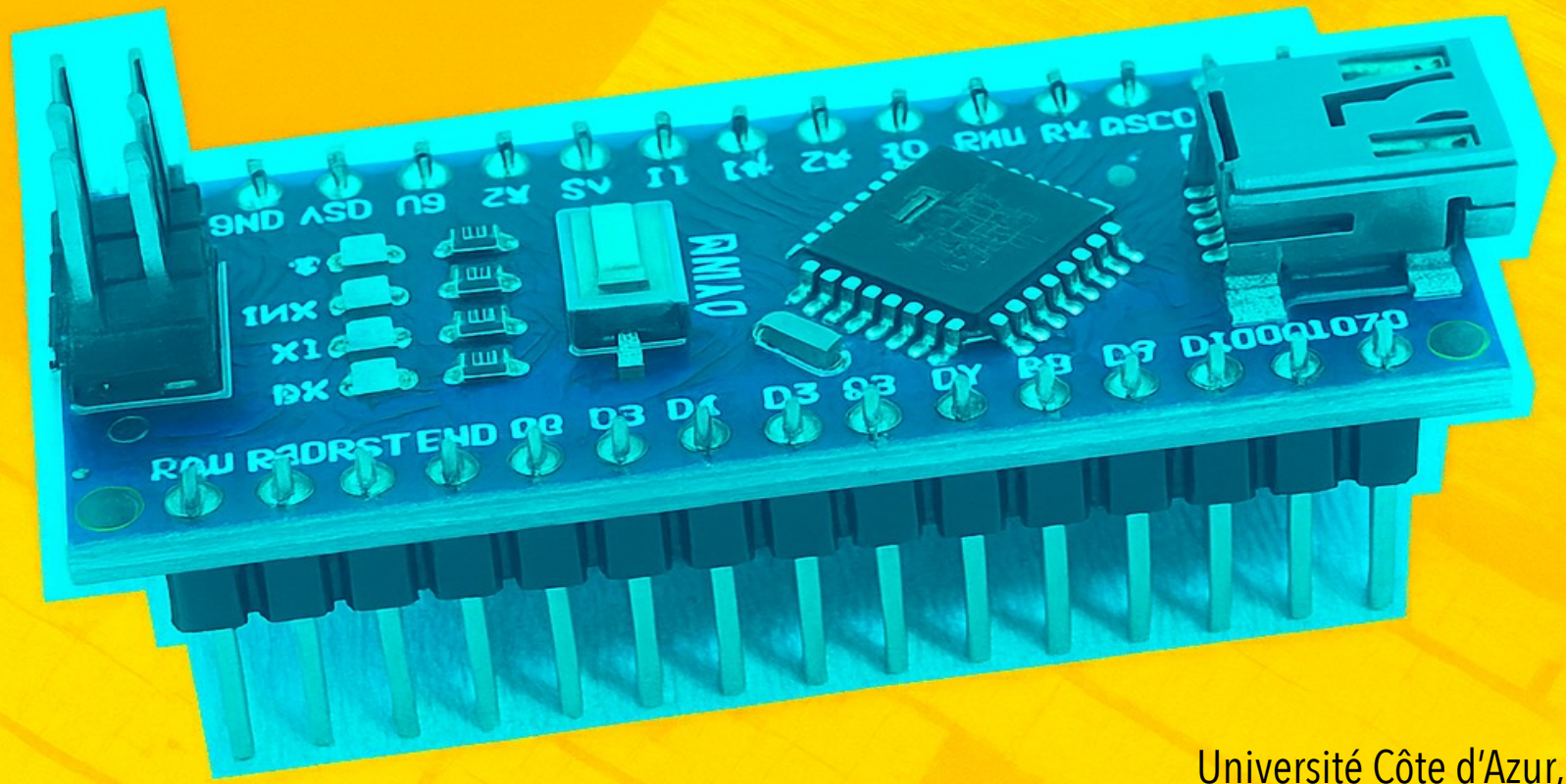
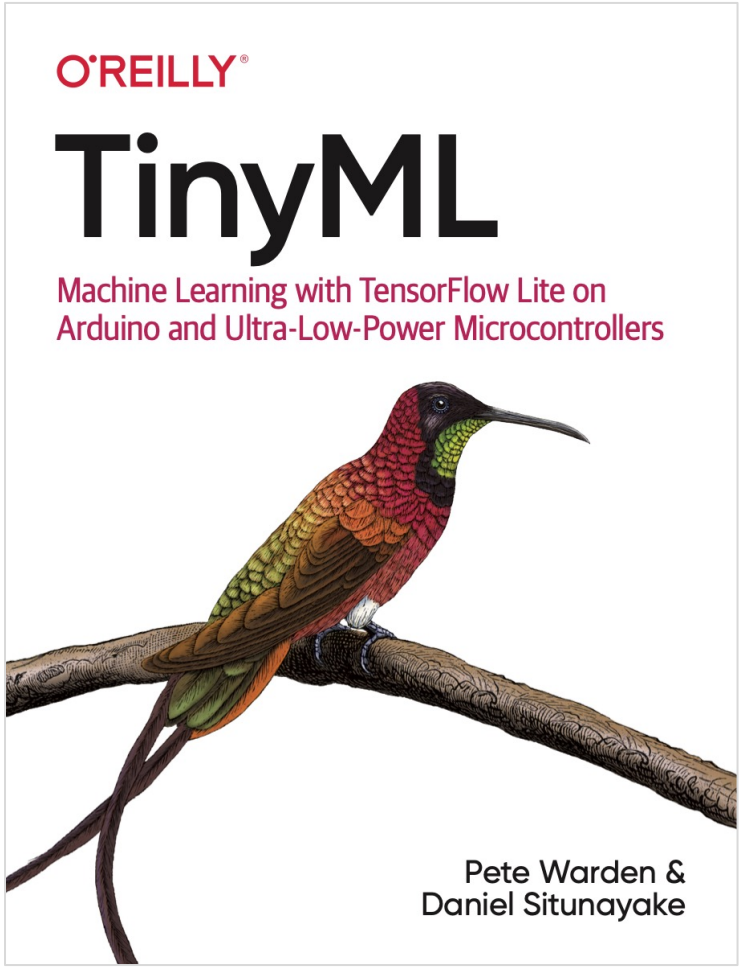
TinyML



