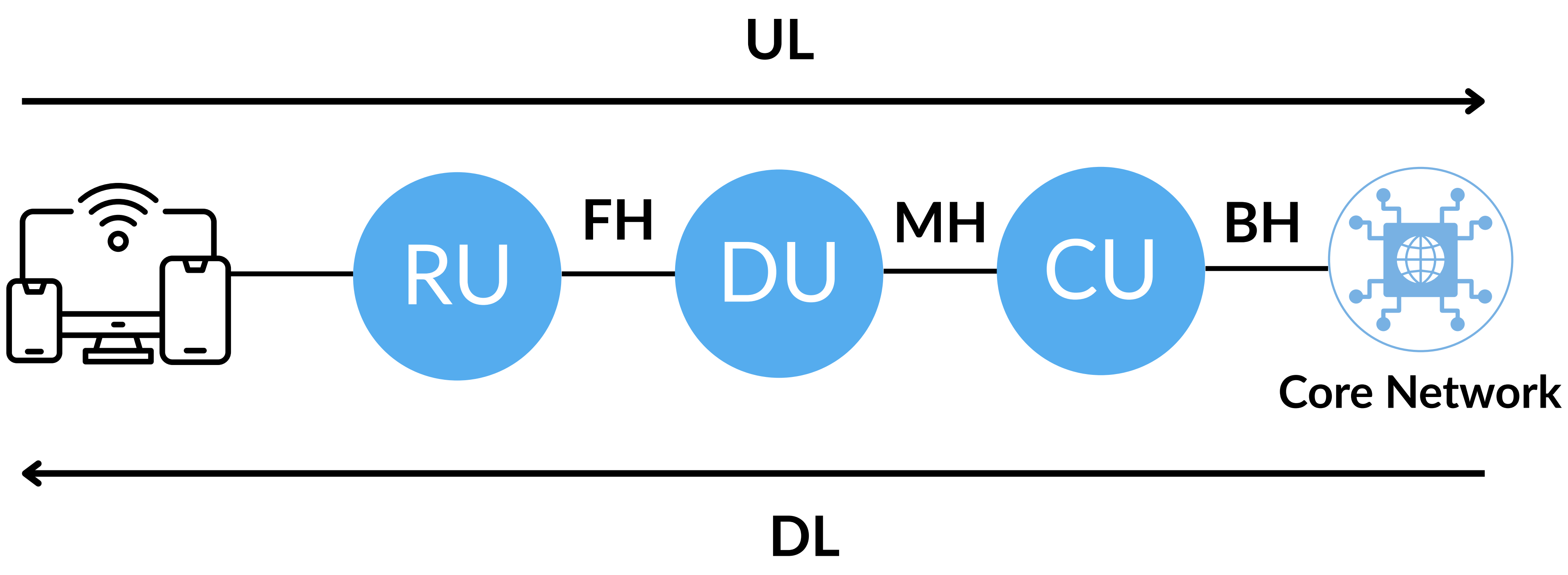
eMBB (Enhanced Mobile Broadband)	URLLC (Ultra-Reliable Low-Latency Communication)	mMTC (Massive Machine-Type Communications)
Needs high bandwidth, but tolerates some latency	Needs super-low latency and near-zero packet loss	Low bandwidth per device, but very high device density

UL (Uplink) – from user/device to the network

- Sensitive to congestion and queuing delays

DL (Downlink) – from network to user/device

- Typically better controlled, but still SLA-bound



Deterministic Worst-Case Estimator and its limits

Traditional xHaul admission control uses fixed, worst-case latency formulas

- **Safe**: Always respects SLA
- **Too conservative**: Rejects many flows that would be fine
- **Rigid**: Same logic for all traffic, no adaptation

$$L_{WC} = \sum_{i=1}^{n} \left(\frac{packet\,Size}{linkRate_i} + queueDelay_i \right)$$

Where:

- **packetSize** = size of the packet being transmitted (in bits)
- **linkRate$_i$** = speed of the network link at hop i (10 Gbps)
- **queueDelay$_i$** = estimated delay at hop i due to other queued traffic
- **i** = index for each network hop (1 to n)

This formula adds up the delay at each hop in the packet's path.
It combines:

