PIKing Neural Networks: Parallel Inference via Kernelizing. A Comparative Study of Tandem-Deployed Space AI Models Across Heterogeneous Hardware

Anonymous CVPR submission

Paper ID *****

Abstract

Autonomous Fault Detection, Isolation, and Recovery (FDIR) is critical for spacecraft operating under strict onboard resource constraints. While lightweight anomaly detection models are effective for telemetry monitoring, their co-execution with heavier perception networks remains underexplored. We study tandem deployment of an LSTM-based anomaly detector with other upstream models such as PyNAS and YOLO11s for Earth Observation and Autonomous Navigation respectively. These models are deployed in tandem across five heterogeneous platforms, evaluating sequential and intermittent execution to reflect supervisory FDIR scheduling. To do this, we introduce the concept of PIKing (Parallel Inference via Kernelizing) the models. Using measured latency, computed FLOPs, and empirical roofline analysis, we show that tandem behavior is strongly hardware-dependent: accelerator-backed systems exhibit near-additive performance, whereas CPU-bound platforms are sensitive to burst-triggered contention. Introducing lightweight execution gaps significantly stabilizes intermittent performance without modifying model architecture, providing practical guidance for resource-constrained space deployment scenarios.

1. Introduction

Modern spacecraft increasingly rely on onboard AI to enable real-time perception and autonomous decision-making under tight power, mass, and thermal budgets [1]. In parallel, learning-based FDIR pipelines, particularly LSTM-based anomaly detection, have demonstrated strong capability for modeling nominal telemetry and identifying out-of-ordinary operations [2, 4]. However, in realistic autonomous systems, supervisory FDIR models must execute alongside heavier primary workloads (e.g., object detection or segmentation) on heterogeneous edge hardware. While prior work has focused on model accuracy, compression, or standalone hardware efficiency [1, 4], the interaction between lightweight FDIR models and primary perception networks under tandem deployment remains largely unquantified. In this paper, we investigate sequential and intermittent tandem execution of an LSTM-based anomaly detector[4] with a lightweight EO (Earth Observation) model PyNAS[1] and YOLO11s used in autonomous navigation scenarios[6], across five compute platforms, analyzing latency, throughput, and empirical roofline behavior to understand hardware-dependent co-execution dynamics relevant to autonomous FDIR (Fault Detection, Isolation and Repair) in space systems.

2. Methodology

The methodology describes the models, deployment strategies, and hardware platforms used to evaluate concurrent perception and supervisory FDIR workloads under representative spaceborne compute constraints.

2.1. Problem Setting

We study the *tandem deployment* of a lightweight LSTM-based anomaly-detection model (representing supervisory FDIR) alongside a heavier primary perception model (representing EO or navigation workloads) under heterogeneous edge-compute constraints. The objective is to quantify latency and compute interactions when both workloads execute either continuously or intermittently, reflecting realistic in-orbit scheduling scenarios.

Let f_p denote the primary model and f_a the anomaly detection model. We instantiate f_p using two architectures representative of onboard perception pipelines:

- **PyNAS** (7-channel, 256×256 input)[7], representing hardware-aware EO segmentation.
- **YOLO11s** (3-channel, 640×640 input)[6], representing real-time object detection or navigation perception.

The anomaly detector f_a is an LSTM autoencoder trained to reconstruct temporal telemetry sequences and detect deviations via reconstruction error, modeling lightweight onboard FDIR.

Our choice of models thus reflects three core areas of AI applications in space namely FDIR, EO and Autonomous Navigation.

All experiments use fixed input shapes and batch size $B = 1$ to emulate real-time onboard inference.

2.2. Model Compute Characterization

To enable hardware-aware comparison, we explicitly compute per-inference floating point operations (FLOPs) directly from exported models.

We denote:

- FLOPs_p for the primary model (YOLO: 21,548,492,800; PyNAS: 1,039,927,378),
- FLOPs_a for the anomaly model (1,645,800)[5].

These values remain constant per inference and allow us to derive empirical throughput (GFLOPs/s) under each deployment configuration, enabling consistent cross-hardware comparison for space AI workloads.

2.3. Tandem Deployment Strategies

We evaluate two deployment strategies reflecting continuous perception and supervisory health monitoring in orbit.