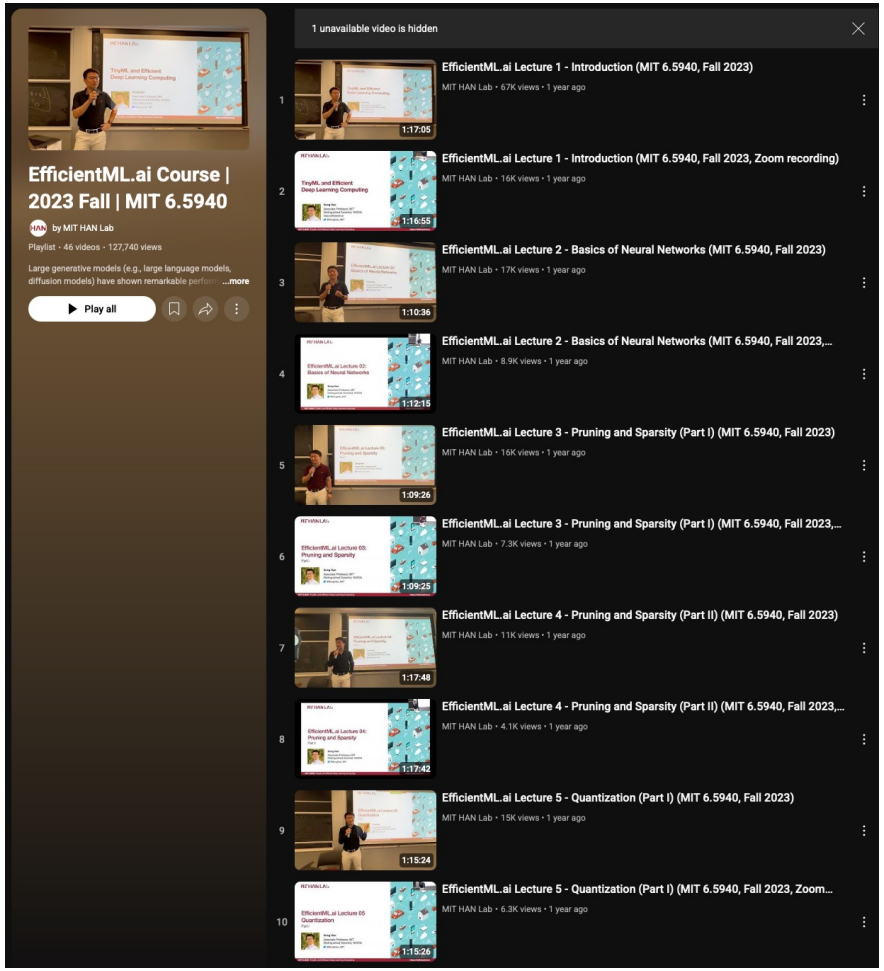
Gérald Rocher
Université Côte d'Azur, CNRS/INRIA Kairos
Mineure IOT-CPS 2025-2026

References



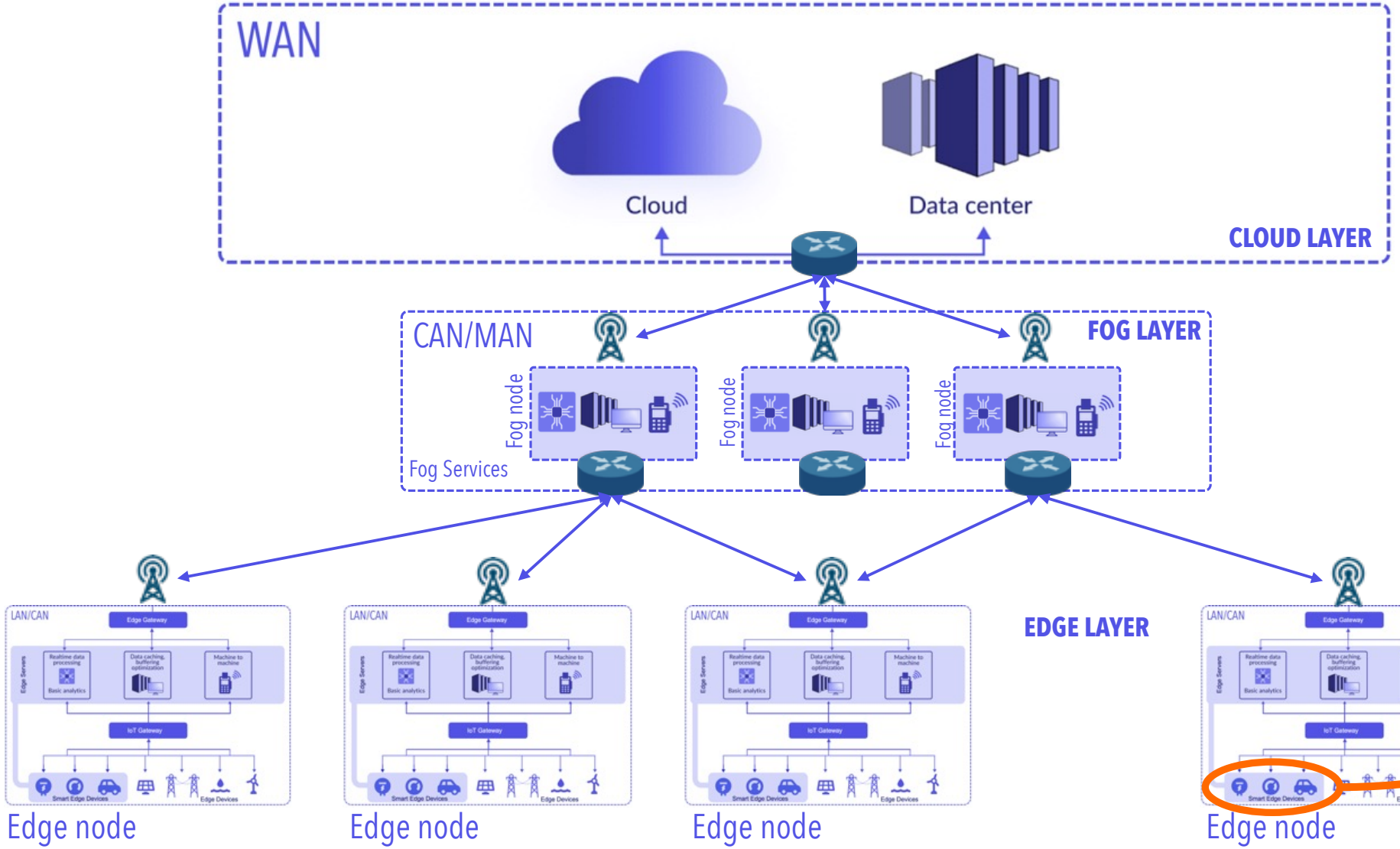
Link

Link



Link

What is TinyML?