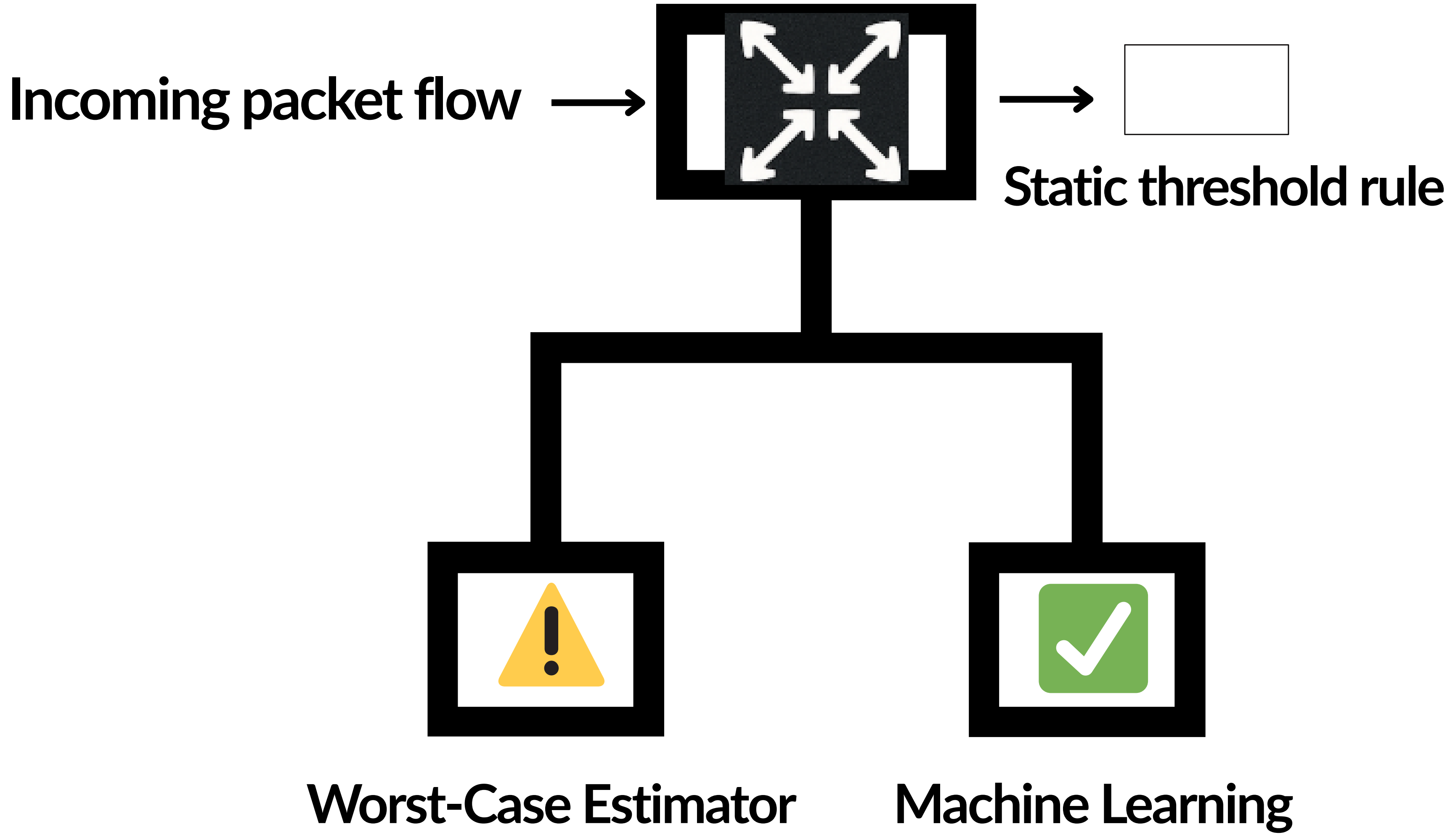
- transmission time (packetSize / linkRate$_i$)
- plus waiting time (queueDelay$_i$)

The result is the worst-case latency estimate for the full route.

Replacing Rigid Rules with Smart Predictions

Instead of always applying rigid worst-case rules and rejecting anything that looks risky, we use ML models to make smarter, flow-aware decisions.

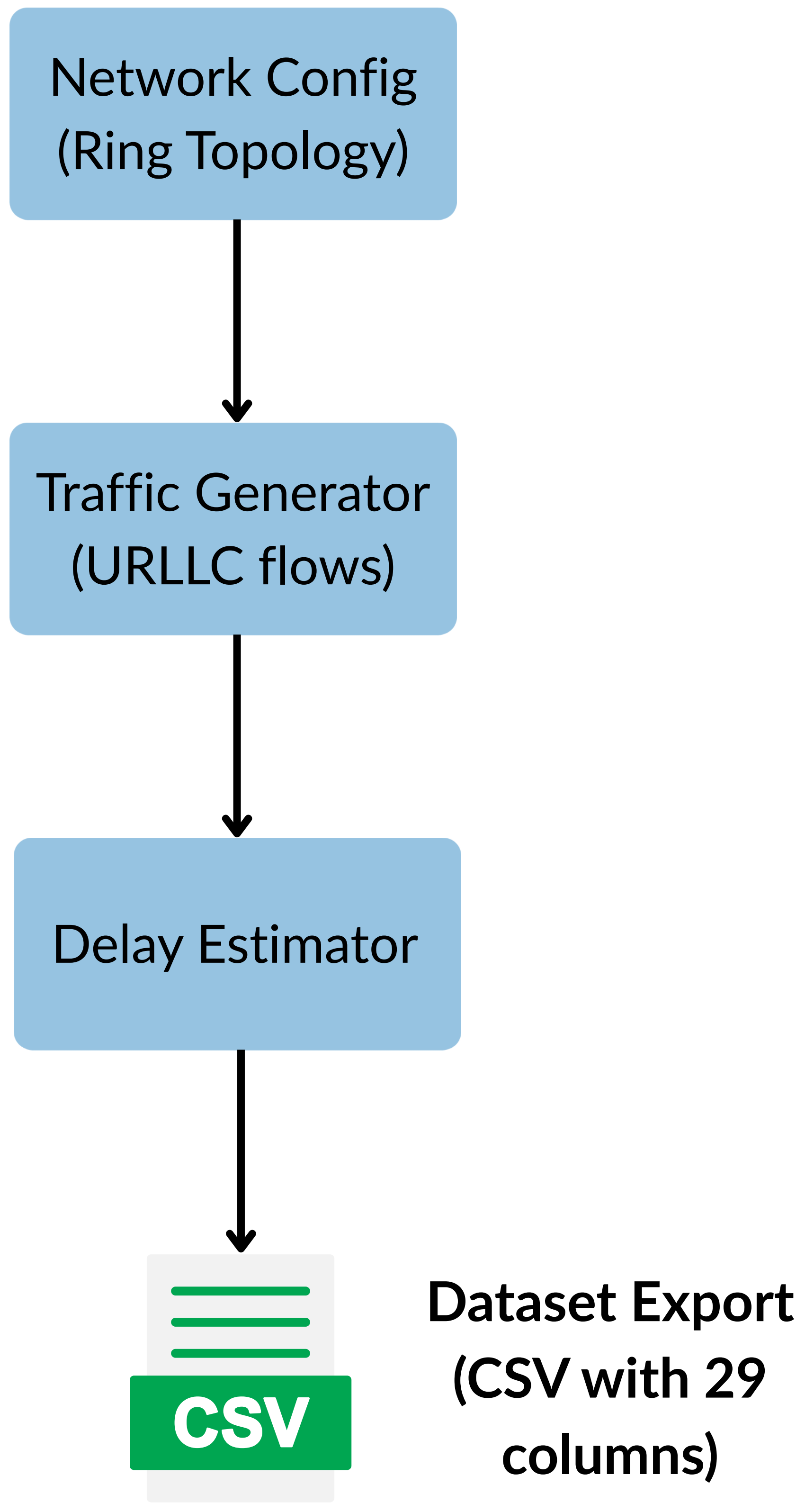
	Worst-Case Estimator	Machine Learning
Logic	Static rule (if WC > SLA → Reject)	Predict violation probability
Traffic Awareness	Same rule for all flows	Adapts to UL/DL, flow type, etc.
Precision	Over-conservative	Selective & accurate
Resource Usage	Over-provisioning, waste	Efficient, smarter allocations
SLA Adaptation	Cannot adapt to flow dynamics	Responds to real-time patterns