2.3.1. Sequential Tandem Deployment

In the sequential configuration, the primary model f_p (EO/navigation) and anomaly model f_a (FDIR) execute back-to-back for every input sample. The representation from f_p is immediately passed to f_a, modeling continuous health supervision during sustained perception.

Although total latency might be expected to equal the sum of standalone latencies, real systems often exhibit non-linear effects due to scheduling, memory reuse, and accelerator pipeline behavior [3]. We therefore measure whether concurrent EO and FDIR execution is strictly additive, incurs interference, or exhibits negligible interaction across hardware tiers.

2.3.2. Intermittent (Triggered) Deployment

In the intermittent configuration, the anomaly model is triggered only after a stream of primary inferences, operating at a lower temporal frequency. This model's realistic onboard FDIR behavior, where anomaly detection serves as a supervisory layer invoked periodically rather than per frame.

Beyond reducing average compute load, this setup allows evaluation of burst-triggered FDIR execution during sustained EO processing. By comparing standalone, sequential, and intermittent deployment, we quantify the impact of supervisory integration on perception performance under heterogeneous edge hardware typical of space systems.

2.4. Experimental Platforms

We evaluate all configurations across five heterogeneous compute platforms representative of candidate spaceborne AI hardware:

- **NVIDIA Tesla T4 GPU** (data center reference),
- **NVIDIA Jetson Orin Nano Super** (embedded GPU),
- **Axelera Metis** on Raspberry Pi 5 (NPU accelerator),
- **Hailo-8L** on Raspberry Pi 5 (NPU accelerator),
- **Raspberry Pi 5 CPU-only**.

This selection enables comparison across: (1) general-purpose GPUs, (2) embedded GPUs, (3) NPU-based accelerators, and (4) CPU-only edge systems.

All models are executed using device-native runtimes (e.g., TensorRT, ONNX Runtime, vendor SDKs) to reflect realistic deployment pipelines.

For each configuration we report:

- Latency (ms),
- GFLOPs/s,
- Tandem deltas relative to standalone baselines.

3. Results and Analysis

Table 1 summarizes latency and throughput across five heterogeneous platforms representative of in-orbit AI deployment. We evaluate hardware behavior under continuous EO and navigation workloads with integrated supervisory FDIR, and analyze how scheduling strategies influence performance under resource constraints.

3.1. Sequential Tandem Behavior

Under continuous EO-style workloads, sequential tandem execution is largely near-additive and in several cases slightly beneficial. On Jetson Orin Nano Super, PyNAS latency improves by 2.37% with a 2.43% GFLOP/s gain, while YOLO improves by 2.04% in latency and 2.08% in throughput. On Tesla T4, PyNAS latency decreases by 0.88% with a 0.89% throughput gain, indicating minimal interference between perception and supervisory FDIR tasks. Axelera similarly shows a 4.23% latency reduction and 4.42% throughput gain for YOLO, demonstrating stable back-to-back execution suitable for sustained EO pipelines.

For the anomaly model representing lightweight FDIR, improvements are also observed in multiple cases (e.g., -3.35% latency and +3.46% GFLOP/s on Jetson under PyNAS). Although isolated degradations occur (e.g., -27.04% GFLOP/s on T4), absolute latency remains small relative to onboard timing budgets. Overall, sequential deployment suggests that accelerator-backed platforms can sustain continuous EO or Autonomous Navigation workloads with integrated FDIR supervision without substantial penalty.

3.2. Intermittent Deployment Dynamics

Intermittent FDIR triggering, representing periodic health monitoring during EO or navigation, introduces greater variability, particularly on CPU-only systems. On Raspberry Pi 5, anomaly trigger latency increases from 2.079 ms