Smart Device

- On-Device Sensor Analytics
  - Low Computational resources
  - Low Power Consumption
  - Battery Operated

It's all about squeezing AI into tiniest edge devices (Edge AI, AIoT)

https://tinyml.seas.harvard.edu/SciTinyML-22/assets/slides/TinyML_The_Future_of_Machine_Learning_is_Tiny_and_Bright.pdf

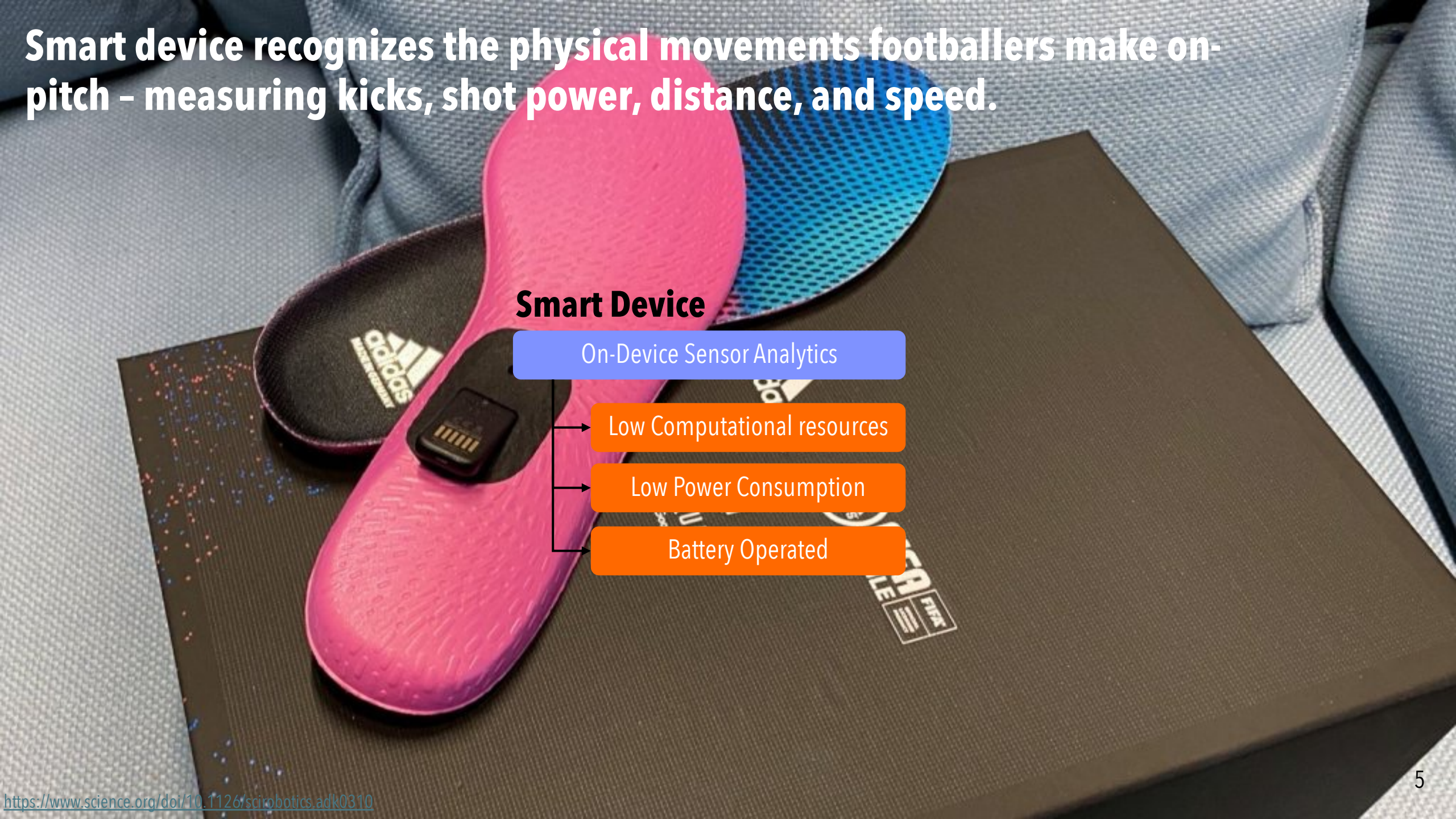
https://tinyml.seas.harvard.edu/SciTinyML-22/assets/slides/TinyML_The_Future_of_Machine_Learning_is_Tiny_and_Bright.pdf

Comprehensive Tracking for Your Health

Smart Device

- On-Device Sensor Analytics
  - Low Computational resources
  - Low Power Consumption
  - Battery Operated

Smart device recognizes the physical movements footballers make on-pitch – measuring kicks, shot power, distance, and speed.



https://www.science.org/doi/10.1126/scirobotics.adk0310

A 64mW DNN-based Visual Navigation Engine for Autonomous Nano-Drones

Daniele Palossi, Antonio Loquercio, Francesco Conti, *Member, IEEE,* Eric Flamand, Davide Scaramuzza, *Member, IEEE,* Luca Benini, *Fellow, IEEE*

***Abstract*—Fully-autonomous miniaturized robots (e.g., drones), with artificial intelligence (AI) based visual navigation capabilities, are extremely challenging drivers of Internet-of-Things edge intelligence capabilities. Visual navigation based on AI approaches, such as deep neural networks (DNNs) are becoming pervasive for standard-size drones, but are considered out of reach for nano-drones with a size of a few cm². In this work, we present the first (to the best of our knowledge) demonstration of a navigation engine for autonomous nano-drones capable of closed-loop end-to-end DNN-based visual navigation. To achieve this goal we developed a complete methodology for parallel execution of complex DNNs directly on board resource-constrained milliwatt-scale nodes. Our system is based on GAP8, a novel parallel ultra-low-power computing platform, and a 27 g commercial, open-source CrazyFlie 2.0 nano-quadrotor. As part of our general methodology, we discuss the software mapping techniques that enable the state-of-the-art deep convolutional neural network presented in [1] to be fully executed aboard within a strict 6 fps real-time constraint with no compromise in terms of flight results, while all processing is done with only 64 mW on average. Our navigation engine is flexible and can be used to span a wide performance range: at its peak performance corner, it achieves 18 fps while still consuming on average just 3.5% of the power envelope of the deployed nano-aircraft. To share our key findings with the embedded and robotics communities and foster further developments in autonomous nano-UAVs, we publicly release all our code, datasets, and trained networks.**

***Index Terms*—Autonomous UAV, Convolutional Neural Networks, Ultra-low-power, Nano-UAV, End-to-end Learning**

Supplementary Material

Supplementary video at: https://youtu.be/57Vy5cSvnaA. The project's code, datasets and trained models are available at: https://github.com/pulp-platform/pulp-dronet.

This work has been partially funded by projects EC H2020 OPRECOMP (732631) and ALOHA (780788), by the Swiss National Center of Competence Research (NCCR) Robotics and by the SNSF-ERC starting grant.

D. Palossi, F. Conti, E. Flamand and L. Benini are with the Integrated Systems Laboratory of ETH Zürich, ETZ, Gloriastrasse 35, 8092 Zürich, Switzerland (e-mail: dpalossi@iis.ee.ethz.ch, fconti@iis.ee.ethz.ch, eflamand@iis.ee.ethz.ch, lbenini@iis.ee.ethz.ch).

A. Loquercio and D. Scaramuzza are with the Robotic and Perception Group, at both the Dep. of Informatics (University of Zürich) and the Dep. of Neuroinformatics (University of Zürich and ETH Zürich), Andreasstrasse 15, 8050 Zürich, Switzerland.

