



Scripting Traffic, Aligning Power: A 5G Testbed Integrating gNB Activity and Commercial UE Power

- **Adel Heidari**, University of California, San Diego

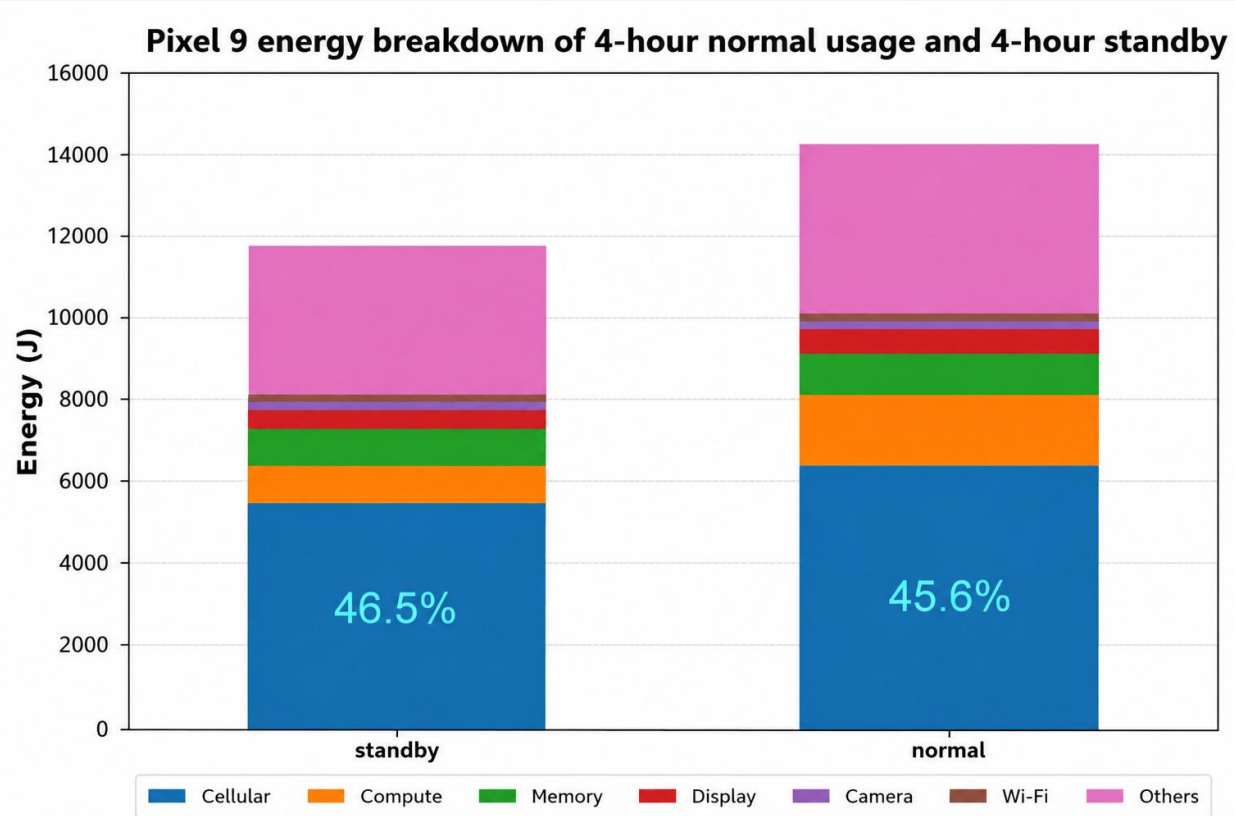
WiNTECH '26 • Austin, TX

Haochen Wang, Aaron Tartz, Adel Heidari, Kotturi Sai Prasanth, Agrim Gupta, Radha Krishna Ganti, Dinesh Bharadia

IIT Madras | UC San Diego

The cellular modem is a top energy consumer

Google Pixel 9 · 4 h normal use vs. 4 h standby



~46%

of total device energy goes to cellular, in **both** standby and normal use

Why it matters

- More than compute, memory or display

- Wi-Fi off: standby = screen off, background apps on

Saving UE power starts with knowing what drives the modem's power

Follow the traffic: from app to UE power

APPLICATION TRAFFIC

- Burst shape and timing
- Downlink vs. uplink mix
- Data volume and rate

What to send?

gNB decisions

- PRBs, symbols, MCS, TBS
- HARQ, power control
- DRX, BWP, SRS / CSI-RS

When and How?

Modem

does the per-slot work

- PDCCH monitoring
- PDSCH reception and decoding
- PUCCH / PUSCH transmission, SRS

Execution?