DATASET & FEATURE ENGINEERING

Simulation Dataset

- Provided by **Dr. Mirosław Klinkowski (5G xHaul simulator at NIT-Poland)**
- He shared them with us through the Spanish research project **TRAINER-A (PID2020-118011GB-C21)**, led by **UPC Computer Architecture Department**
- Generated using OMNeT++ simulation framework
- Simulates FH and MH flows; latency limits sampled between 100–250 µs (FH), 1 ms (MH)
- Our work focuses on 100 µs SLA, consistent with URLLC requirements



Dataset Variants and Scope of Study

- Each dataset represents a different network topology and includes detailed flow-level metrics and interference effects.
- The RING dataset is the primary focus of our experiments, while the MESH dataset supports generalization testing.
- Two datasets:
  - RING: 86,399 samples
  - MESH: 86,399 samples
- Each **contains 29 features**, including raw inputs, latency estimations, and interference metrics
- This thesis focuses on the RING dataset, particularly **FH_UL** and **FH_DL** flow types

Dataset	Topology	Flows	SLA Range	Used in Thesis
Ring	Ring	86,399	100–250 µs	Main Experiments
Mesh	Mesh	86,399	100–250 µs	Generalization

Label Definition: SLA Compliance Detection

Target variable:

- **latClass** is a binary label indicating whether a flow violates its latency SLA

Based on best latency metric:

- **latOverallSim** – includes full transmission and queuing delays

We define the SLA target:

- Acceptable one-way latency must be ≤ 100 µs (5G fronthaul standard)

Label construction rule:

$$latClass = \begin{cases} 1 & \textbf{if simulated latency (latOverallSim)>100 µs} \\ 0 & \textbf{otherwise} \end{cases}$$

Goal of ML model:

- Learn to predict **latClass** based on other known flow features.

Feature Engineering: Cleaning and Preparation Steps

Before model training, the dataset underwent a structured feature engineering process to ensure data quality and select informative variables.

- **Target Variable:** A binary class (latClass) was created based on the 100 µs SLA threshold
  - → latClass = (latOverallSim > 100) → {0, 1}
- **Data Integrity Check:** No missing values found
- **Flow Filtering:** Focused only on FH_UL and FH_DL flows using a mask
- **Initial Feature Candidates (11 columns):**

  ['flowTypeId', 'priority', 'trWindow', 'bitrate',

  'burstSize', 'latLimit', 'hops', 'buffers',

  'latStatic', 'latWCmodel', 'latEPsum']
- **Normalization - not applied:** Algorithms like Random Forest are not affected by feature scaling

Feature Selection: Correlation, VIF, and Recursive Elimination

To reduce multicollinearity and identify the most relevant features, several selection techniques were applied.

- **Correlation Heatmap**:
  - Visualized feature interdependence. Features with strong correlation were flagged for redundancy
- **VIF (Variance Inflation Factor)**:
  - Quantifies multicollinearity — VIF > 5 usually means high redundancy
- **RFE (Recursive Feature Elimination)**:
  - Stepwise method to eliminate the least important features
  - Applied with both **RandomForestRegressor** and **LogisticRegression** for robustness

More about the feature engineering can be found in section 3.3 of the report.

