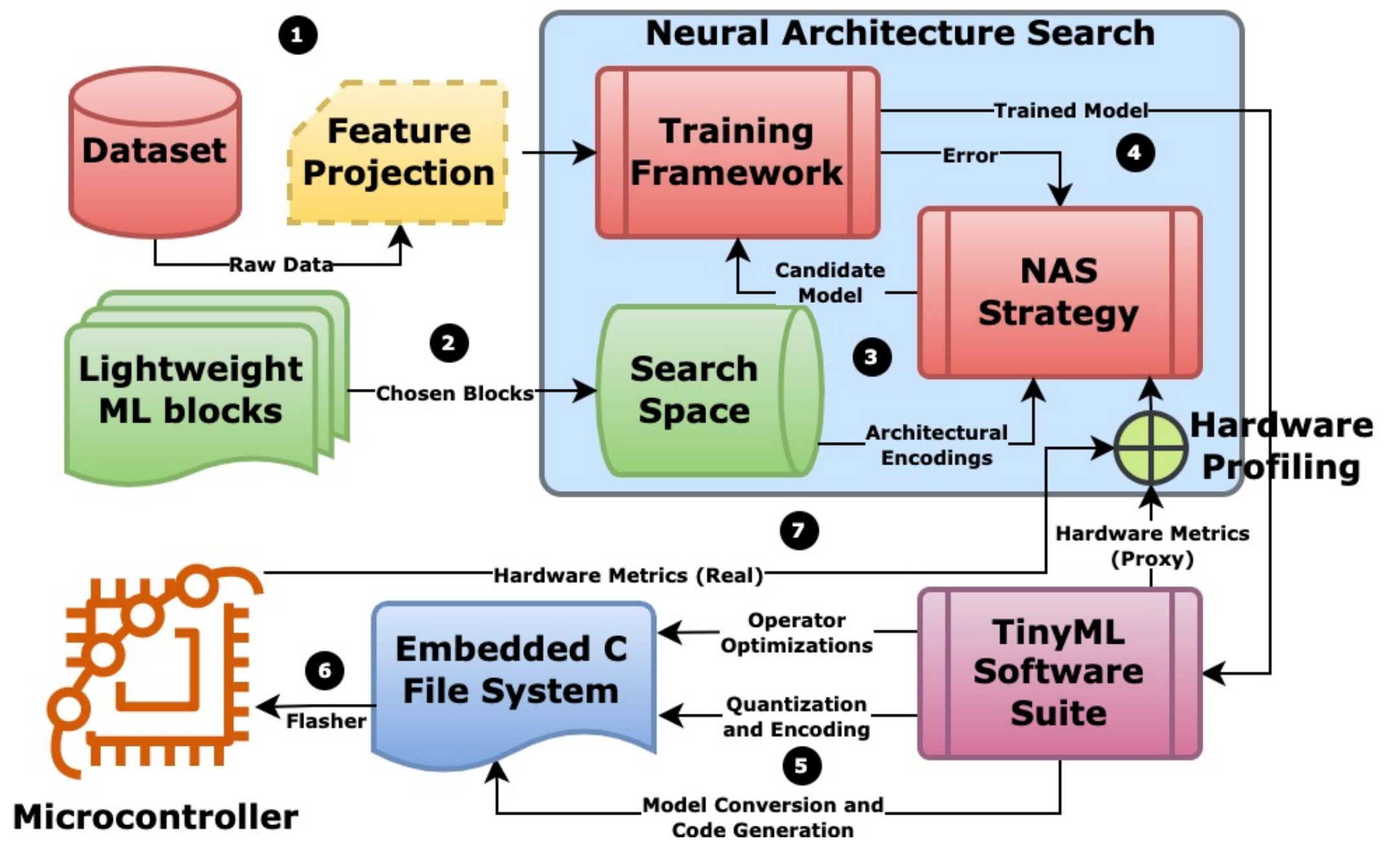
THIN-Bayes: Platform-Aware Machine Learning for Low-End IoT Devices

Swapnil Sayan Saha*, Sandeep Singh Sandha*, Mohit Aggarwal^, and Mani Srivastava*

*University of California, Los Angeles; ^ARM Research, Austin

Introduction

- Neural Architecture Search: Integral component of the first-generation TinyML workflow.