Power



app



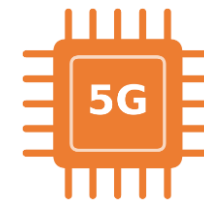
packets, in bursts



gNB



per-slot grants



modem

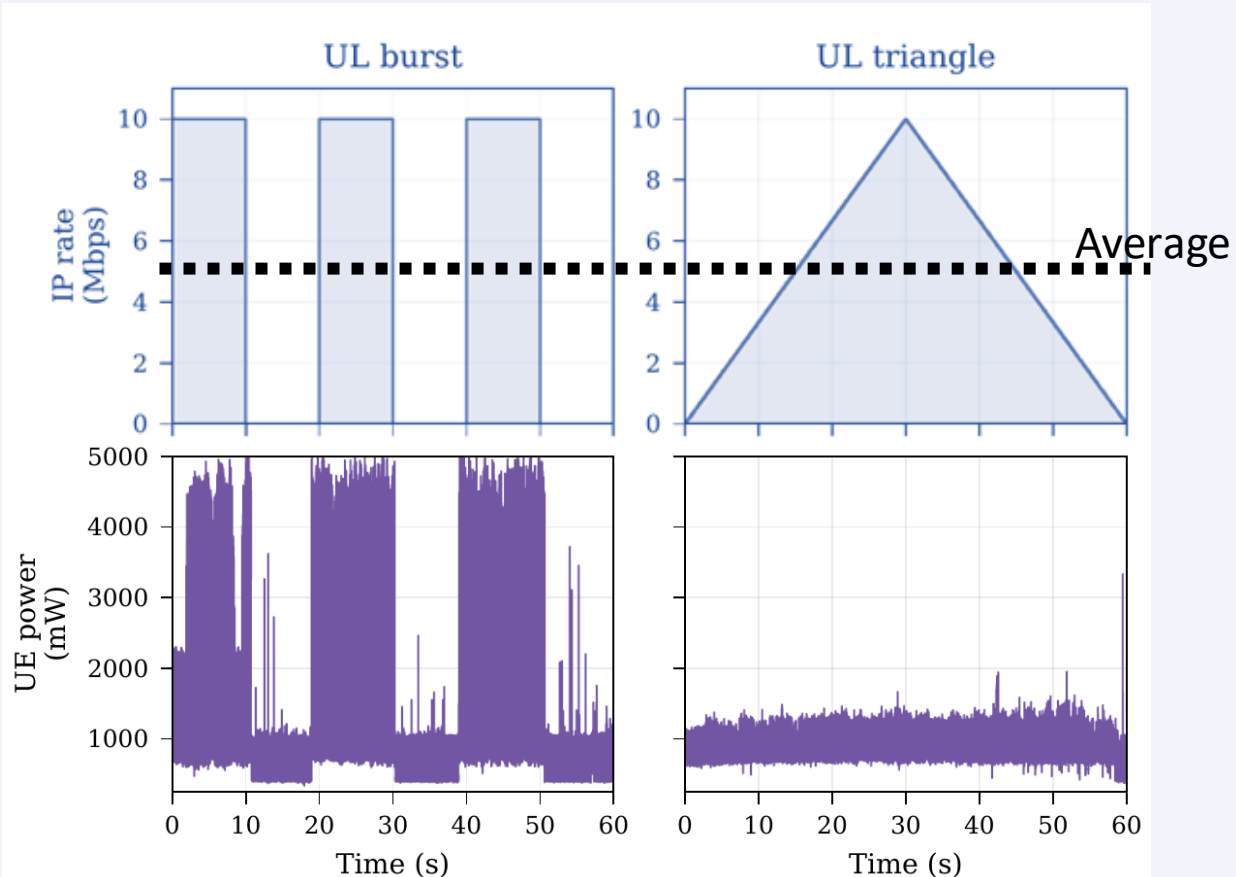


UE power (measured)

UE power is set jointly by application traffic and gNB decisions

Same throughput, different shape → different energy

UL burst vs. UL triangle: same total data over 60 s



Same aggregate UL throughput over 60s

Different UE energy: bursts draw far more power

Average traffic volume does not predict UE energy

Why is this hard to study?

1

Isolated systems

Application traffic, gNB and UE run on separate systems, so they are hard to instrument and control together.

Needs one testbed that sees all three

2

Fine-grained measurements

0.5msec slot-level power measurements

Needs one testbed that is as granular as cellular scheduling

3

Unrepeatable traffic patterns

UE power depends on how traffic evolves over time: bursts, ramps, DL/UL mixes. Average throughput hides this, so real-app traces alone are not enough.