Final Features Used in Model Training

- The final selected features after all evaluation steps are:

["flowTypeId", "bitrate", "burstSize", "latLimit", "hops", "latWCmodel"]

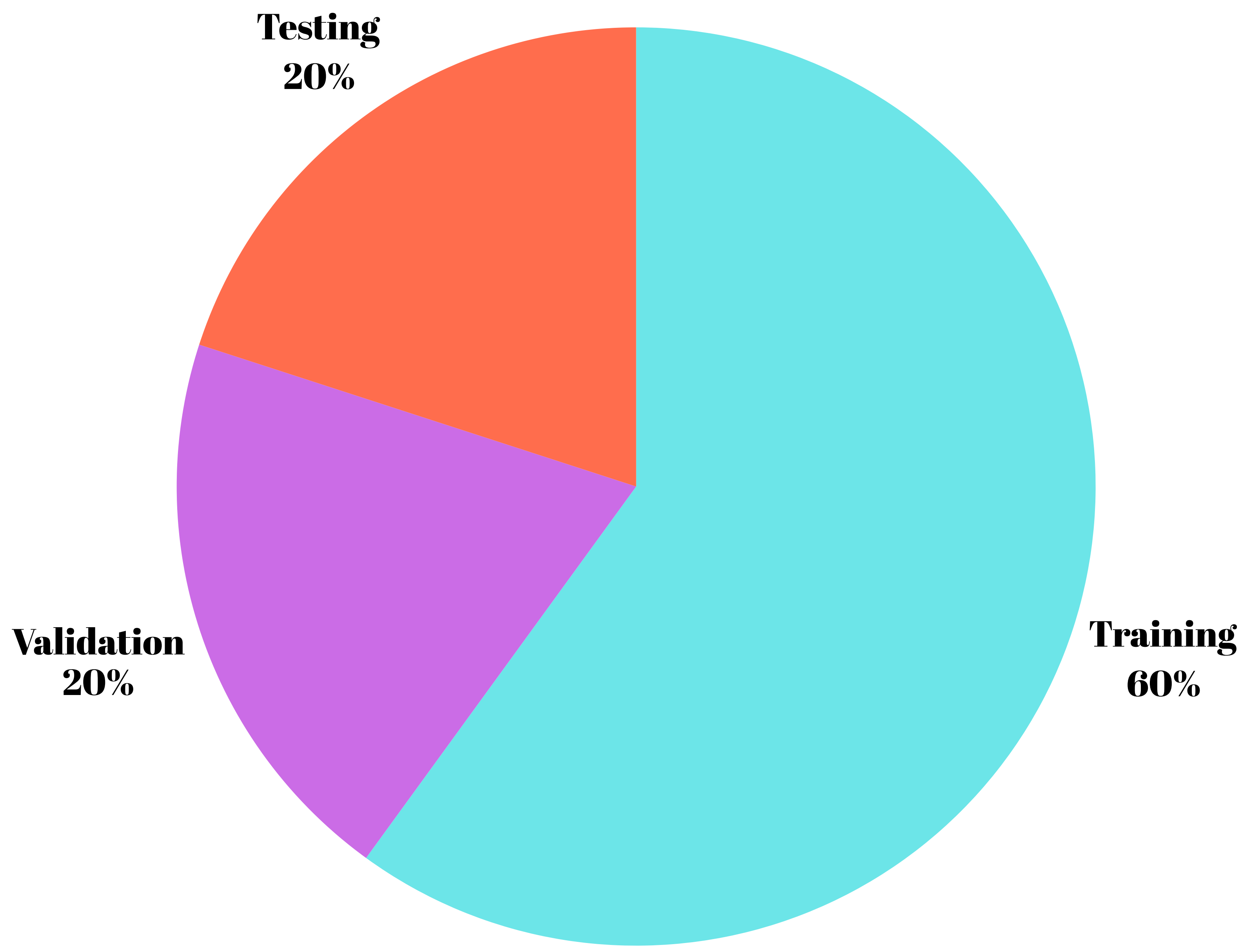
- Chosen for their:
  - Informative for latency
  - Low redundancy
  - Availability at decision time

Feature	What It Means (Simply)	Why It Matters
flowTypeId	Categorical ID for the flow type (FH_UL, FH_DL)	Some flows (uplink) are more sensitive to delay than others
bitrate	How much data is being transmitted per second	Higher bitrates can increase load and congestion risk
burstSize	How large the data bursts are	Bigger bursts = more pressure on buffers and queues
latLimit	SLA latency threshold for that flow	Helps target the classifier
hops	Number of intermediate links/nodes the packet must cross	More hops = more points of potential delay
latWCmodel	Traditional worst-case delay estimate	It captures a conservative baseline the model can learn to

FIRST RESULTS (BASELINE SETUP)

Dataset Split and Evaluation Strategy

- Dataset split:
  - 60% training
  - 20% validation
  - 20% testing
- Purpose of each split:
  - **Training set:** Fit model weights
  - **Validation set:** Tune threshold, evaluate cost
  - **Test set:** Final unbiased evaluation



Label Meaning and Confusion Matrix

$$\begin{vmatrix} \mathrm{TP} & \mathrm{FP} \\ \mathrm{FN} & \mathrm{TN} \end{vmatrix}$$

- TP (SLA violation correctly predicted)
- TN (SLA met correctly predicted)
- FP (SLA met predicted as violation → overly conservative)
- FN (SLA violation missed → risky)