F. Conti and L. Benini are also with the Department of Electrical, Electronic and Information Engineering of University of Bologna, Viale del Risorgimento 2, 40136 Bologna, Italy (e-mail: f.conti@unibo.it, luca.benini@unibo.it).

E. Flamand is also with GreenWaves Technologies, Pépinière Bergès, avenue des Papeteries, 38190 Villard-Bonnot, France (e-mail: eric.flamand@greenwaves-technologies.com).

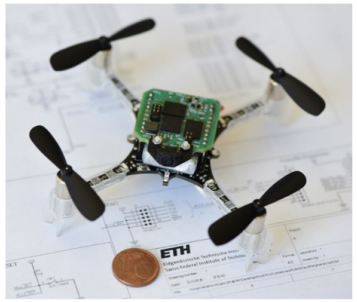


Fig. 1: Our prototype based on the COTS *Crazyflie 2.0* nano-quadrotor extended with our *PULP-Shield*. The system can run the *DroNet* [1] CNN for autonomous visual navigation up to 18 fps using only *onboard* resources.

I. Introduction

With the rise of the Internet-of-Things (IoT) era and rapid development of artificial intelligence (AI), embedded systems ad-hoc programmed to act in relative isolation are being progressively replaced by AI-based sensor nodes that acquire information, process and understand it, and use it to interact with the environment and with each other. The "ultimate" IoT node will be capable of autonomously navigating the environment and, at the same time, sensing, analyzing, and understanding it [2].

Fully autonomous nano-scale unmanned aerial vehicles (UAVs) are befitting embodiments for this class of smart sensors: with their speed and agility, they have the potential to quickly collect information from both their onboard sensors and from a plethora of devices deployed in the environment. Nano-UAVs could also perform advanced onboard analytics, to pre-select essential information before transmitting it to centralized servers [3]. The tiny form-factor of nano-drones is ideal both for indoor applications where they should safely operate near humans (for surveillance, monitoring, ambient awareness, interaction with smart environments, etc.) [4] and for highly-populated urban areas, where they can exploit

Smart Device

- On-Device Sensor Analytics
  - Low Computational resources
  - Low Power Consumption
  - Battery Operated

https://www.science.org/doi/10.1126/scirobotics.adk0310

TinyML Benefits

ML has the potential to affect various industries positively. Still, for ML to be more inclusive, the size and energy consumption can be an issue. TinyML benefits can help with multiple issues and make ML more inclusive

Reduced Delay

The overall dependence and latency decrease if the gadget has its own data processor, which is possible with TinyML.

Energy Efficiency

The entire stack of machine learning algorithms, including the hardware and software, can be changed to do this with TinyML.

Low Bandwidth

As mentioned above, since the data doesn't need to be frequently transferred, TinyML requires less internet bandwidth

Data Security

Compared to adding security measures to the entire network, securing an IoT device is significantly simpler and less expensive than adding security measures to the entire network.

Independence from Connectivity

TinyML allows smart edge devices to conclude even when there is no internet connection.

https://www.allerin.com/wp-blog/wp-content/uploads/2022/10/TinyML-Benefits.png

Top Players

Google

Harnesses its AI expertise and cloud infrastructure to create scalable machine learning models for edge devices, with **TensorFlow Lite for Microcontrollers** enabling real-time AI in wearables, IoT sensors, and mobile devices.

STMicroelectronics STM32 Cube.AI

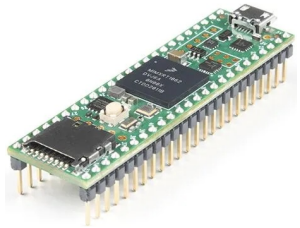
Their STM32 microcontrollers and STM32Cube.AI software tool streamline the process of integrating ML models into embedded systems.

ARM®

Provides ultra-energy-efficient processors like the **Cortex-M series and Ethos-U NPUs** to power TinyML solutions across industries, enabling high-performance AI on edge devices without draining battery life.

… and many other actors MICROCHIP NXP EDGE IMPULSE SYNTIANT® Making Edge AI a Reality™ Microsoft

“Tiny” Hardware



Teensy 4.1

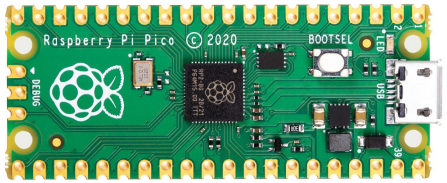
ARM Cortex-M7 600MHz,
7936K Flash, 1024K RAM, 4K EEPROM,
Float point math unit, 64 & 32 bits,

USB device 480 Mbit/sec & USB host 480 Mbit/sec,
55 digital input/output pins, 35 PWM output pins,
18 analog input pins
8 serial, 3 SPI, 3 I2C ports
2 I2S/TDM and 1 S/PDIF digital audio port
3 CAN Bus (1 with CAN FD)
1 SDIO (4 bit) native SD Card port
Ethernet 10/100 Mbit with DP83825 PHY
32 general purpose DMA channels
Cryptographic Acceleration & RNG
…

Sipeed Maixduino

Dual-core 64bit RISC-V 400MHz,
8MiB 64bit on-chip SRAM,
16MiB Flash (max 128GB),

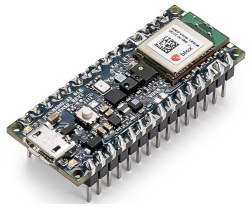
Built-in neural network accelerator.



Raspberry Pi Pico W

Dual-core Arm Cortex-M0 133MHz
264kB on-chip SRAM, 2MB on-board QSPI flash

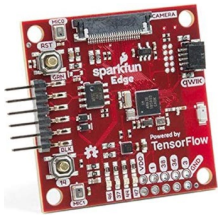
2.4GHz 802.11n wireless LAN
26 multifunction GPIO pins, including 3 analog inputs
2 UART,
2 SPI controllers,
2 I2C controllers,
16 PWM channels
1 USB 1.1
8 Programmable I/O (PIO)
Accelerated integer and floating-point libraries on-chip



Arduino Nano 33 BLE Sense

32-bit ARM® Cortex®-M4 CPU 64MHz ,
1MB memory,

9 axis inertial sensor,
Humidity, and temperature sensor,
Barometric sensor,
Microphone,
Gesture, proximity, light color and light intensity sensor.