Needs arbitrary, scripted traffic

Requirement

- Integrated measurements of app, gNB and modem
- Fine-grained measurements (slot-level)
- Repeatable traffic patterns

Platform overview

Integrated 5G hardware platform

srsRAN gNB + commercial modem, over the air

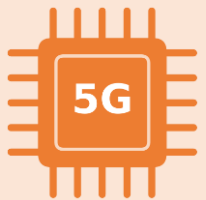


srsRAN gNB logs

per-slot PHY, scheduler



real or scripted traffic (DL + UL)



Qualcomm X65 modem

measured at the 0.5ms granularity

Slot alignment

puts both streams on one 0.5ms slot grid

Slot-aligned UE power + gNB logs

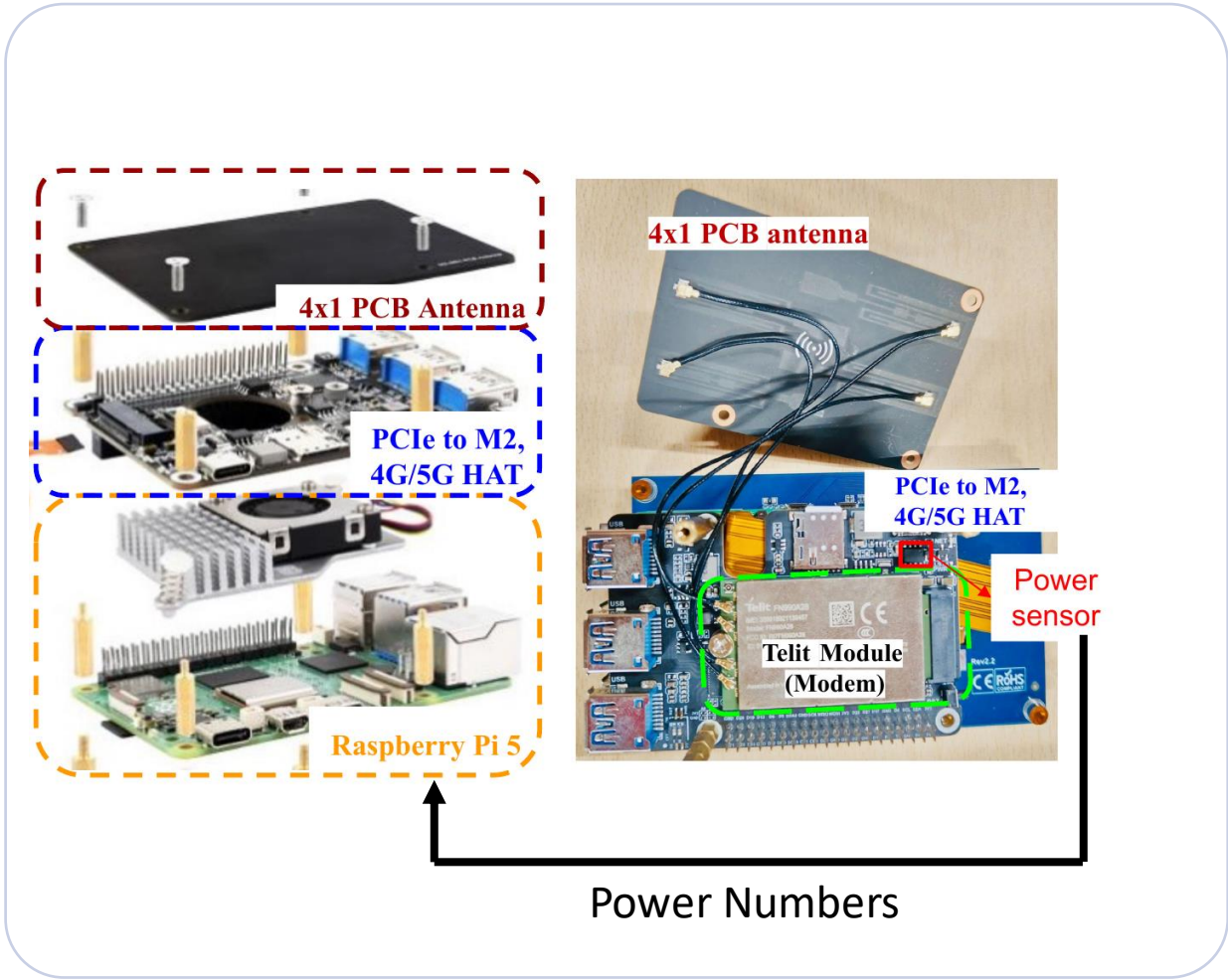
slot	gNB logs	UE power
t	■ ■ ■	■
t + 1	■ ■ ■	■
t + 2	■ ■ ■	■