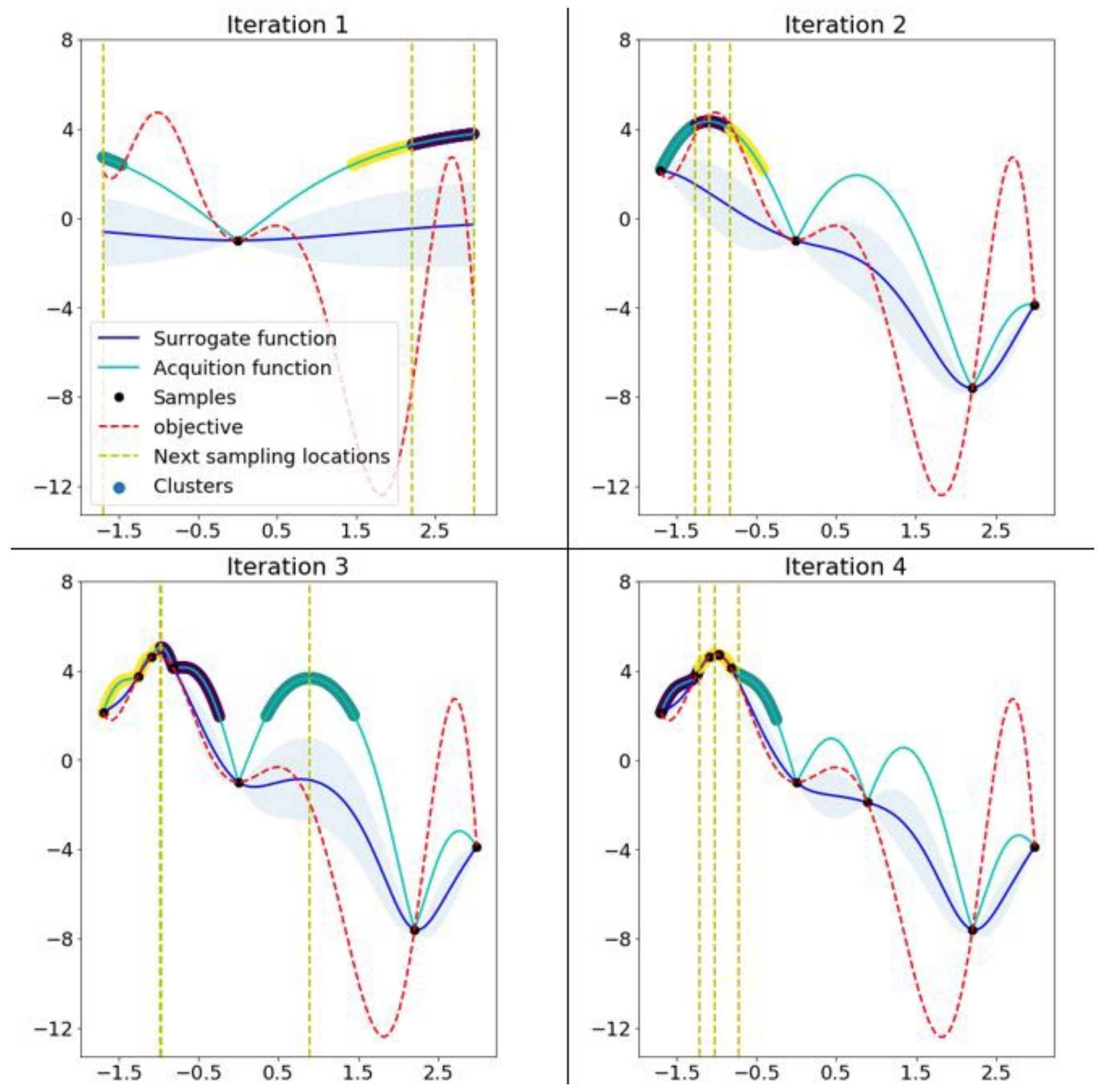
Challenges of Adopting Existing Neural Architecture Search

- Existing frameworks for low-end IoT devices: SpArSe, MCUNet, MicroNets, and μNAS.
- Lack of open-source tools.
- Use of coarse or inaccurate hardware metrics / proxies.
- Problematic formulation – inability to handle loss contour discontinuities and categorical variables; assumes usage of only CNN and MLP for toy applications.
- Long convergence time and requires expensive compute infrastructure.

