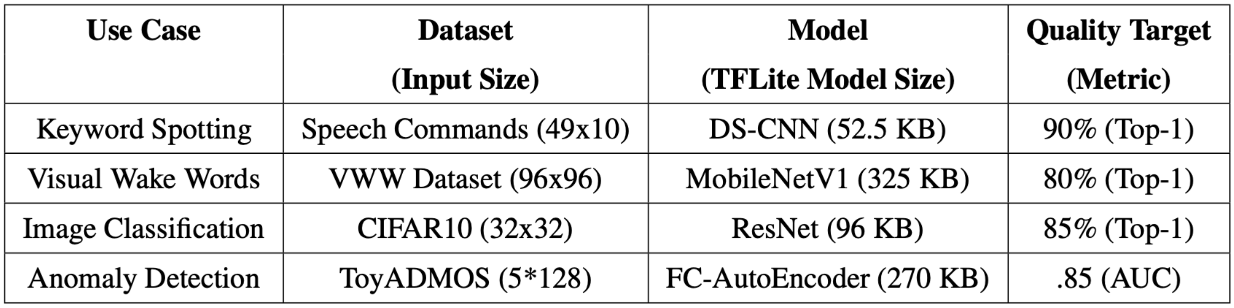
MLPerf Tiny

What is MLPerf Tiny

Now that we understand the importance of benchmarking for assessing performance, a natural question for us to ask is: what benchmarks exist for TinyML? This is where MLPerf comes in. MLPerf is a benchmark for ML performance, but has recently been extended to include MLPerf Tiny: a benchmark for measuring TinyML performance. Like any benchmark, it aims to provide fair comparisons between solutions by outlining a set of tasks and rules. MLPerf Tiny was created via a collaboration between numerous individuals from academia and industry who participated in a working group organized by MLCommons.org.

Use Cases

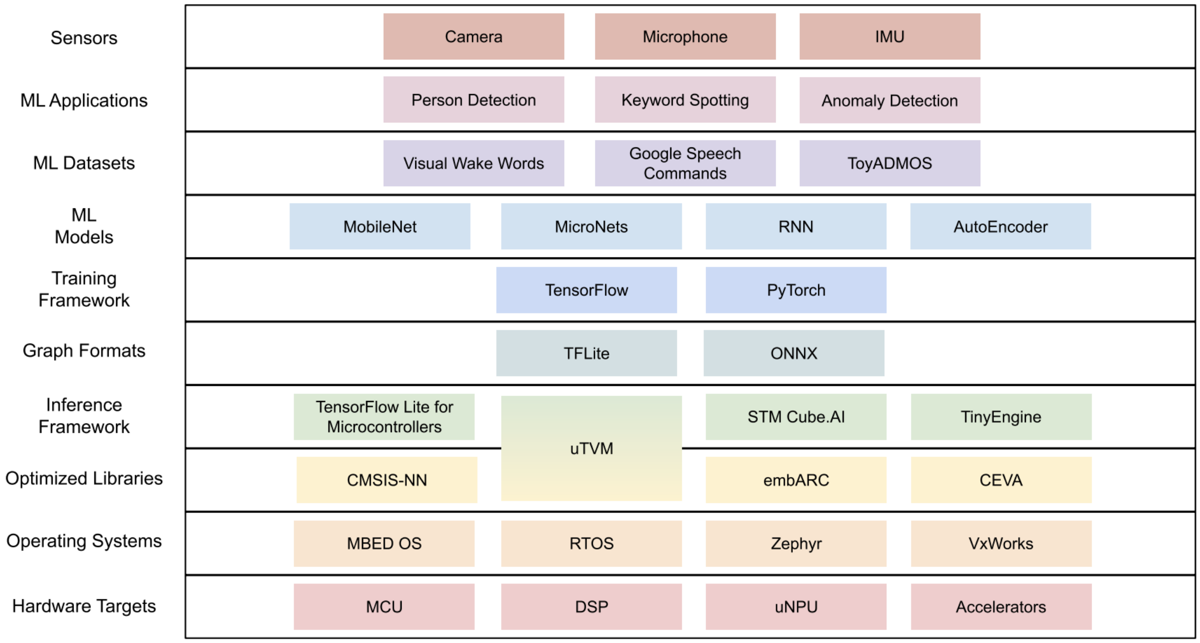
MLPerf Tiny currently supports four different ML application areas: keyword spotting, visual wake words, image classification, and anomaly detection. There are two submission windows per year, and two separate submission divisions that focus either on hardware or data-centric design. Each benchmark has a quality target, below which the submission will not be included.



Use Case	**Dataset** **(Input Size)**	**Model** **(TFLite Model Size)**	**Quality Target** **(Metric)**
Keyword Spotting	Speech Commands (49x10)	DS-CNN (52.5 KB)	90% (Top-1)
Visual Wake Words	VWW Dataset (96x96)	MobileNetV1 (325 KB)	80% (Top-1)
Image Classification	CIFAR10 (32x32)	ResNet (96 KB)	85% (Top-1)
Anomaly Detection	ToyADMOS (5*128)	FC-AutoEncoder (270 KB)	.85 (AUC)