...

one row per UE per slot

Meets all three: integrated · slot-level (0.5 ms) · repeatable traffic

Testbed: commercial modem, open RAN, over the air



Component	Configuration
RAN	srsRAN 5G standalone gNB
Core	Open5GS 5GC
Radio	USRP B210, OTA link
Carrier	Band n41, 20 MHz, 30 kHz SCS
UE	Telit FN990A28, Snapdragon X65
Measured	Modem-supply power via INA219
gNB grain	One row per UE per 0.5 ms slot
Workload	5 app traces; scripted DL/UL UDP

Commercial Qualcomm X65 modem · open-source RAN

Slot-level alignment

Slot alignment

Enumerate integer slot shifts s ;
closed-form least squares gives the
affine transform (scale a , offset b);
keep lowest error

1-D search over s

$$(s^*, a^*, b^*) = \arg \min_{s \in \mathbb{Z}, a, b \in \mathbb{R}} \sum_t \left(P_t - \left(a \hat{P}_{t+s}^{\text{3GPP}} + b \right) \right)^2$$

P_t measured UE power on the slot grid · p_t^{3GPP} 3GPP power model per-slot activity value from the gNB log · s slot shift
 a, b affine transform (scale, offset): turns the model's relative units into mW to compare with measured power

Alignment decides when a power sample belongs to a slot, not how to model it

Estimates track measured power



1 • YouTube, ~1.5 s
measured $\approx 1100 \text{ mW}$,
model $\approx 800 \text{ mW}$

2 • Instagram
model never drops to
the idle floor

3s windows show the slot-alignment follow the mean power

Testbed application: Improving UE power from gNB logs

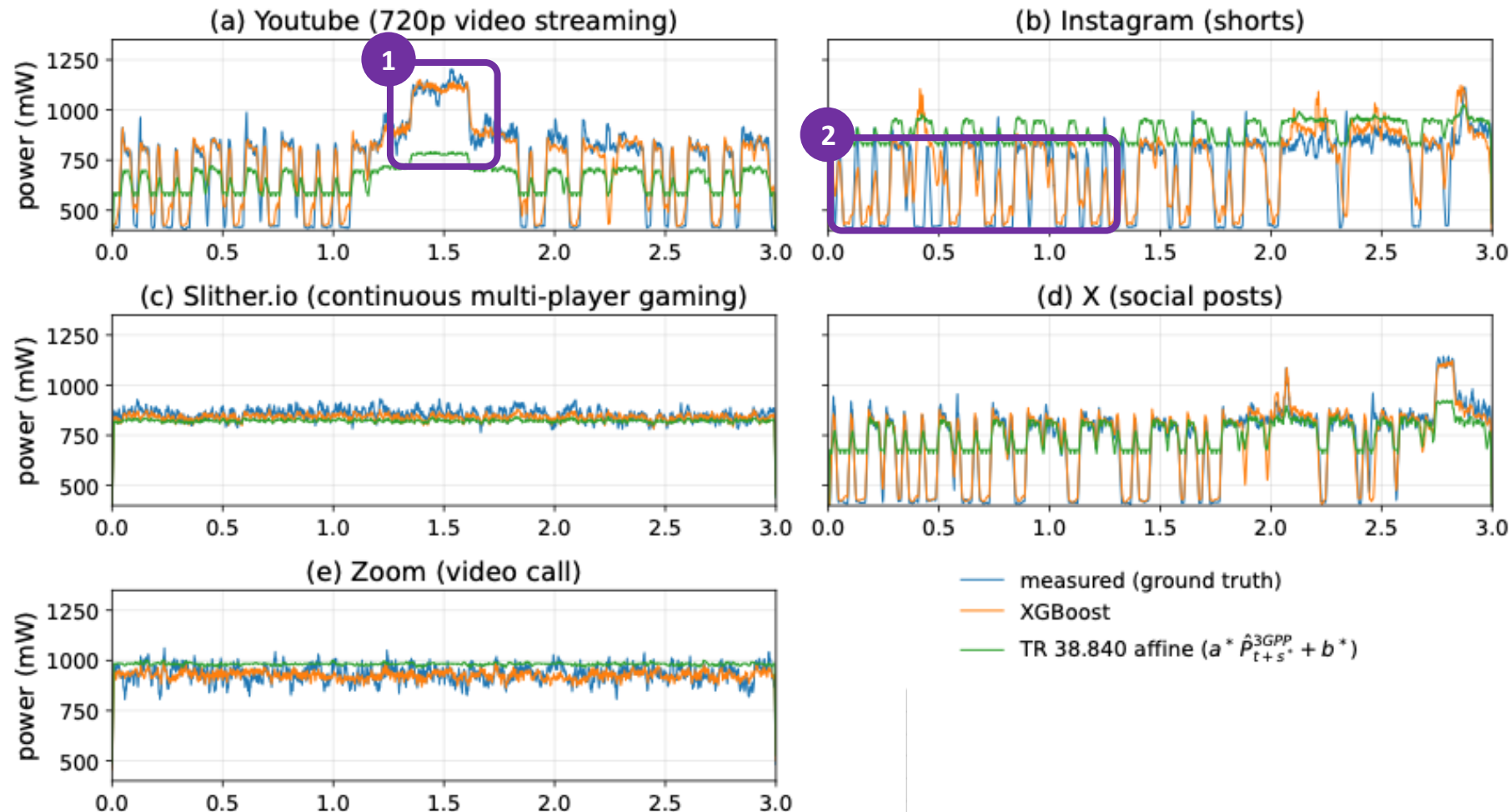
$$\hat{P}_t = f_{\theta}(\mathbf{x}_t - H, \dots, \mathbf{x}_t)$$