MANGO: Fast, Parallel and Gradient-free Bayesian Optimizer

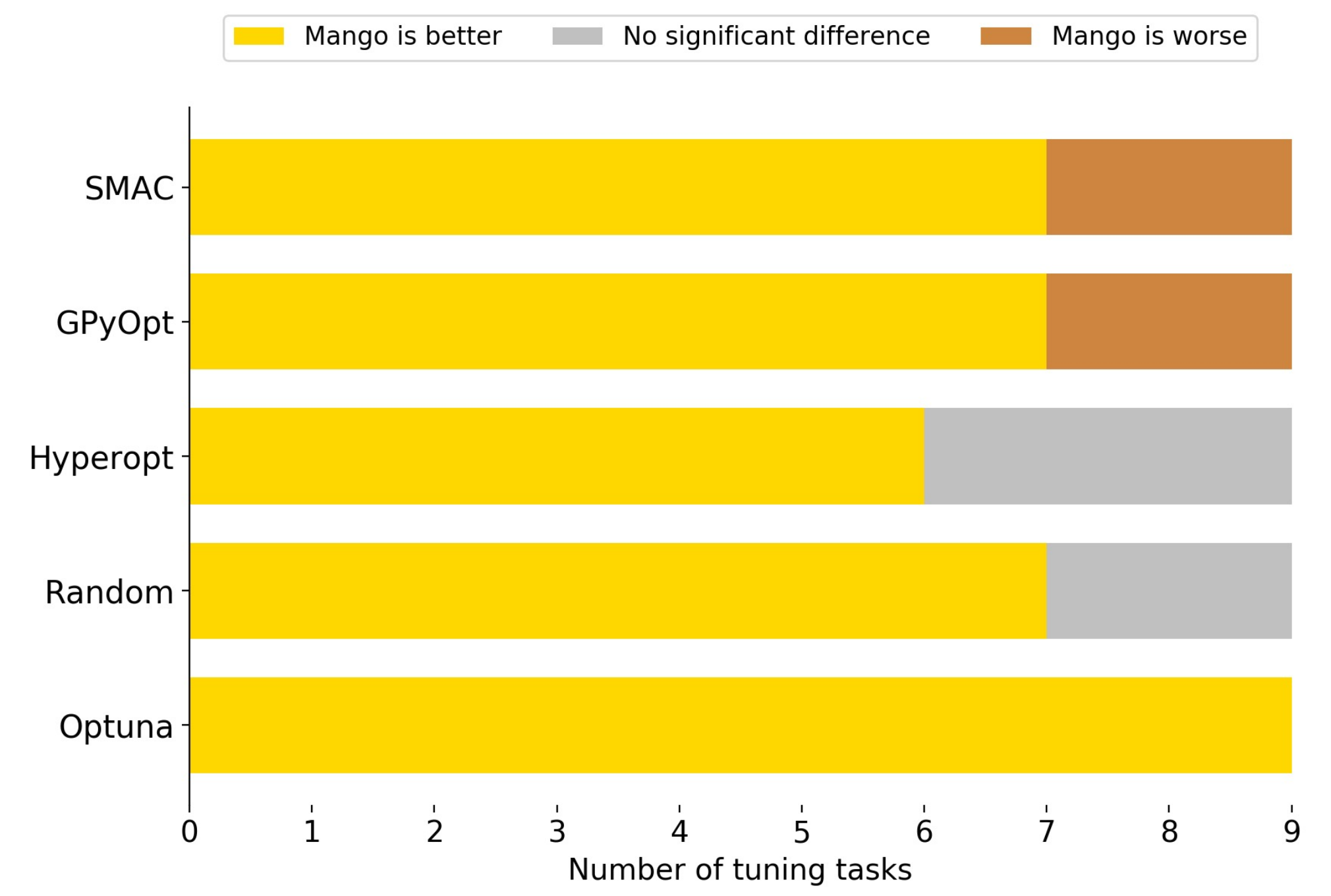
- **Mango:** A new state-of-the-art optimizer.
- **Scalability:** Outperforms current parallel searches.
- **Fault-Tolerance:** Detects failures at the application layer.
- Supports **categorical & continuous** search spaces.
- **Compatible** with SciPy and Scikit-learn.
- **Open-source** and **expandable**.

Visualizing Parallel Optimization in Mango



- Mango is adopted in commercial IC design at Arm.
- Mango is 45% faster over previous deployments at Arm.

Mango vs Others on 9 ML Classifier Tasks



Neural Architecture Search Formulation

$$f_{\text{opt}} = \lambda_1 f_{\text{error}}(\Omega) + \lambda_2 f_{\text{flash}}(\Omega) + \lambda_3 f_{\text{SRAM}}(\Omega) + \lambda_4 f_{\text{latency}}(\Omega)$$

$$f_{\text{error}}(\Omega) = \mathcal{L}_{\text{validation}}(\Omega), \Omega = \{\{V, E\}, w, \theta, v\}$$

$$f_{\text{flash}}(\Omega) = \begin{cases} -\frac{||h_{\text{FB}}(w,\{V,E\})||_0}{\text{flash}_{\text{max}}} \vee -\frac{\text{HIL information}}{\text{flash}_{\text{max}}} \\ \infty, f_{\text{flash}}(\Omega) > \text{flash}_{\text{max}} \end{cases}$$

$$f_{\text{latency}}(\Omega) = \frac{\text{FLOPS}}{\text{FLOPS}_{\text{target FLOPS}}} \vee \frac{\text{HIL information}}{\text{Latency}_{\text{target latency}}}$$