Regimen	Composition	Model/Config	Axelera Metis				Jetson Orin Nano Super				Raspberry Pi 5				Pi 5 + Hailo-8L				Tesla T4			
			Lat	Δ	GF/s	Δ	Lat	Δ	GF/s	Δ	Lat	Δ	GF/s	Δ	Lat	Δ	GF/s	Δ	Lat	Δ	GF/s	Δ
Alone	Sole Model	Pynas					2.483	0	418.849	0	23.855	0	43.595	0	3.653	0	284.661	0	1.192	0	872.151	0
		Yolo	30	0	953.333	0	12.66	0	1702.072	0	508.727	0	42.358	0	26.728	0	806.208	0	12.195	0	1767.05	0
		Anomaly	13.52	0	31.164	0	3.253	0	32.385	0	2.079	0	25.328	0	1.859	0	28.337	0	25.419	0	16.575	0
Tandem Seq	pynas + anomaly	Primary (tandem)					2.424	-2.37	429.009	2.43	19.04	-20.18	54.617	25.28	3.535	-3.24	294.185	3.35	1.182	-0.88	879.913	0.89
		Anomaly (tandem)					3.144	-3.35	33.506	3.46	1.918	-7.78	27.465	8.44	1.808	-2.72	29.129	2.79	18.546	-27.04	22.717	37.06
		Anomaly (gap)					3.045	-6.37	34.589	6.81	1.778	-14.47	29.613	16.92	1.785	-3.97	29.51	4.14	18.546	-27.04	22.717	37.06
	yolo + anomaly	Primary (tandem)	28.731	-4.23	995.451	4.42	12.402	-2.04	1737.526	2.08	468.147	-7.98	46.029	8.67	26.701	-0.1	807.033	0.1	12.374	1.47	1741.494	-1.45
		Anomaly (tandem)	12.509	-7.48	33.682	8.08	3.553	-3.35	29.642	3.46	2.86	-7.78	18.413	8.44	1.808	-2.72	29.126	2.79	18.546	-27.04	22.717	37.06
		Anomaly (gap)	12.509	-7.47	33.681	8.08	3.437	-6.37	30.65	6.81	2.773	-14.47	18.994	16.92	1.785	-3.97	29.285	4.14	18.546	-27.04	22.717	37.06
Tandem Dep	pynas + anomaly	Stream					2.397	-3.44	433.786	3.57	24.53	2.83	42.395	-2.75	3.558	-2.61	292.291	2.68	1.193	0.01	872.023	-0.01
		Trigger					3.603	10.77	29.235	-9.72	5.948	186.04	8.855	-65.04	2.89	55.47	18.226	-35.68	18.546	-27.04	22.717	37.06
		Trigger (gap)					3.158	-2.9	33.353	2.99	2.964	42.52	17.771	-29.83	2.361	27.02	22.309	-21.27	18.546	-27.04	22.717	37.06
	yolo + anomaly	Stream	29.302	-2.33	976.029	2.38	11.762	-7.09	1832.051	7.64	585.616	15.11	36.796	-13.13	27.894	4.36	772.508	-4.18	11.669	-4.31	1846.636	4.5
		Trigger	40.163	197.07	10.49	-66.34	3.319	10.77	31.739	-9.72	9.626	186.04	5.471	-65.04	2.874	55.47	18.325	-35.68	18.546	-27.04	22.717	37.06
		Trigger (gap)	33.138	145.11	12.714	-59.2	3.199	-2.9	32.926	2.99	10.756	42.52	4.896	-29.83	2.428	27.02	21.692	-21.27	18.546	-27.04	22.717	37.06

Table 1. Tandem deployment results across five platforms. Δ columns report percent change relative to standalone baselines. Due to compatibility constraints, YOLOv8s is deployed on Axelera-Metis and LSTM inference on Hailo-8L is executed on the Pi CPU.

to 5.948 ms (+186.04%) under PyNAS, with a 65.04% throughput drop. Similar penalties are observed under YOLO-triggered anomaly execution.

In contrast, accelerator-backed platforms exhibit smaller deviations. On Jetson Orin Nano Super, anomaly trigger latency increases by 10.77% with a -9.72% throughput change, while Tesla maintains stable absolute performance despite percentage shifts driven by its high baseline throughput. These results indicate that hardware architecture strongly influences the feasibility of integrating burst-triggered FDIR within sustained EO pipelines.

3.3. Effect of Gap Insertion

Introducing a lightweight scheduling gap between EO stream processing and anomaly triggering consistently stabilizes intermittent performance. On Raspberry Pi 5 (PyNAS), inserting a gap reduces anomaly latency from 5.948 ms to 2.964 ms and improves throughput degradation from -65.04% to -29.83%, effectively halving the penalty. On Jetson Orin Nano Super, the gap reduces anomaly latency from 3.603 ms to 3.158 ms and converts a -9.72% throughput drop into a +2.99% gain.