causal history H · no future slots · XGBoost

Family	Representative features
Timing / context	slot, SCS, TDD direction, active BWP, DRX
DL control / data	PDCCH count & aggregation; PDSCH PRBs, symbols, MCS, TBS, layers, HARQ
UL control / data	PUCCH format & bits; PUSCH PRBs, symbols, MCS, TBS, SINR, HARQ
Reference signals	SRS / CSI-RS occurrence and bandwidth
Scheduler state	buffer occupancy, PHR, retransmissions
Temporal context	lags {1...320} slots; rolling mean/max {4...160}

Only causal, gNB-observable features; no UE-side instrumentation at inference

Estimates track measured power



1 • YouTube, ~1.5 s
XGBoost follows $\approx$ 1100 mW; 3GPP stays $\approx$ 800

2 • Instagram
XGBoost reaches the idle floor; 3GPP never does

Held-out 3 s windows: the estimate follows slot-level bursts, not just the mean

In-capture accuracy

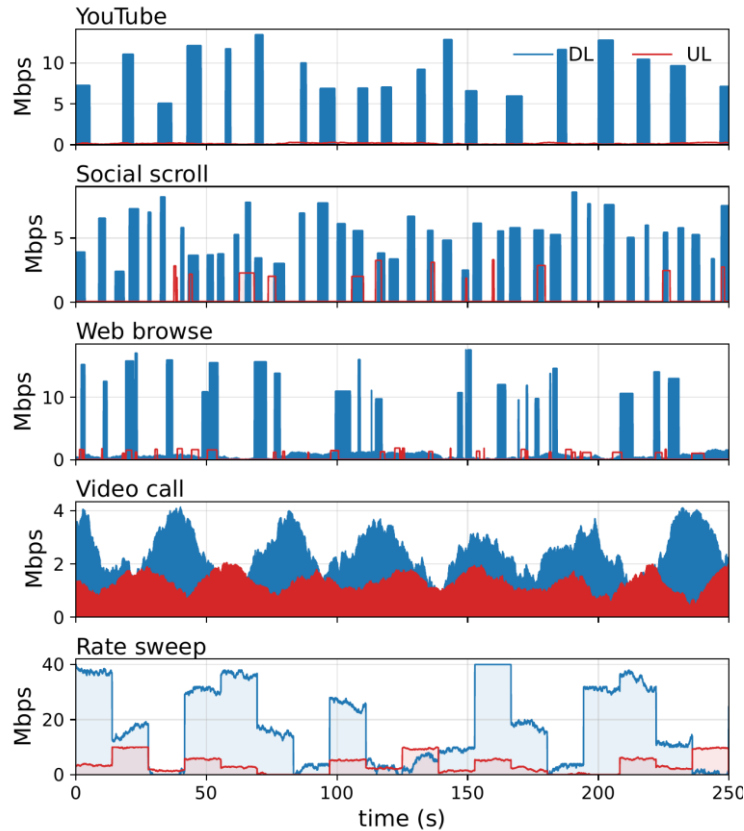
60 s capture per app: first 50 s align + train, final 10 s held out. Each cell: MAPE (%), lower is better