Penalty-Based Evaluation (FP vs FN)

$$Cost = \alpha \cdot FP + \beta \cdot FN, \beta > \alpha$$

- Use validation set to:
  - Evaluate cost at threshold = 0.5
  - Try other thresholds (from 0.1 to 0.9)
  - Pick the one that minimizes Cost

Models used

- **Logistic Regression:** A simple linear classifier that estimates the probability of binary outcomes
- **Random Forest Classifier:** An ensemble of decision trees that improves prediction accuracy and robustness
- **Multi-Layer Perceptron (MLP):** A basic neural network with hidden layers that captures nonlinear patterns in the data

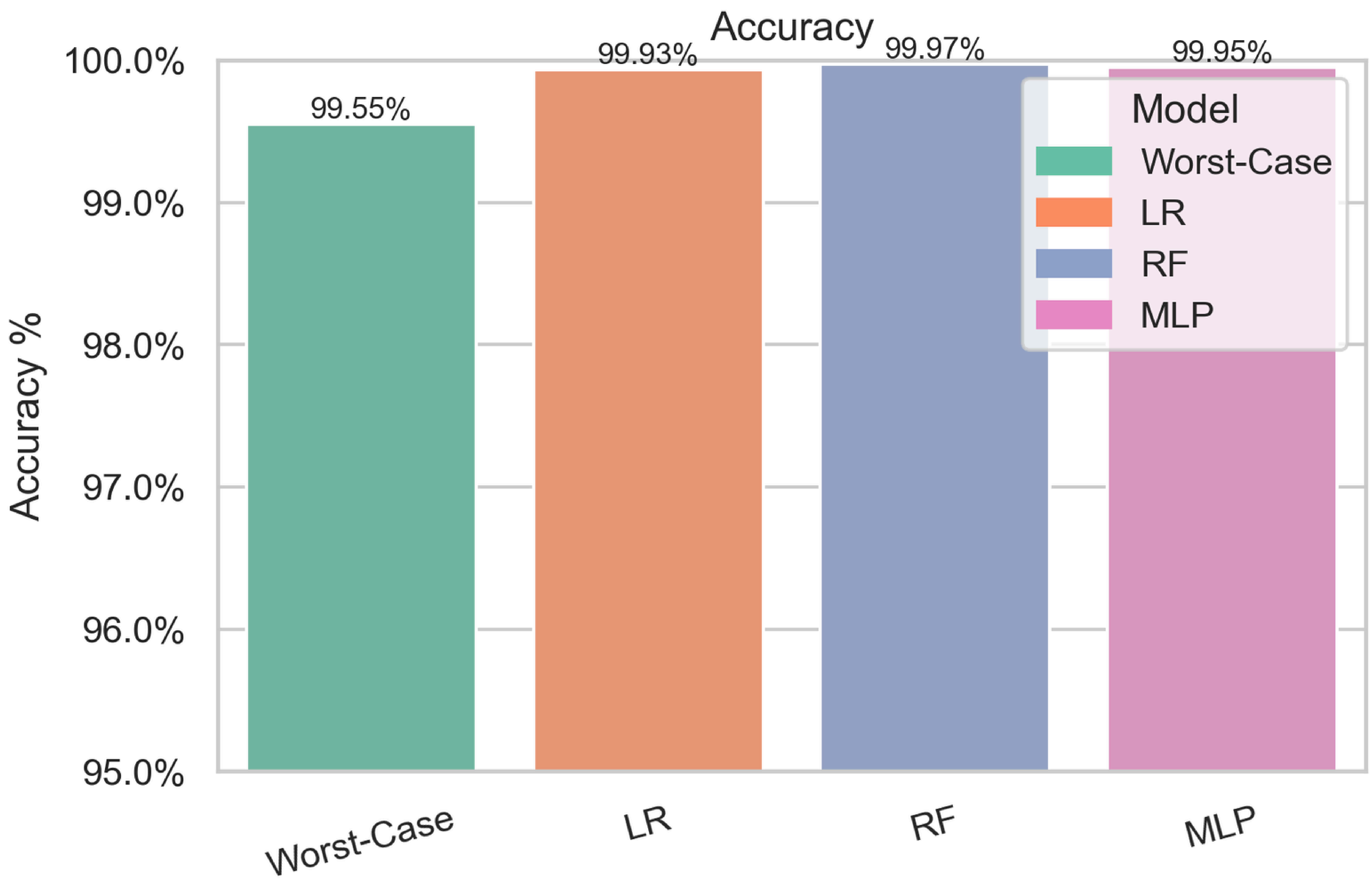
- **Libraries:**
    - Scikit-learn for LR, RF, and evaluation
    - Keras (TensorFlow backend) for MLP

Baseline Results and Cross-Validation

Model	Accuracy	Cost	FP	FN	TP	TN
Worst-Case Estimator	99.55%	39	39	0	18	8,583
Logistic Regression	99.93%	10	5	1	17	8,617
Random Forest	99.97%	3	3	0	18	8,619
Multi-Layer Perceptron	99.95%	4	4	0	18	8,618