Collectively, these findings show that while accelerator-backed platforms can support concurrent EO, navigation, and lightweight FDIR workloads predictably, CPU-only systems are highly sensitive to burst-induced contention and benefit significantly from minimal scheduling separation during autonomous space operation.

3.4. Roofline Analysis

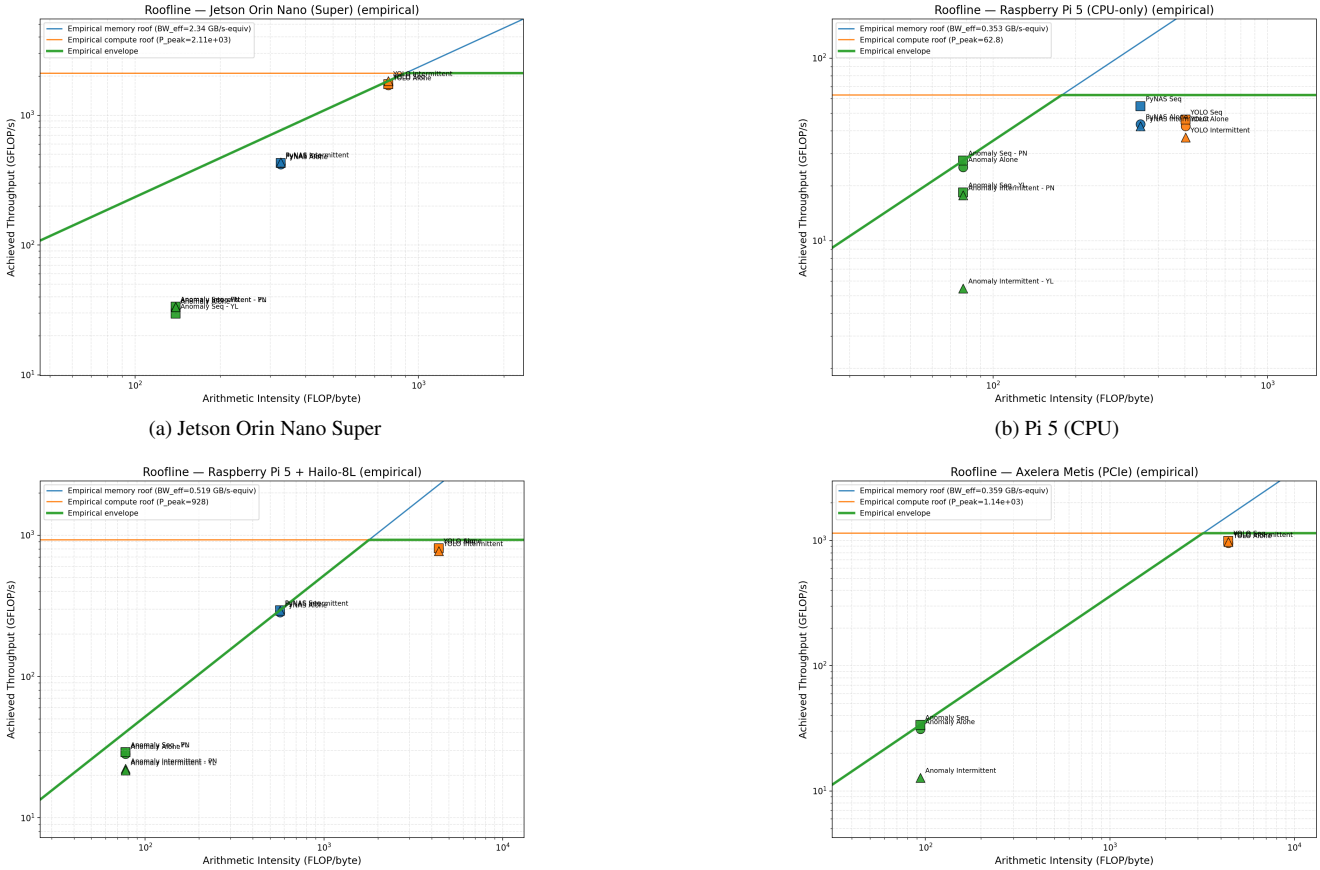
We employ an empirical roofline analysis to relate achieved throughput (GFLOP/s) to arithmetic intensity (AI), defined as FLOPs per byte transferred. Instead of using vendor peak specifications, device-specific memory and compute ceilings are derived directly from measured data, yielding a practical roofline envelope for each platform Figure 1 shows the roofline analysis for all four edge devices.