Application	3GPP power model (raw)	3GPP power model (scaled)
YouTube	27.4%	27.5%
Instagram	27.8%	49.6%
slither.io	30.9%	12.2%
X	29.5%	24.3%
Zoom	31.5%	19.6%
Macro	29.4%	26.6%

Our estimator halves the error of the 3GPP power model: 13% vs. 27% MAPE

Train on synthetic traffic, test on real apps

Synthetic traffic (training only)



Real app traffic

traffic types and
rate ranges

AI agents

script synthetic
DL/UL profiles · 18
captures

Train XGBoost

on synthetic data
only

Predict UE power

on unseen real
apps

MAPE on unseen real apps (lower is better)

Ours 21.5% · raw 3GPP 27.3% · aligned 3GPP 26.6%

Trained only on scripted traffic: 21.5% MAPE on unseen real apps

Takeaways

Testbed

Real or scripted DL/UL traffic, gNB logs and commercial modem power in one setup

Integrated · slot-level · repeatable

Slot-aligned dataset

Per-slot gNB activity and measured modem power on one time axis

Aligned on a 0.5 ms slot grid

UE power estimator

UE power from gNB-side features alone, even when trained on scripted traffic

13% vs. 27% MAPE (aligned 3GPP)

Future work

- **App activity → UE energy.** Record, edit and replay IP traffic (tcpdump / tcpreplay). E.g., push notifications that keep the modem active long after the data (C-DRX inactivity timer).
- **gNB decisions → UE energy.** Sweep DRX, MCS and grants with traffic fixed.
- **Beyond one testbed.** More modems and bands; whole-device energy.

Thank you!

Questions?

