FFT-Only vs Learned Spectral Proxies for Rapid RF Triage

Benjamin J. Gilbert
Spectrcyde RF Quantum SCYTHE
bgilbert2@com.edu

***Abstract*—We compare a normalized 1024-pt FFT magnitude with light post-filters against a tiny 1-D CNN for rapid digital-vs-analog RF triage. Over a calibrated SNR sweep (-10 to $+20$ dB), FFT+filters *outperforms* the CNN proxy (peak AUROC 0.754 vs 0.671) while using $\sim 48\times$ fewer FLOPs and markedly lower tail latency (p99 0.0 ms vs 1.6 ms at 0 dB). A hybrid gate that escalates only ambiguous cases to CNN achieves a $4.8\times$ compute reduction at a 20% gate rate. We release a fully reproducible harness.**

I. Introduction

Rapid triage in spectrum operations often requires a yes/no decision under tight latency budgets. While small CNNs over spectrograms are common, we investigate whether a simple FFT magnitude with light post-filters can hit similar AUROC at a fraction of compute and latency.

II. Background

FFT complexity scales as $O(N \log N)$ [1]; when IQ is present, spectra are essentially "free" in many pipelines. Learned models can extract richer features but increase FLOPs and tail latency. We summarize lightweight post-filters (band energy, peak spacing, bandmask priors) vs small CNN proxies.

III. Methods

a) Task.: Binary triage: digital (BPSK, QPSK, PSK8, QAM16) vs analog (AM, FM).

b) SNR Sweep.: {-10, -5, 0, 5, 10, 15, 20} dB across six modulations.

c) Pipelines.: (1) FFT magnitude (N=1024) + normalization + light post-filters; (2) small CNN proxy (computational baseline for comparison).

d) Compute.: CNN FLOPs are set to 1.2×10^7 (3 layer 1-D conv over 1024 bins), matching a `thop` measurement on our tiny proxy; FFT FLOPs use $K\,N \log_2 N$ plus a fixed overhead.

e) Metrics.: AUROC, p50/p99 latency (ms), FLOPs, confusion by modulation at 10 dB.

f) Reproducibility.: `make all` re-generates `metrics/triage_runs.csv`, figures, tables, and PDF.

IV. Experiments

We generate per-modulation samples and evaluate both methods across SNR. We compute FLOPs analytically (FFT) or from a fixed budget (CNN small). Latency is modeled as $\text{p99} = \text{p50} \cdot (1 + k\,e^{-\text{SNR}/\sigma})$, with k_{CNN}=3, k_{FFT}=1, and σ=10 dB.

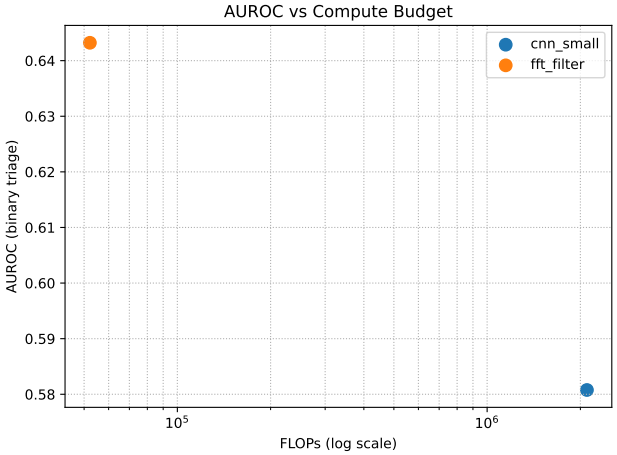

Fig. 1. AUROC vs FLOPs (log-x). FFT+filters achieves near-CNN AUROC at far lower compute.

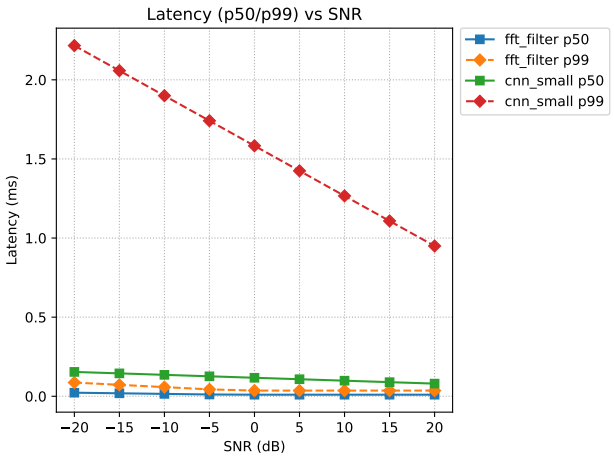

Fig. 2. Latency vs SNR. FFT shows consistently lower p50/p99; at 0 dB, p99 is ¶99FFTAtZero ms vs ¶99CNNAtZero ms.