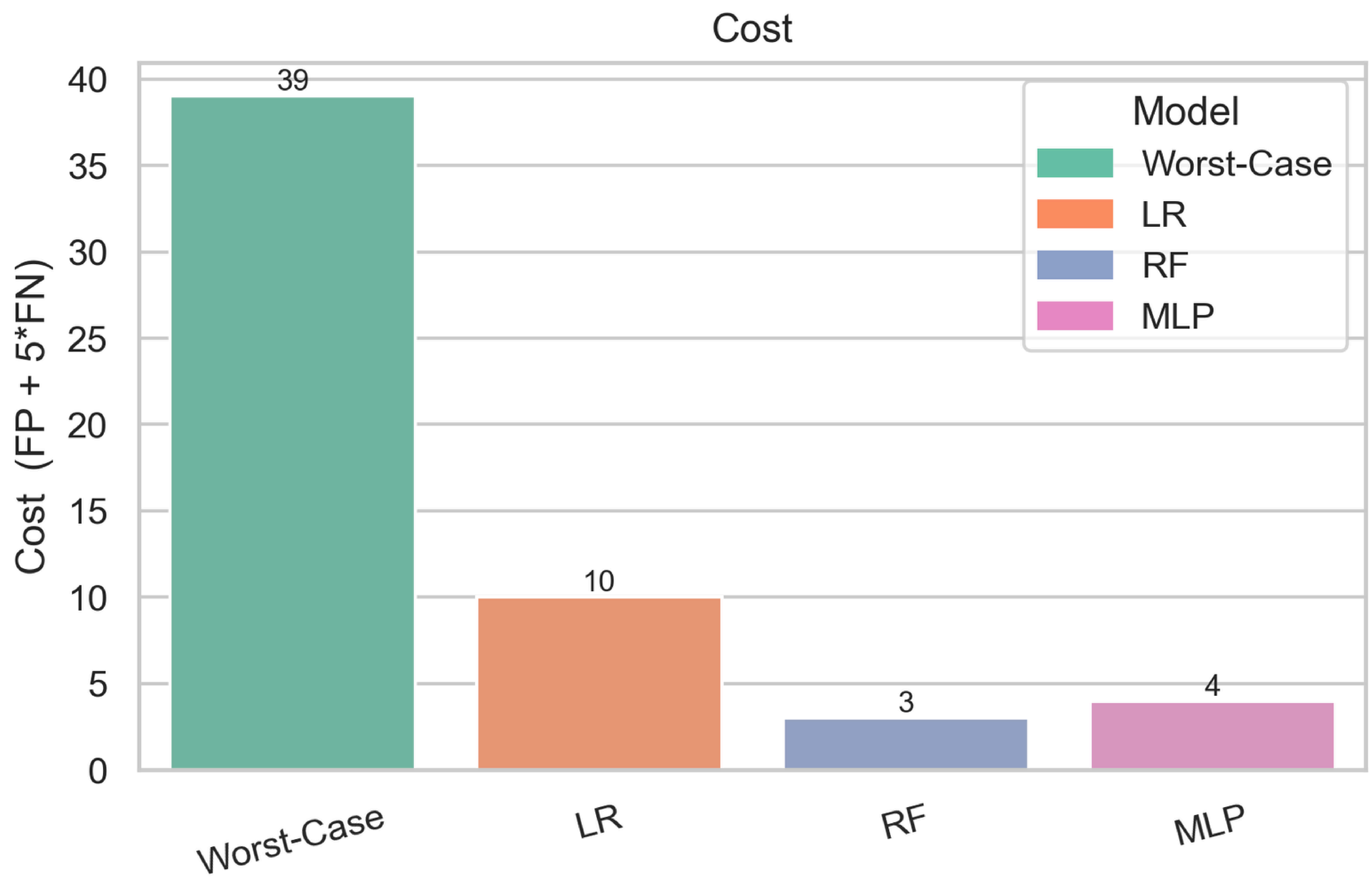
- Final evaluation done on test set using the best threshold per model.
- Results are averaged across 5-fold cross-validation to ensure stability.
- Gaussian noise added during robustness tests to evaluate model resilience.

Accuracy



Accuracy
100.0%
99.0%
98.0%
97.0%
96.0%
95.0%
Accuracy %
99.55%
99.93%
99.97%
99.95%
Model
Worst-Case
LR
RF
MLP
Worst-Case
LR
RF
MLP