SparkFun Edge

32-bit ARM Cortex-M4F processor with DMA,
48MHz CPU clock, 96MHz with TurboSPOT™,
1MB Flash, 384KB SRAM,

Dedicated Bluetooth processor with BLE 5,
ST LIS2DH12 3-axis accelerometer
2x MEMS microphones with operational amplifier
Himax HM01B0 camera connector

Seeed Studio XIAO ESP32S3 Sense

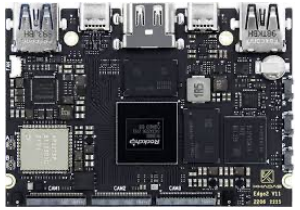
48MHz SAMD21G18 chip,
32KB of SRAM, and 256KB of flash memory

11x analog / 11x digital Pins,
1x I2C interface,
1x UART port,
1 SPI port

ML Applications	TinyML on Microcontrollers	Edge ML on SBCs / Servers
Forecasting & Anomaly Detection	Yes, with limited model size	Yes
Image Recognition	Yes, with lower-resolution and speed	Yes, real time processing with high-end hardware
Audio Recognition	Keyword / wake word detection only	Full natural language processing (NLP)

Raspberry Pi 5

64-bit quad-core Arm Cortex-A76 2.4GHz
800MHz VideoCore VII GPU,
LPDDR4X-4267 SDRAM (2GB, 4GB, 8GB and 16GB),
Dual-band 802.11ac Wi-Fi®,
Bluetooth 5.0 / Bluetooth Low Energy (BLE),
Raspberry Pi standard 40-pin header



Khadas Edge 2

Quad Core ARM Cortex-A76 2.25GHz + Quad Core Cortex-A55 CPU 1.8GHz ,
ARM Mali-G610 MP4 GPU up to 1GHz
8GB/16GB LPDDR4X SDRAM,

Build-in 6 TOPS Performance NPU

Jetson Nano ORIN

6 Cores ARM Cortex-A78AE + 512-core Nvidia Ampere GPU + 16 Tensor Cores ,

4GB/8GB LPDDR5 SDRAM,

What makes TinyML Challenging?

🔥**Resource Limits**

- **Devices run on tiny batteries, with kilobytes of RAM and low CPU power**, causing issues like lost training progress and degraded accuracy.

🔥 **Heterogeneity**

- Different chip architectures (ARM, RISC-V), frameworks, and sensor data formats make model portability and maintenance hard.

🔥 **Network Strain**

- While TinyML reduces dependence on constant connectivity, some applications still need timely updates or coordination, making intermittent or low-bandwidth links a challenge.

🔥 **Security & Privacy**

- Although TinyML reduces some security and privacy risks, it doesn't eliminate all risks like fake devices pretending to be legitimate, difficulty verifying many small devices, and potential leaks of sensitive sensor or location data.

🔥 **Model Design**

- **Needs aggressive pruning, quantization, or model splitting** to fit constraints without killing accuracy.

https://www.unitec.ac.nz/epress/wp-content/uploads/2024/07/20-CITRENZ2023-Proceedings-Adlakha-Kabbar.pdf

What makes TinyML Challenging?

Resource Limits

- **Devices run on tiny batteries, with kilobytes of RAM and low CPU power**, causing issues like lost training progress and degraded accuracy.

Heterogeneity

- Different chip architectures (ARM, RISC-V), frameworks, and sensor data formats make model portab

There are currently many efforts by chip vendors, software companies, and open-source developers to seamlessly enable ML implementation on edge devices.

Netwo

- While timely

Security & Privacy

- Although TinyML reduces some security and privacy risks, it doesn't eliminate all risks like fake devices pretending to be legitimate, difficulty verifying many small devices, and potential leaks of sensitive sensor or location data.

Model Design

- **Needs aggressive pruning, quantization, or model splitting** to fit constraints without killing accuracy.

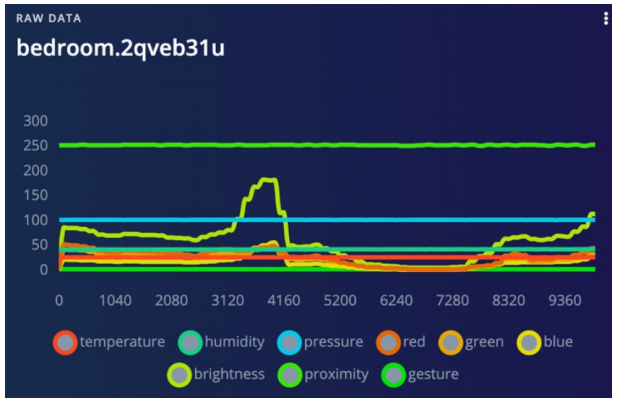
https://www.unitec.ac.nz/epress/wp-content/uploads/2024/07/20-CITRENZ2023-Proceedings-Adlakha-Kabbar.pdf

No/Low-code Platforms examples

EDGE IMPULSE

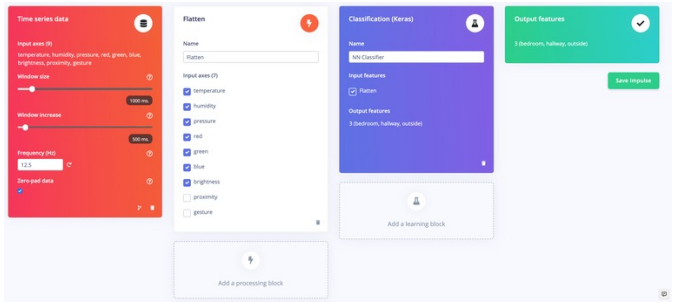
EDGE IMPULSE Platform → AI for MANY Edge Device (MCUs, NPUs, CPUs, GPUs)

1. Build a dataset



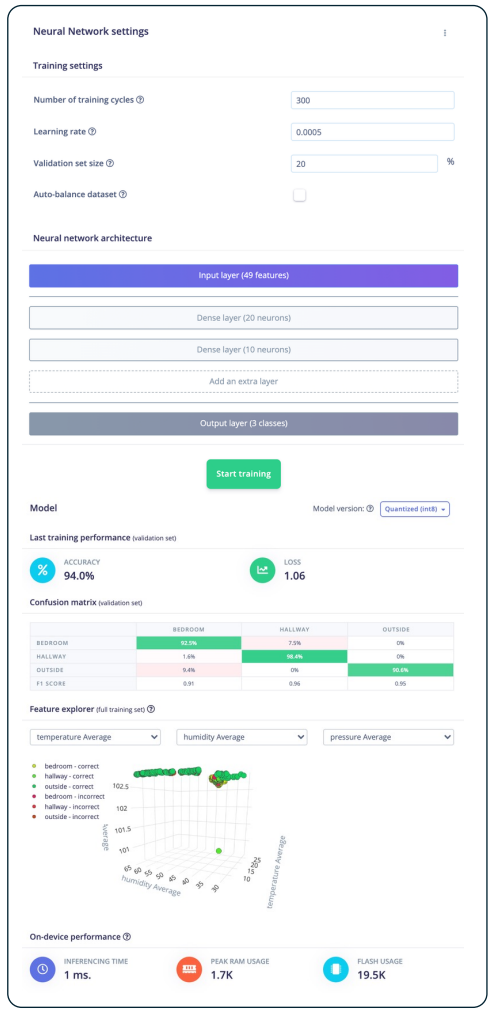
Click Start sampling and slowly move the board around while data is collected