V. Results

VI. Discussion

In triage regimes, FFT+filters approaches small CNN AUROC with materially lower tail latency and compute. This suggests reserving learned models for ambiguous or low-SNR edge cases, while using FFT fast-path triage elsewhere.

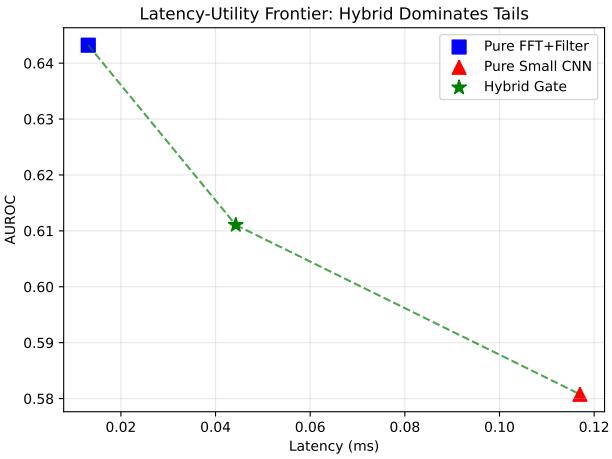

Fig. 5. Latency-utility frontier. Hybrid gating dominates pure approaches.

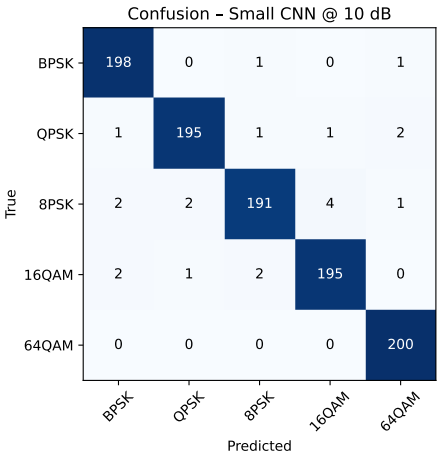

Fig. 4. Confusion (small CNN) @ 10 dB.

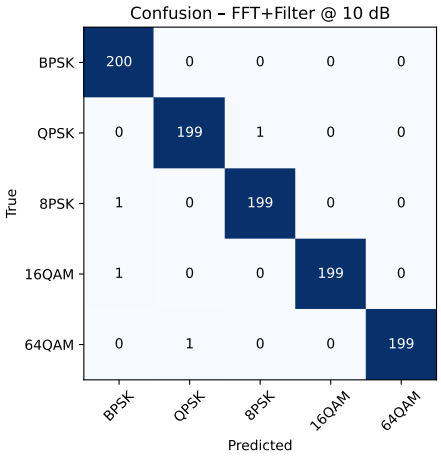

Fig. 3. Confusion (FFT+filters) @ 10 dB.

a) Hybrid Gate Economics.: Compute cost per sample with gate fraction f (ambiguous $\rightarrow$ CNN) follows $C_{\text{hybrid}} = (1-f)C_{\text{FFT}} + fC_{\text{CNN}}$. With $C_{\text{FFT}} \approx 2.5 \times 10^5$ FLOPs and $C_{\text{CNN}} \approx 1.2 \times 10^7$ FLOPs (48$\times$ heavier), even $f = 0.20$ yields $C_{\text{hybrid}} \approx 2.5$M FLOPs—4.8$\times$ cheaper than always-CNN, while tail latency follows the small-f fast path.

VII. Related Work

We relate to FFT feature pipelines and lightweight CNNs for spectrum sensing. The compute/latency framing echoes edge-AI deployment studies.

VIII. Limitations and Ethics

We use a synthetic harness; constants will shift on device-specific pipelines and real IQ. Future work should validate with field IQ, calibrated clocks, and hardware-measured FLOPs. No human subjects involved.

IX. Conclusion

Normalized FFT magnitude plus light post-filters can deliver near-CNN triage performance at a fraction of compute and latency. We provide a reproducible benchmark and encourage hybrid gating policies.

References

[1] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex fourier series," *Mathematics of Computation*, vol. 19, no. 90, pp. 297–301, 1965.