Cost



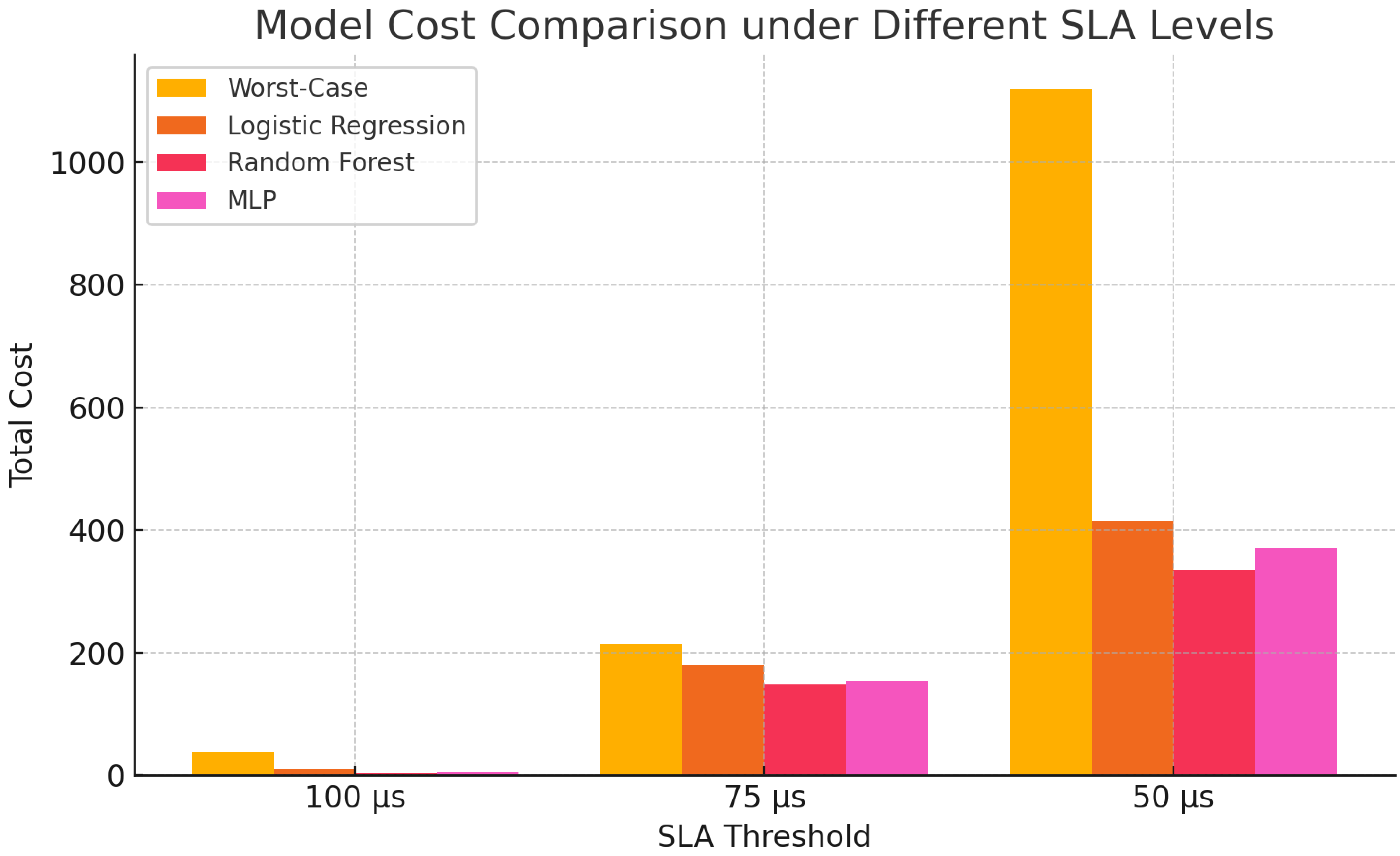
Cost
Cost (FP + 5*FN)
40
35
30
25
20
15
10
5
0
39
10
3
4
Model
Worst-Case
LR
RF
MLP
Worst-Case
LR
RF
MLP

OTHER TESTS (STRESS + GENERALIZATION)

Stricter SLA (75 µs & 50 µs)

- Tests model behavior under tighter delay constraints, simulating more demanding applications
- Evaluates if ML-based admission can still outperform WC when SLA margins are reduced
- Helps determine the robustness of each model when the problem becomes harder
- Bellow is presented the total cost, where FN is penalized with 5 and FP with 1

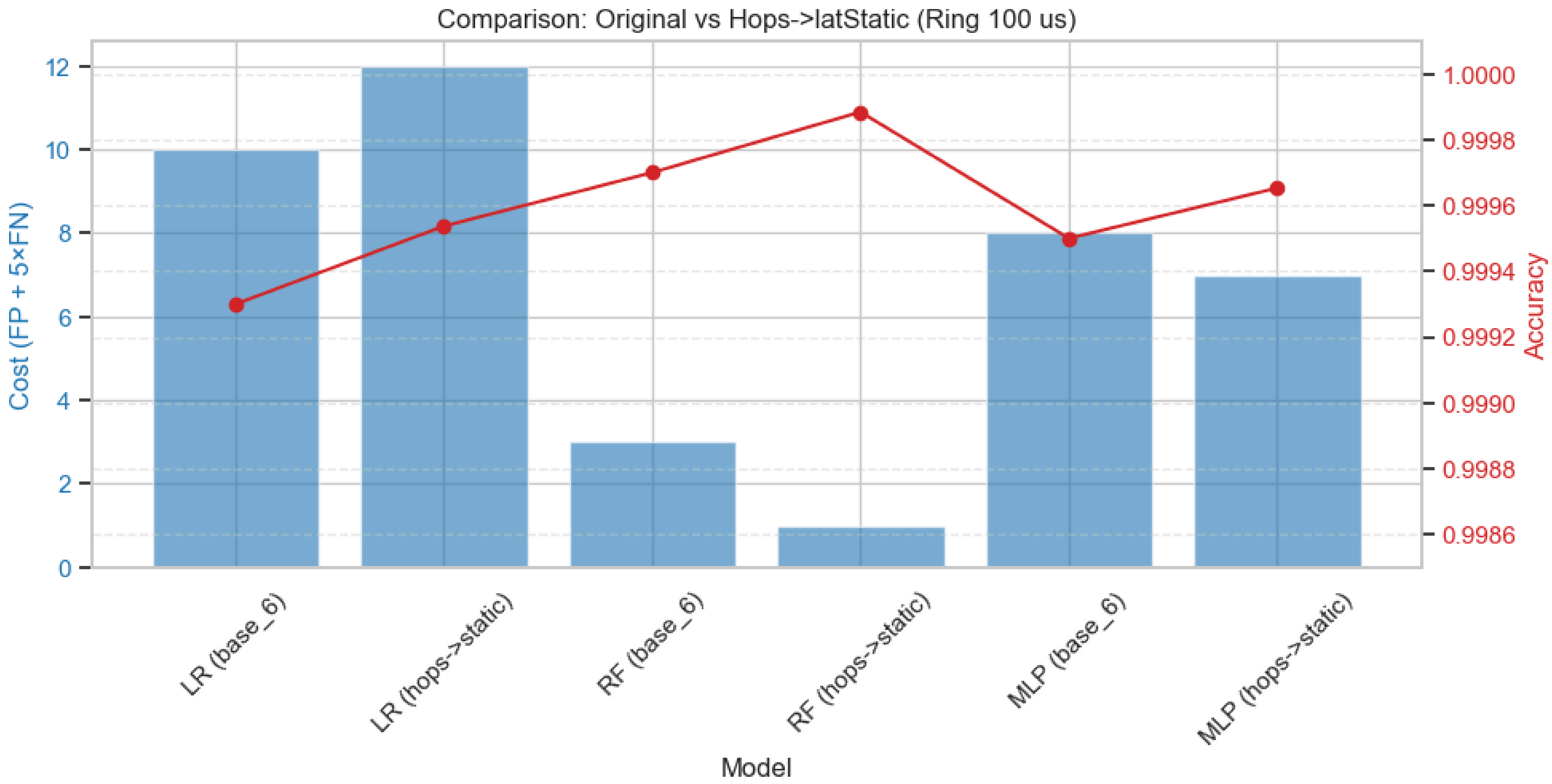
Model	100 µs SLA	75 µs SLA	50 µs SLA
Worst-Case Estimator	39	214	1,120
Logistic Regression	10	181	415
Random Forest	3	148	335
Multi-Layer Perceptron	4	154	371