Adel Heidari · UC San Diego · adheidari@ucsd.edu

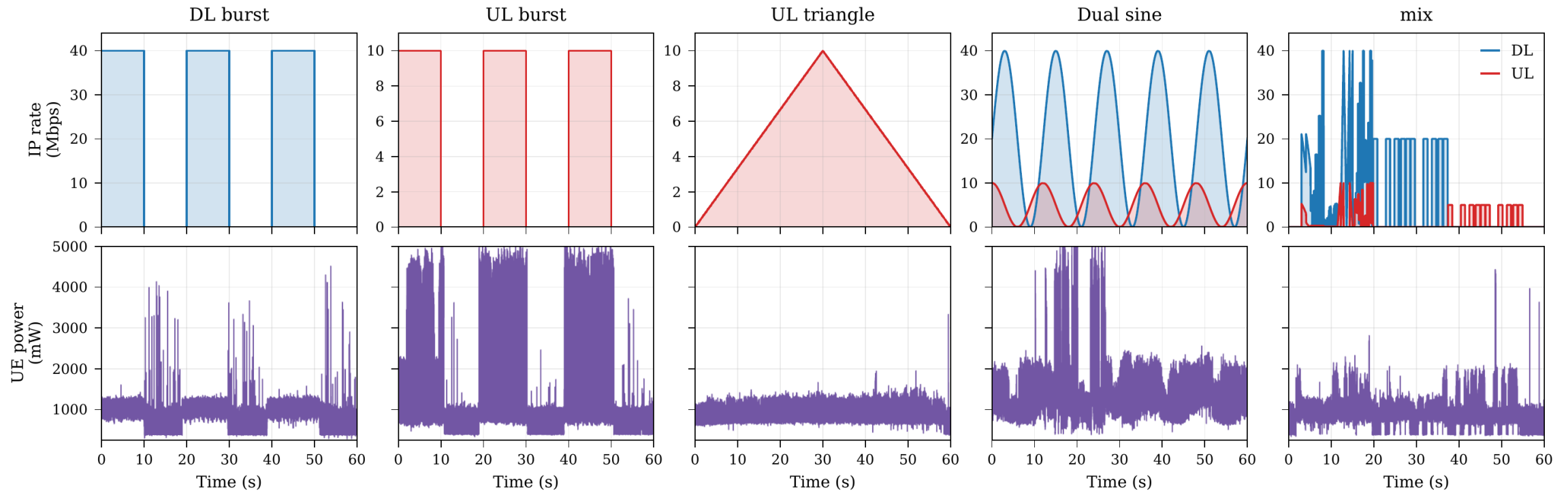


wcsng.ucsd.edu/scripting-power



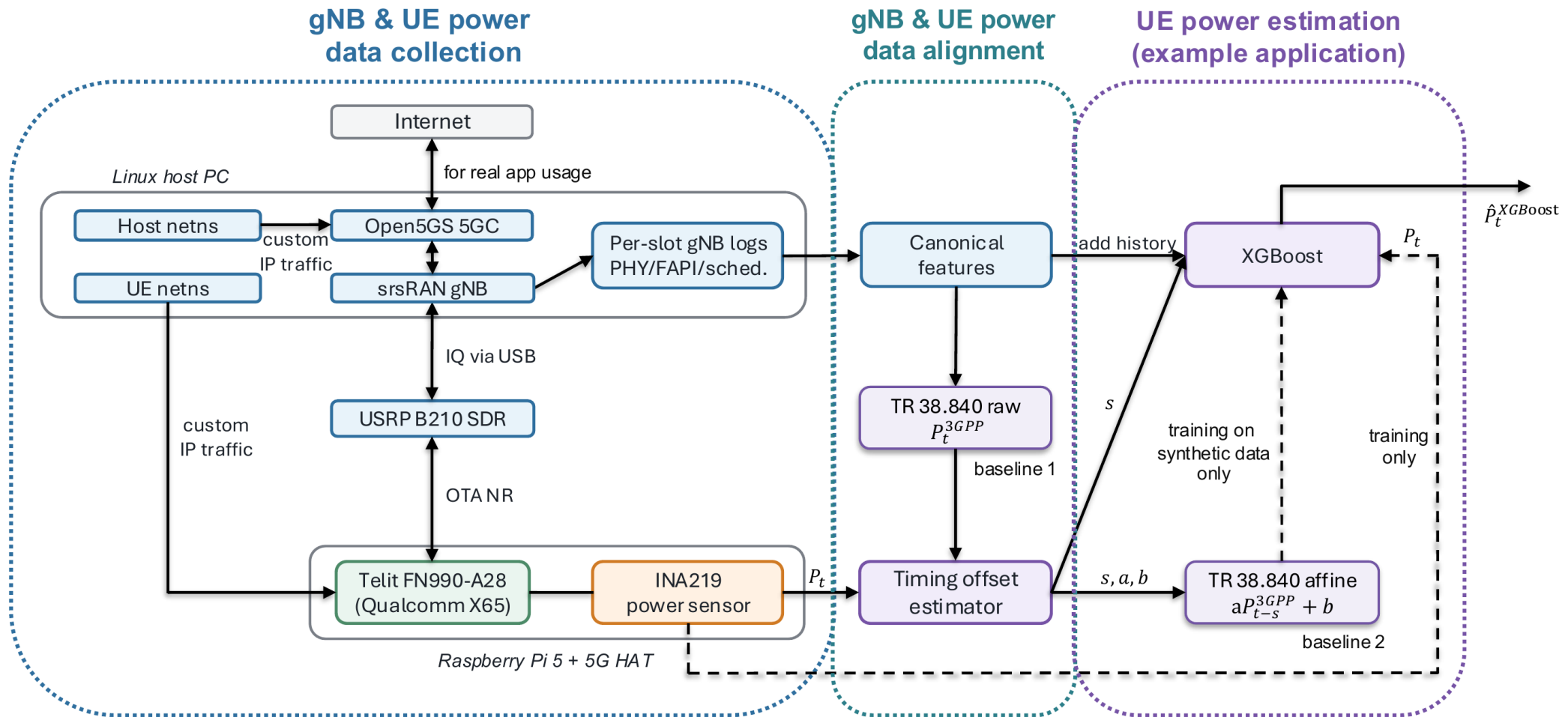
adheidari.github.io

Backup: programmed traffic probes



Top: offered DL/UL IP rate. Bottom: measured UE power

Backup: full platform diagram



Backup: synthetic-to-real generalization

MAE (mW) / MAPE (%) / interval energy error (%). Target-fit 3GPP power model: aligned, with a, b fitted on held-out target power.