$$f_{\text{SRAM}}(\Omega) = \begin{cases} -\frac{\max_{l \in [1,L]} \{||x_l||_0 + ||a_l||_0\}}{\text{SRAM}_{\text{max}}} \vee -\frac{\text{HIL information}}{\text{SRAM}_{\text{max}}} \\ \infty, f_{\text{SRAM}}(\Omega) > \text{SRAM}_{\text{max}} \end{cases}$$

$$a = w \vee y, \qquad y = \sum_{k=1}^{K} v_k g_k(x, w_k)$$

$$\hat{f}(\Omega) \sim \mathcal{GP}(\mu(\Omega), k(\Omega, \Omega'))$$

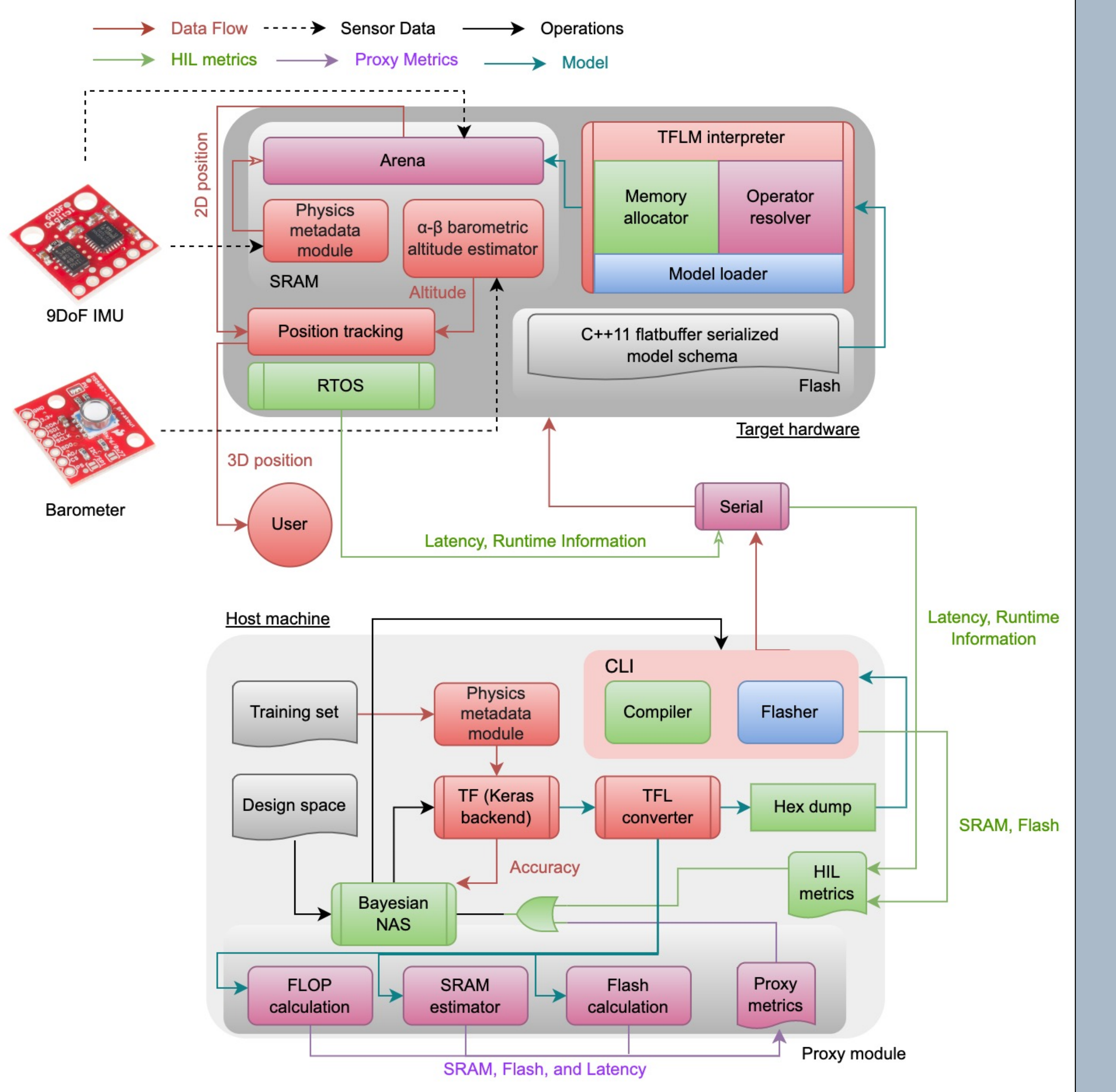
$$\Omega_t = \arg\max_{\Omega}(\mu_{t-1}(\Omega) + \beta^{0.5}\sigma_{t-1}(\Omega))$$