Across devices, the primary models (YOLO11s and PyNAS) operate at moderate-to-high arithmetic intensity and approach the empirical compute roof, indicating largely compute-bound execution. In contrast, the LSTM-based anomaly model consistently resides in the memory-bound regime due to its lower arithmetic intensity. Under intermittent deployment, arithmetic intensity remains unchanged while achieved throughput shifts downward, particularly on CPU-constrained systems, indicating reduced effective utilization caused by burst scheduling rather than intrinsic model complexity.

Overall, the roofline analysis indicates that tandem inefficiencies arise from execution dynamics and resource contention, not from computational density, an important consideration for resource-constrained AI4Space systems.

3.5. Conclusion

Sequential tandem execution is largely near-additive and often slightly beneficial, yielding 2–4% latency reductions and corresponding GFLOP/s gains on accelerator-backed systems. In contrast, intermittent anomaly triggering incurs significant penalties on CPU-only platforms, up to +186% latency and -65% throughput, whereas it remains comparatively modest on GPUs and NPUs. Introducing lightweight execution gaps substantially mitigates these penalties, reducing anomaly latency by more than 50% on CPU systems and stabilizing throughput across devices. Overall, tandem interaction dynamics are strongly hardware-dependent: accelerator-class platforms maintain predictable co-execution behavior, whereas CPU-dominated systems are highly sensitive to burst-induced contention.



(a) Jetson Orin Nano Super

(b) Pi 5 (CPU)

(c) Pi 5 + Hailo-8L

(d) Axelera Metis

Figure 1. Roofline characterization of tandem deployment across heterogeneous platforms. Each point represents a model-regimen pair plotted as achieved throughput (GFLOP/s) versus arithmetic intensity (FLOP/byte). The empirical envelope is derived from measured effective bandwidth and peak throughput.

References

[1] Roberto Del Prete, Parampuneet Kaur Thind, Andrea Mazzeo, Matthew Whitley, Lorenzo Papa, Nicolas Longépé, and Gabriele Meoni. Optimizing deep learning models for on-orbit deployment through neural architecture search. *Scientific Reports*, 15(1):37783, 2025. 1

[2] Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 387–395, 2018. 1

[3] Myeongjae Jeon, Shivaram Venkataraman, Amar Phanishayee, Junjie Qian, Wencong Xiao, and Fan Yang. Analysis of {Large-Scale}{Multi-Tenant}{GPU} clusters for {DNN} training workloads. In *2019 USENIX Annual Technical Conference (USENIX ATC 19)*, pages 947–960, 2019. 2

[4] James Murphy, Muhammad Deedahwar Mazhar Qureshi, Jake O'Brien, and Brian Mac Namee. Autonomous satellite health monitoring using eirsat-1 telemetry. In *AICS*, pages 279–289, 2024. 1

[5] James Murphy, Fangyijie Wang, Muhammad Deedahwar Mazhar Qureshi, and Brian Mac Namee. From ground to orbit: Enhancing satellite autonomy with ai-powered anomaly detection. 2024. 2

[6] Jake O'Brien, Gerard Lacey, and James Murphy. A closed-loop vision system framework for autonomous spacecraft rendezvous. In *Proceedings of the Irish Machine Vision and Image Processing Conference 2025*, page 244, Ulster University, Derry, Northern Ireland, 2025. Irish Pattern Recognition and Classification Society. 1

[7] Parampuneet Kaur Thind, Roberto Del Prete, Matthew Whitley, Andrea Mazzeo, Nicolas Longépé, and Gabriele Meoni. Naseo: Neural architecture search for earth observation on-board processing. In *2025 European Data Handling & Data Processing Conference (EDHPC)*, pages 1–8. IEEE, 2025. 1