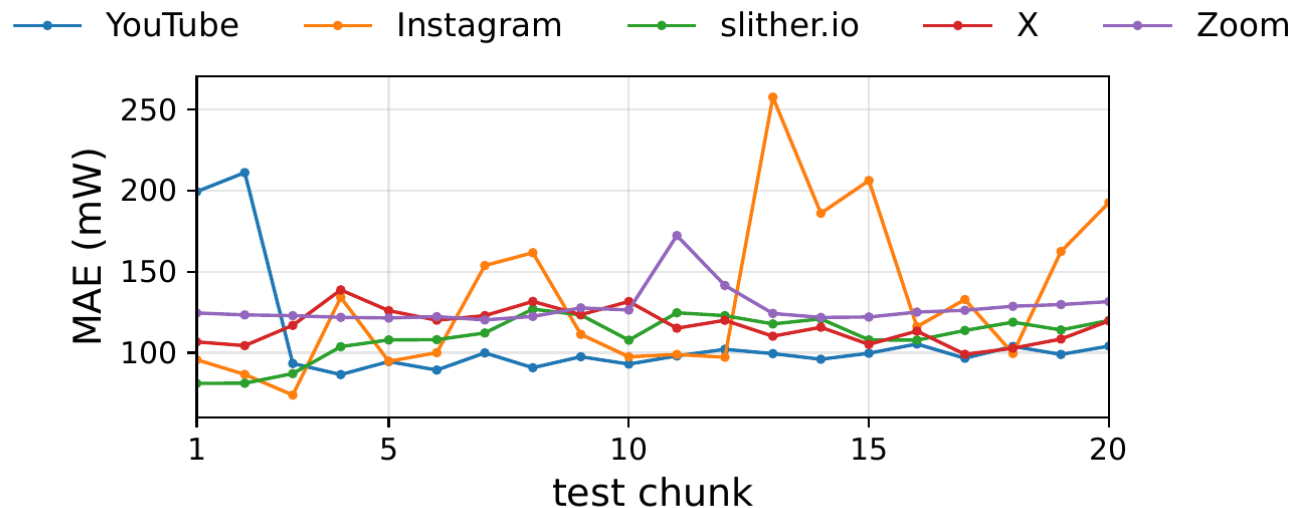
App	In-capture (ref.)	TR synth-fit	TR target-raw	TR target-fit	Ours (synth-trained)
YouTube	93 / 13.5 / 1.5	203 / 35.4 / 16.8	192 / 26.2 / 5.6	182 / 27.5 / 6.8	148 / 24.0 / 3.6
Instagram	105 / 15.8 / 0.9	201 / 35.7 / 17.1	188 / 26.4 / 4.7	251 / 49.6 / 29.9	164 / 28.5 / 8.3
slither.io	86 / 10.0 / 1.6	165 / 20.1 / 12.4	229 / 27.2 / 19.8	119 / 14.5 / 4.3	107 / 12.2 / 3.8
X	86 / 10.3 / 0.9	194 / 26.9 / 8.1	245 / 27.7 / 10.1	176 / 24.3 / 4.5	152 / 21.6 / 4.0
Zoom	133 / 15.4 / 0.7	195 / 22.0 / 4.1	265 / 29.1 / 12.0	162 / 19.6 / 5.2	167 / 18.8 / 1.2
Macro	101 / 13.0 / 1.1	192 / 28.0 / 11.7	224 / 27.3 / 10.4	176 / 26.6 / 10.1	150 / 21.5 / 4.3

Synthetic-trained estimator: 150 mW macro MAE vs 176 mW for target-fit 3GPP power model

Stable over time, cheap to run

MAE on consecutive 10 s chunks, frozen model

Train on first 50 s, test on the next 200 s



No runaway drift

Accuracy holds across 20 chunks, **no retraining**. Instagram (bursty) varies most.

$\approx 5 \mu\text{s}$

per slot, batched, 1 core

$\sim 90\times$

below 500 μs slot

Single-slot path $\approx 350 \mu\text{s}$, still under budget.

Model evaluation only; end-to-end online deployment is future work.

Runs forward without retraining, at a fraction of the slot budget

One radio path, two workload modes

Real applications

Browser in a UE network namespace $\leftrightarrow$ public Internet, through Open5GS $\rightarrow$ srsRAN $\rightarrow$ OTA

YouTube · Instagram · slither.io · X · Zoom

Scripted traffic

Programmable UDP generator in the host namespace, injected into the **same** Open5GS $\rightarrow$ gNB $\rightarrow$ UE path

Same radio stack; only the traffic source changes

Programmable bidirectional generator

A schedule is a timeline of short intervals, each with a target DL and UL rate. Zero rate sends no packets, so idle gaps stay idle.

10 ms

rate-target updates

1 ms

pacing slices

1200 B

UDP payloads

DL + UL

synchronized clocks

Current approaches: model the power instead

1 Throughput $\rightarrow$ power

$$P \approx f(\text{average throughput})$$

- ✓ App traffic
- ✗ gNB decisions
- ✗ real modem power

2 Analytical: 3GPP power model

$$P_{\text{rel}}(t) \text{ from a per-slot state table}$$

- ✗ App traffic
- ✓ gNB decisions
- ✗ real modem power

3 Calibrated: affine 3GPP power model

$$a \cdot P_{\text{rel}}(t) + b \rightarrow \text{mW}$$

- ✗ App traffic
- ✓ gNB decisions
- ✓ real modem power

Throughput and additive state tables cannot capture how a real modem draws power