Model Cost Comparison under Different SLA Levels
Worst-Case
Logistic Regression
Random Forest
MLP
Total Cost
0
200
400
600
800
1000
100 µs
75 µs
50 µs
SLA Threshold

Hops → latStatic Swap

- Swaps the feature "**hops**" with "**latStatic**", which estimates propagation delay
- Compares topological (hops) vs physical-delay-based (latStatic) predictors
- Checks if ML captures network structure or link quality better

Model	**Accuracy**	**Cost (FP + 5·FN)**	**FP**	**FN**	**TP**	**TN**
Logistic Regression	99.95%	12	2	2	16	8,620
Random Forest	99.99%	1	1	0	18	8,621
MLP	99.97%	7	2	1	17	8,620


Comparison: Original vs Hops->latStatic (Ring 100 us)
Cost (FP + 5×FN)
Accuracy
0
2
4
6
8
10
12
0.9986
0.9988
0.9990
0.9992
0.9994
0.9996
0.9998
1.0000
LR (base_6)
LR (hops->static)
RF (base_6)
RF (hops->static)
MLP (base_6)
MLP (hops->static)
Model

Cost-weight Sensitivity - Comparison

- Logistic Regression

FN Cost	Threshold	FP	FN	Total Cost
3	0.306	5	1	8
5	0.306	5	1	10
7	0.122	16	0	16
10	0.122	16	0	16

- Random Forest

FN Cost	Threshold	FP	FN	Total Cost
3	0.143	3	1	6
5	0.02	3	0	3
7	0.102	8	1	15
10	0.102	8	1	18

- MLP

FN Cost	Threshold	FP	FN	Total Cost
3	0.265	2	1	5
5	0.265	2	1	7
7	0.265	2	1	9
10	0.041	18	0	18

Cross-Topology Generalization (Ring → Mesh)

- Tests if models trained on Ring data generalize to the Mesh topology
- Frozen Models: direct test with no retraining – checks overfitting
- Retraining: re-learn weights using Mesh data – checks adaptability

Frozen Models

Model	Accuracy	FP	FN	Cost
Worst-Case Estimator	97.53%	213	0	213
Logistic Regression	99.06%	350	57	635
Random Forest	97.54%	737	324	2,357
MLP	99.00%	292	139	987

Retrained

Model	Accuracy	FP	FN	Cost
Worst-Case Estimator	97.53%	213	0	213
Logistic Regression	99.21%	57	11	112
Random Forest	99.39%	43	10	93
MLP	99.19%	62	8	102

CONCLUSION & FUTURE RESEARCH

Key Takeaways & Lessons Learned

- ML classifiers can outperform deterministic WC estimators while preserving SLA compliance
- Random Forest achieved the best trade-off: high accuracy, zero FN, and lowest cost
- Feature selection was critical — a compact set of 6–7 inputs was enough
- Threshold tuning allowed flexible optimization between FP and FN risks
- ML maintained robustness under stricter SLAs and noisy conditions

Future Work

- **Online deployment:** Extend the models to real-time admission control environments
- **Scalability:** Test on larger topologies and dynamic traffic traces
- **Explainability:** Add interpretable models (SHAP) for operator trust
- **Multi-objective admission:** While our current focus has been on latency, real world networks often require balancing multipleobjectives, such as jitter and energy consumption

THANK YOU FOR ATTENTION!