Qualitative Comparison Against Other Frameworks

Method	Search Strategy	Profiler	Tested Models	Optimization Parameters	Open-Source
SpArSe	Gradient-driven Bayesian	Analytical	CNN, MLP	Error, SRAM, Flash	No
MCUNet	Evolutionary	Lookup tables, prediction models	CNN, MLP	Error, SRAM, Flash, Latency	No
MicroNets	DNAS	Analytical	CNN, MLP	Error, SRAM, Flash, Latency	No
μNAS	Evolutionary	Analytical	CNN, MLP	Error, SRAM, Flash, Latency	Yes
THIN-Bayes	Gradient-free Bayesian	Platform-in-the-loop, analytical	Any model using TFLM operators	Any scalar term	Yes

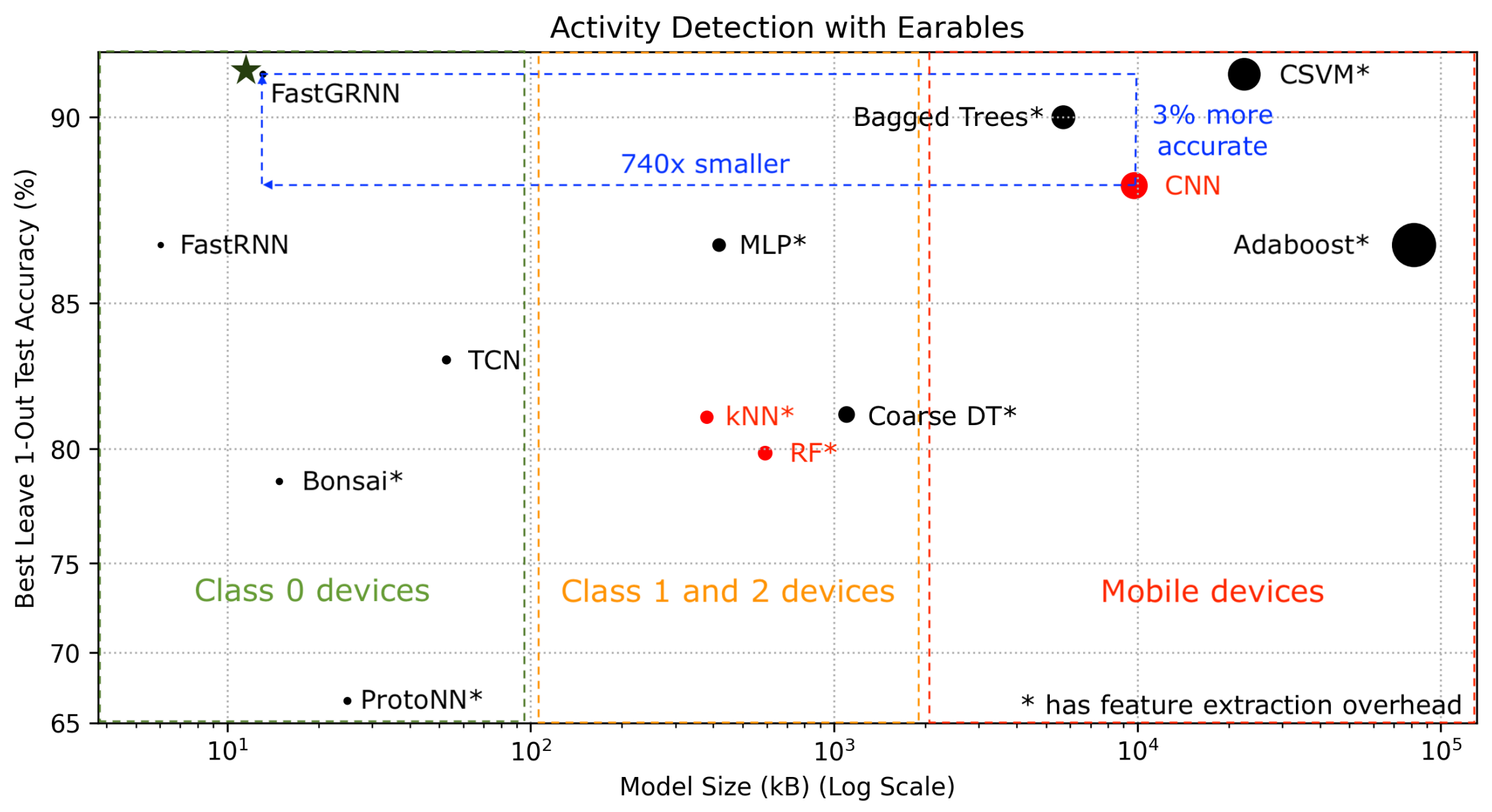
Example Implementation

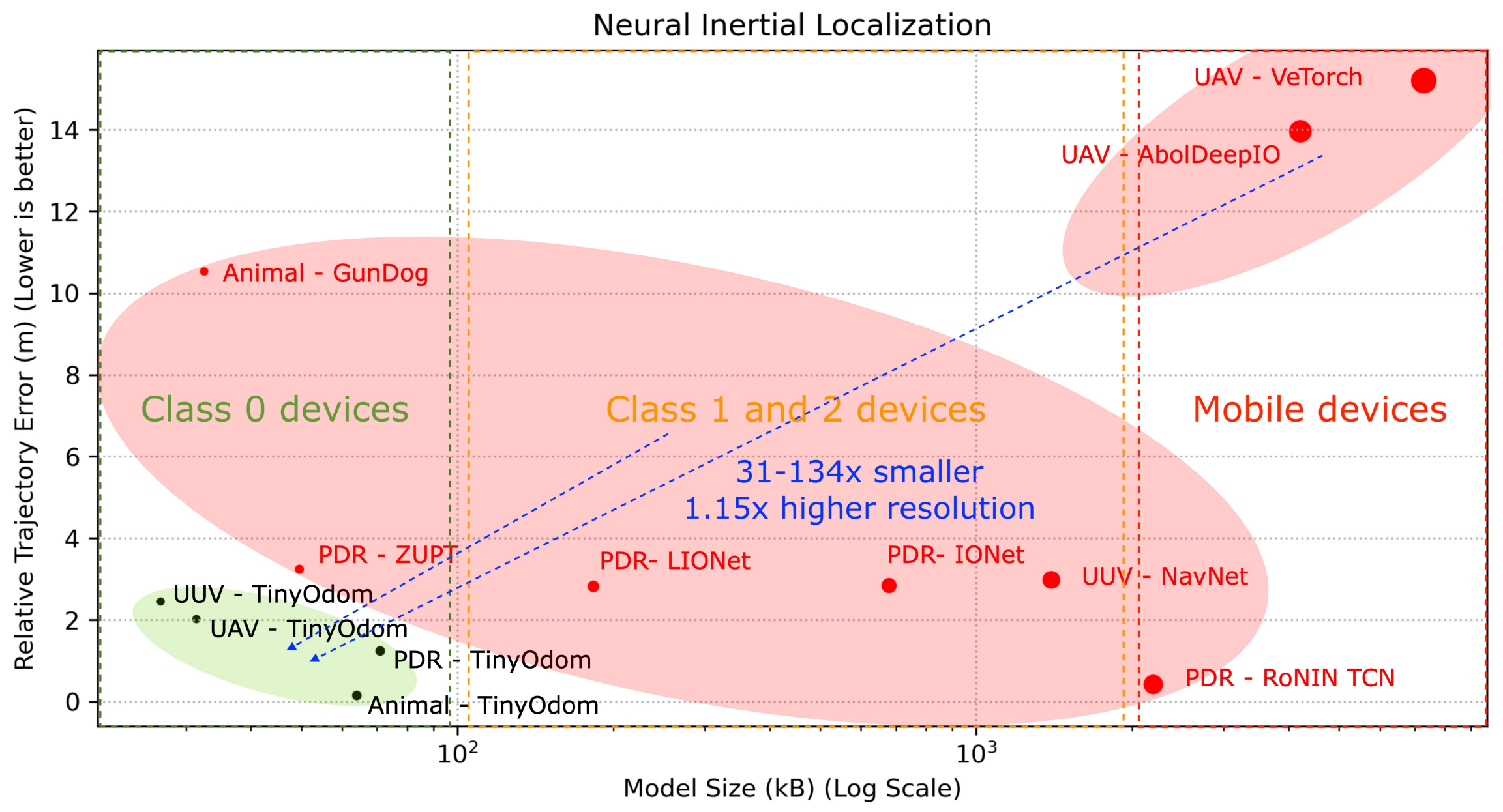
- Example implementation for ARM Cortex-M processors to perform neural inertial navigation, using TensorFlow Lite Micro as runtime interpreter and Mbed RTOS.