2. Design an 'Impulse'

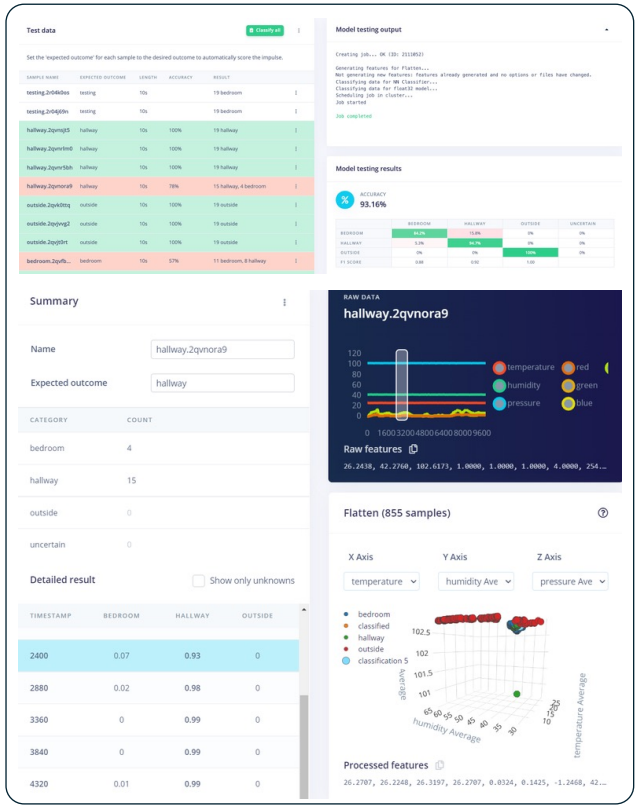


A combination of preprocessing blocks followed by machine learning blocks.

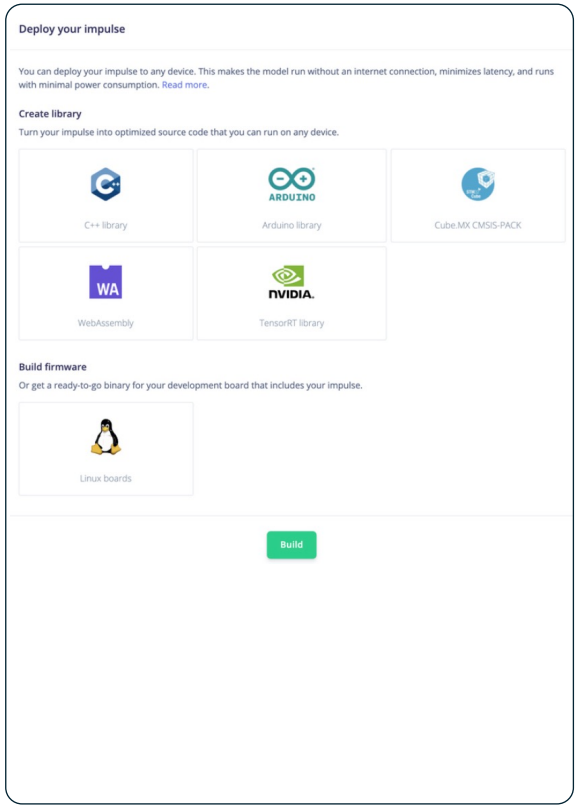
3. Train the model



4. Test the model



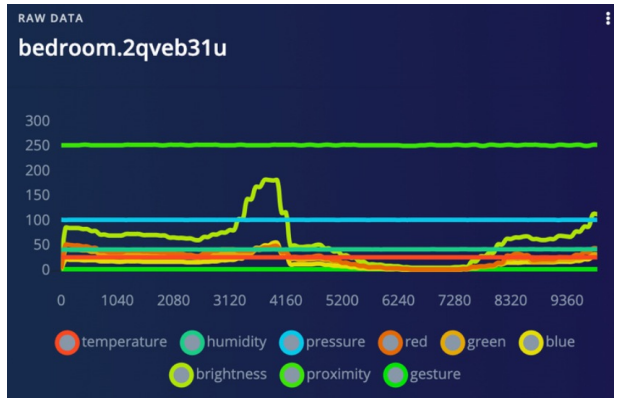
5. Deploy the model



No/Low-code Platforms examples

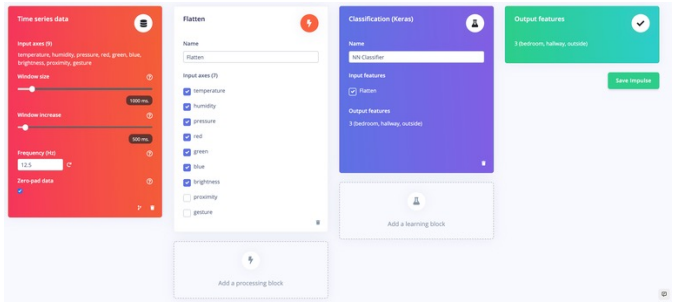
EDGE IMPULSE Platform → AI for MANY Edge Device (MCUs, NPUs, CPUs, GPUs)

1. Build a dataset



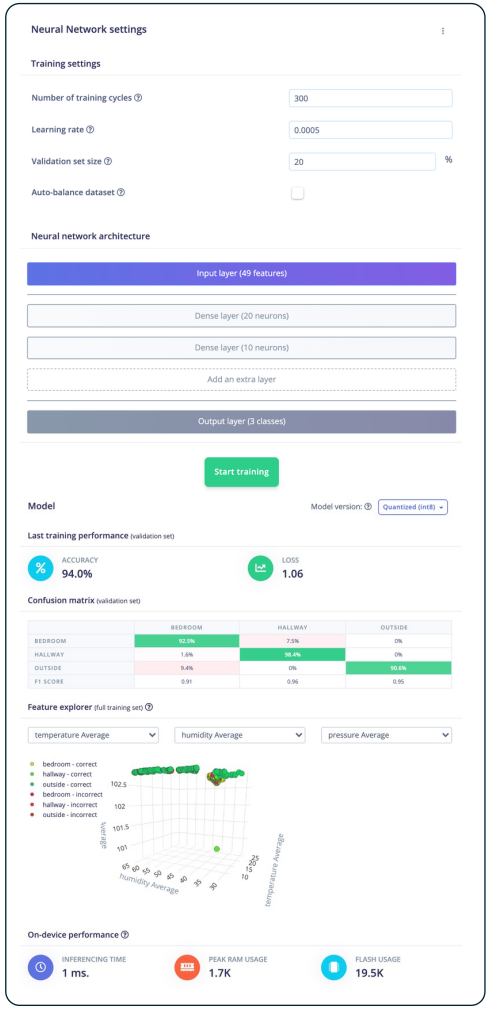
Click Start sampling and slowly move the board around while data is collected