MLPerf design: flexibility and representativeness

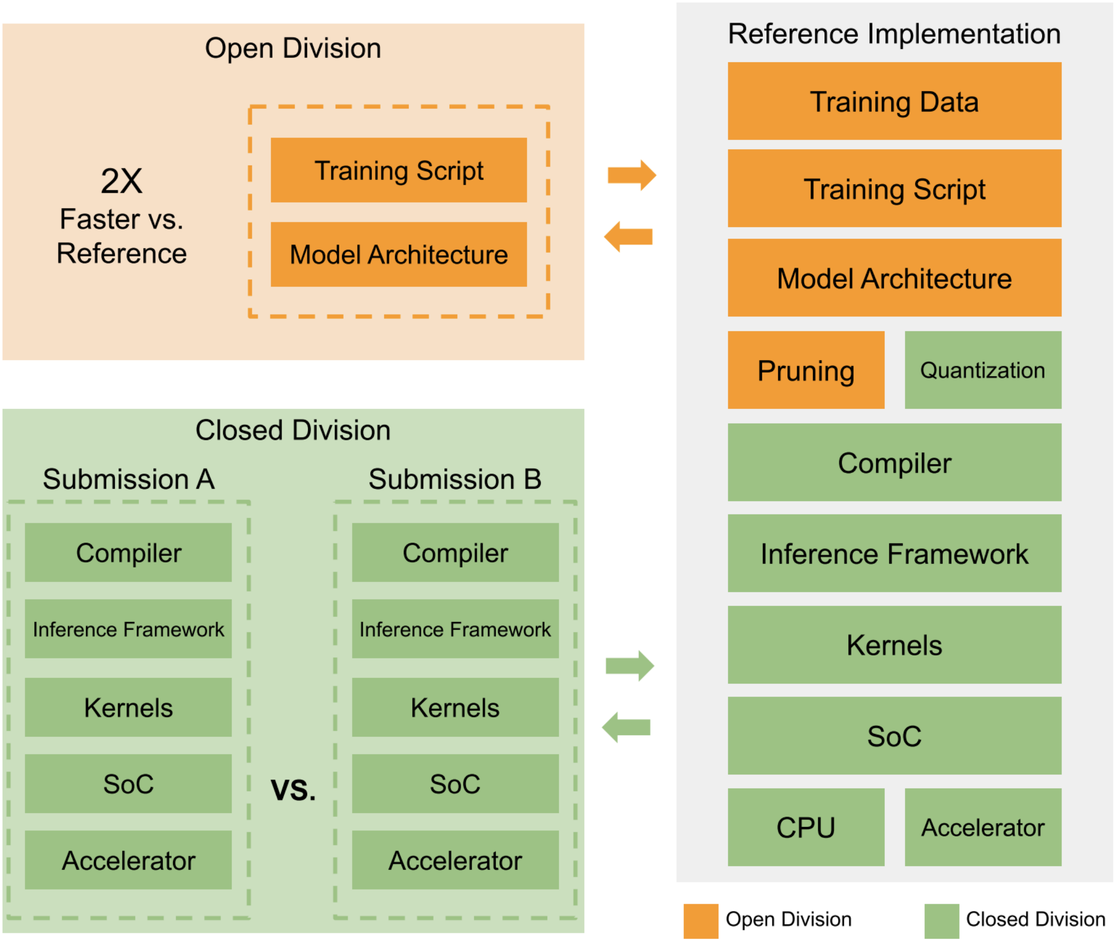
MLPerf Tiny is more than a set of datasets like ImageNet that allow people to compare classification accuracies. One of the things that makes benchmarking TinyML devices more challenging than traditional models is additional functional requirements, such as latency, memory footprint, and power consumption. MLPerf Tiny must keep track of all of these, including model performance, in order to suitably compare performance across systems, which can exhibit heterogeneity at all levels of the system.



Sensors	Camera	Microphone	IMU	
ML Applications	Person Detection	Keyword Spotting	Anomaly Detection	
ML Datasets	Visual Wake Words	Google Speech Commands	ToyADMOS	
ML Models	MobileNet	MicroNets	RNN	AutoEncoder
Training Framework	TensorFlow	PyTorch		
Graph Formats	TFLite	ONNX		
Inference Framework	TensorFlow Lite for Microcontrollers	uTVM	STM Cube.AI	TinyEngine
Optimized Libraries	CMSIS-NN		embARC	CEVA
Operating Systems	MBED OS	RTOS	Zephyr	VxWorks
Hardware Targets	MCU	DSP	uNPU	Accelerators

For example, let’s say that an individual used the Visual Wake Words dataset to perform person detection. Assume that they create a model using a specialized hardware platform that has a very high throughput compared to other submissions for the person detection. If our benchmark was purely focused on throughput (i.e., inferences per second), then even if the benchmark achieved 0% accuracy, it would appear to be the best model.

In reality, there is always some tradeoff between these parameters, and so MLPerf Tiny calculates accuracy, power consumption, and latency to allow these to be effectively compared. MLPerf Tiny also has two submission divisions, one which targets hardware (closed division) where only hardware parameters can be altered, and a second division (open division) focused on data-centric design, wherein the model architecture and training data can be altered.



How to get involved

MLPerf Tiny is managed by MLCommons and is currently on v0.7. The project is still in its early stages and is likely to evolve beyond what we have discussed here. For the interested reader, we have provided links to the MLPerf TinyML code and paper, and instructions on how to run the inference benchmark and provide submissions.

The code for MLPerf Tiny is available on GitHub: https://github.com/mlcommons/tiny
The benchmark design is describe in the paper: https://arxiv.org/pdf/2106.07597.pdf

If you are interested in running the benchmark, you should start by following the instructions on how to deploy the image classification reference submission. Next you would use EEMBCs EnergyRunner™ benchmark framework to connect to the system under test and run the benchmarks.