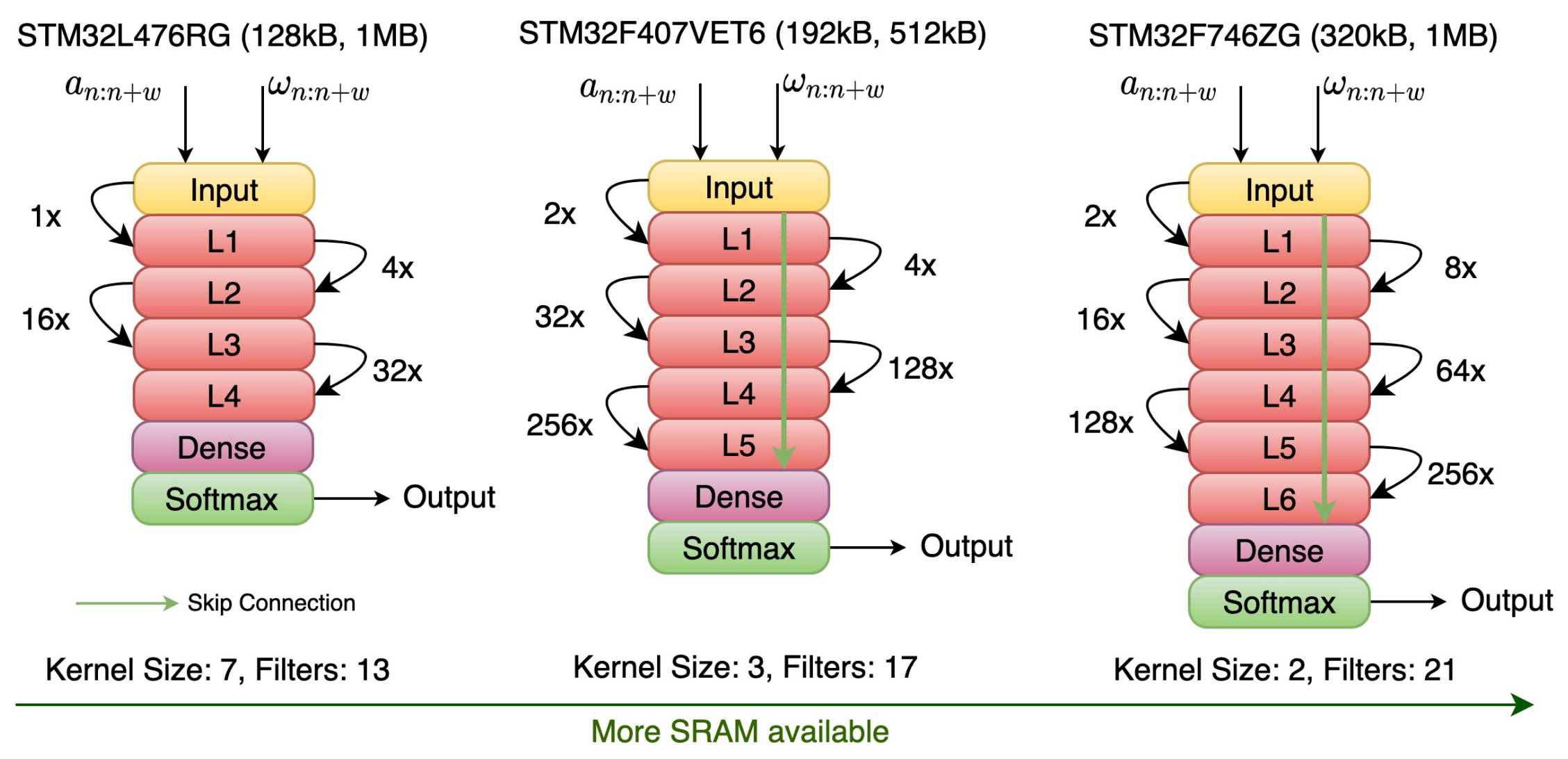
Evaluation – Neural Inertial Navigation and Activity Detection

- Lightweight models combined with our NAS provides state-of-the-art performance for making rich and complex inferences from temporal sensor data for challenging applications.