2. Design an 'Impulse'

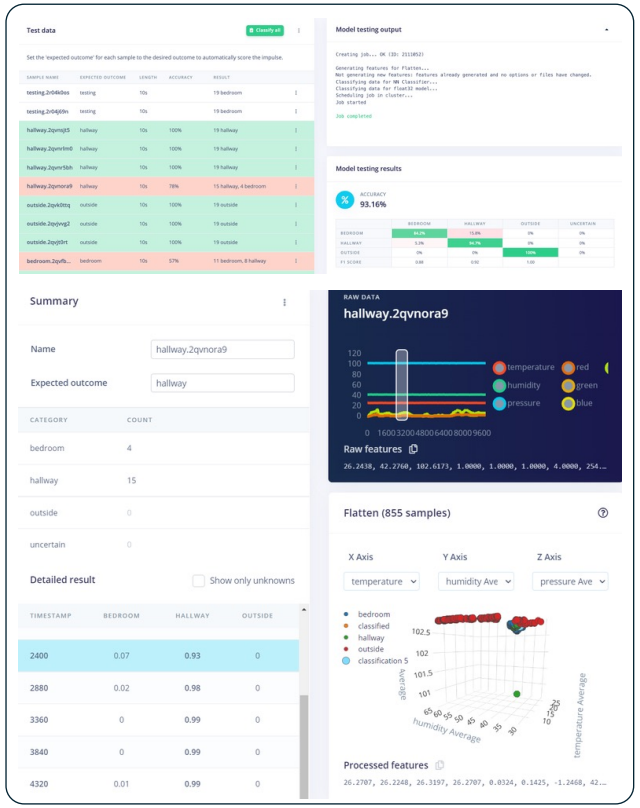


A combination of preprocessing blocks followed by machine learning blocks.

3. Train the model



4. Test the model

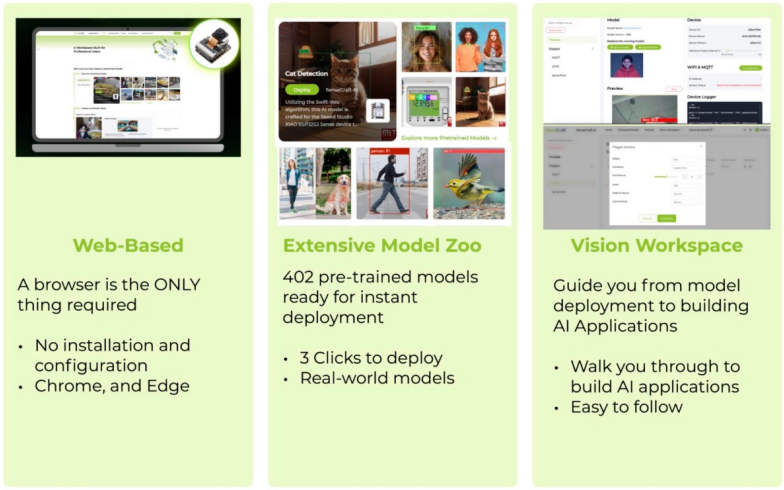


5. Deploy the model

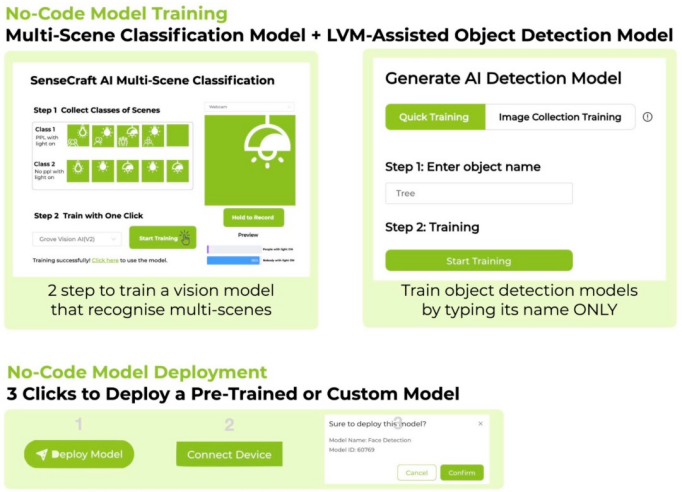
- Arduino Nicla Vision M4 (Cortex-M4 240MHz)
- Arduino Portenta H7 (Cortex-M7 480MHz)
- Arduino UNO Q (Qualcomm QRB2210)
- BrainChip AKD1000 or AKD1500
- BrickML (Cortex-M33 192MHz)
- Cortex-M4F 80MHz
- Cortex-M7 216MHz
- Digi ConnectCore 93 (NXP i.MX 93 - Cortex-A55 1.7GHz + Ethos-U65-256)
- Digi ConnectCore 93 (NXP i.MX 93 - Cortex-A55 1.7GHz)
- Espressif ESP-EYE (ESP32 240MHz)
- Himax WE-I (ARC DSP 400MHz)
- Himax WiseEye2 (Cortex-M55 400 MHz)
- Himax WiseEye2 (Cortex-M55 400 MHz) + Ethos-U55-64
- IMDT V2H (CPU)
- IMDT V2H (with Renesas' RZ/V2H)
- Infineon PSoC6 CY8C624 (Cortex-M4F 150 MHz)
- Infineon PSoC6 CY8C6347 (Cortex-M4F 150 MHz)
- MacBook Pro 16" 2020 (Intel Core i9 2.4GHz)
- MacBook Pro 16" 2021 (Apple M1 Pro)
- MemryX MX3
- Microchip SAMA7D65 Evaluation Kit
- Microchip SAMA7G54 Evaluation Kit
- Nordic nRF52840 DK (Cortex-M4F 64MHz)
- Nordic nRF5340 DK (Cortex-M33 128MHz)
- Nordic nRF54L15 DK (Cortex-M33 128MHz)
- Nordic nRF9151 DK (Cortex-M33 64MHz)
- Nordic nRF9160 DK (Cortex-M33 64MHz)
- Nordic nRF9161 DK (Cortex-M33 64MHz)
- Nvidia Jetson Nano
- Nvidia Jetson Orin NX
- Nvidia Jetson Orin Nano
- OpenMV Cam H7 Plus
- Particle Boron
- Particle Photon 2
- Qualcomm Dragonwing RB3 Gen 2 Development Kit
- Raspberry Pi 4
- ✓ Raspberry Pi 5
- Raspberry Pi RP2040 (Cortex-M0+ 133MHz)
- Raspberry Pi RP2350 (Cortex-M33 150MHz)
- Renesas RA6M5 (Cortex-M33 200MHz)
- Renesas RA8D1 (Cortex-M85 480MHz)
- Renesas RZ/G2L
- Renesas RZ/V2H (CPU)
- Renesas RZ/V2H (with DRP-AI3 accelerator)
- Renesas RZ/V2L (CPU)
- Renesas RZ/V2L (with DRP-AI accelerator)
- ST IoT Discovery Kit (Cortex-M4F 80MHz)
- ST STM32N6 (Cortex-M55 800MHz + ST Neural-ART accelerator)
- Seeed SenseCAP A1101 (HX6537-A ARC DSP 400MHz)
- Seeed Studio Wio Terminal (Cortex-M4F 120MHz)
- Seeed Vision AI Module (HX6537-A ARC DSP 400MHz)
- SiLabs EFR32MG24 (Cortex-M33 78MHz)
- SiLabs Thunderboard Sense 2 (Cortex-M4F 40MHz)
- Sony Spresense (Cortex-M4F 156MHz)
- Synaptics KA10000
- TI AM62A (with Deep Learning Accelerator)
- TI AM68A (with Deep Learning Accelerator)
- TI LAUNCHXL-CC1352P (Cortex-M4F 48MHz)
- TI TDA4VM (with matrix multiply accelerator (MMA))
- Thundercomm Rubik Pi 3