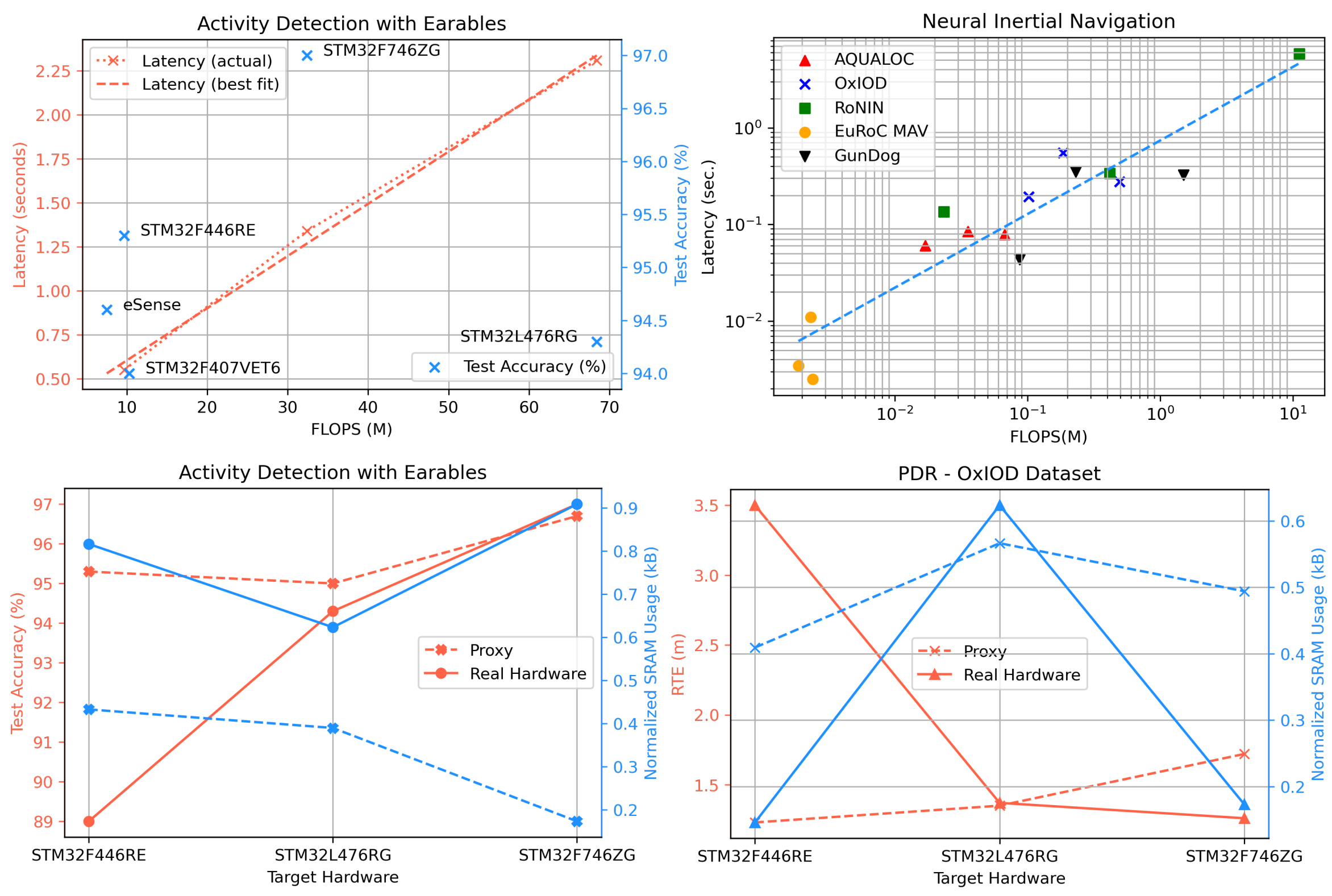


Evaluation - Architectural Adaptation



- Our NAS performs intelligent architectural adaptations to exploit full hardware capabilities in order to improve error.

Evaluation – Analytical Proxies Are Problematic



- SRAM and Flash proxies tend to overestimate HW constraints without considering dynamic runtime SW overhead or faults.
- FLOPS is not always proportional to latency.

Conclusion

- Our gradient-free Bayesian NAS framework supports usage of any lightweight models for challenging applications on any low-end IoT platform with arbitrary optimization parameters.
- Built over state-of-the-art optimizer, Mango, that is used in production pipelines.
- Focuses on application development; extendible by application developers without extensive domain knowledge.

References

1. Sandeep Singh Sandha, Mohit Aggarwal, Igor Fedorov, and Mani Srivastava. "Mango: A python library for parallel hyperparameter tuning." *International Conference on Acoustics, Speech and Signal Processing*, IEEE, 2020.
2. Sandeep Singh Sandha, Mohit Aggarwal, Swapnil Sayan Saha, and Mani Srivastava. "Enabling Hyperparameter Tuning of Machine Learning Classifiers in Production" *Third IEEE International Conference on Cognitive Machine Intelligence*. IEEE, 2021.
3. Swapnil Sayan Saha, Sandeep Singh Sandha, Luis Garcia, and Mani Srivastava, "TinyOdom: Hardware-Aware Efficient Neural Inertial Navigation," in *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, ACM New York, NY, USA, 2022. (under review)
4. Swapnil Sayan Saha, Sandeep Singh Sandha, Siyou Pei, Vivek Jain, Ziqi Wang, Yuchen Li, Ankur Sarker, and Mani Srivastava, "Auritus: An Open-Source Optimization Toolkit for Training and Deployment of Human Movement Models and Filters using Earables," in *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, ACM New York, NY, USA, 2022. (under review)
5. Swapnil Sayan Saha, Sandeep Singh Sandha, and Mani Srivastava, "Machine Learning for Microcontroller-Class Hardware – A Review", in *IEEE Sensors Journal*, 2022. (under review)