No/Low-code Platforms examples

SenseCraft AI

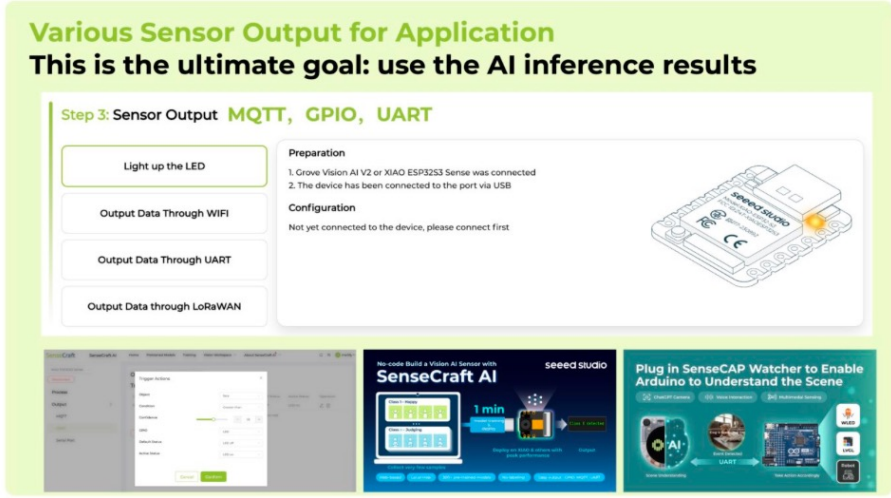


Web-based.



Collect and Classify Scenes

- Hold to capture around 10 photos for each class, and then group your examples into distinct classes (like "room occupied with lights on" vs. "empty room with lights off"),
- Train with One Click: Hit train and instantly preview your model's accuracy.



Application Focused

SenseCraft AI supports multiple output methods (MQTT, GPIO, UART, I2C, SPI) to stream your AI inference results with any microcontrollers or systems.

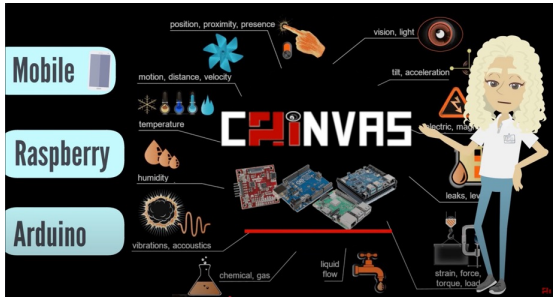
- Supports XIAO ESP32S3 Sense, Grove Vision AI Module V2, SenseCAP Watcher, reCamera, SenseCAP A1102 LoRaWAN Vision AI Sensor, and reComputer Jetson Edge Computing Devices.

No/Low-code Platforms examples

cAInvas Platform

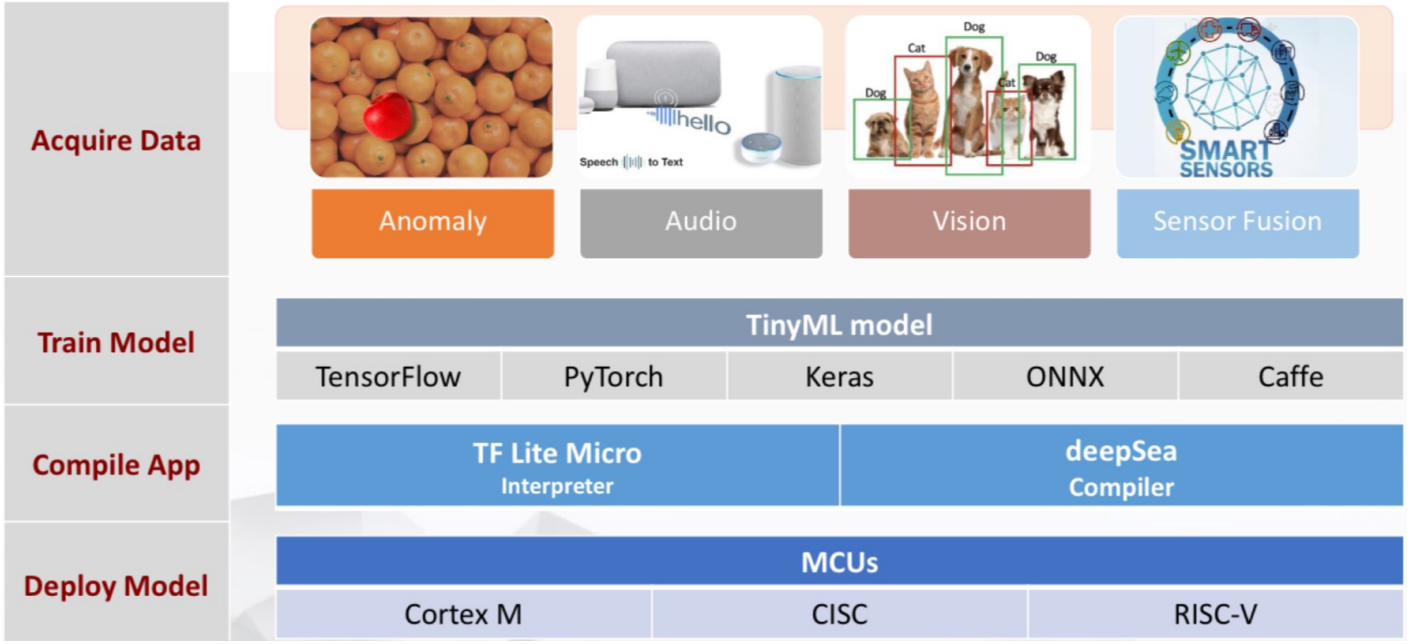
- Building Machine Learning Models For Edge Devices
  - Supported plateforms : ARM Cortex M3, M4, M55, M7, A series, X86, Xtensa (ESP32),
  - Supported OS : Bare Metal, FreeRTOS, Linux derivative.
  - Output : C++, static/dynamic lib.

- www.tinyML.studio
- https://www.ai-tech.systems/cainvas/



1. **Create dataset straight from the device,**
2. **Use toolset including deepsea edge compiler to build a high-performance app to run on boards**
3. **Download and deploy app and monitor with or without cloud.**

https://www.youtube.com/watch?v=JM8BQlm9QCE&t=84s

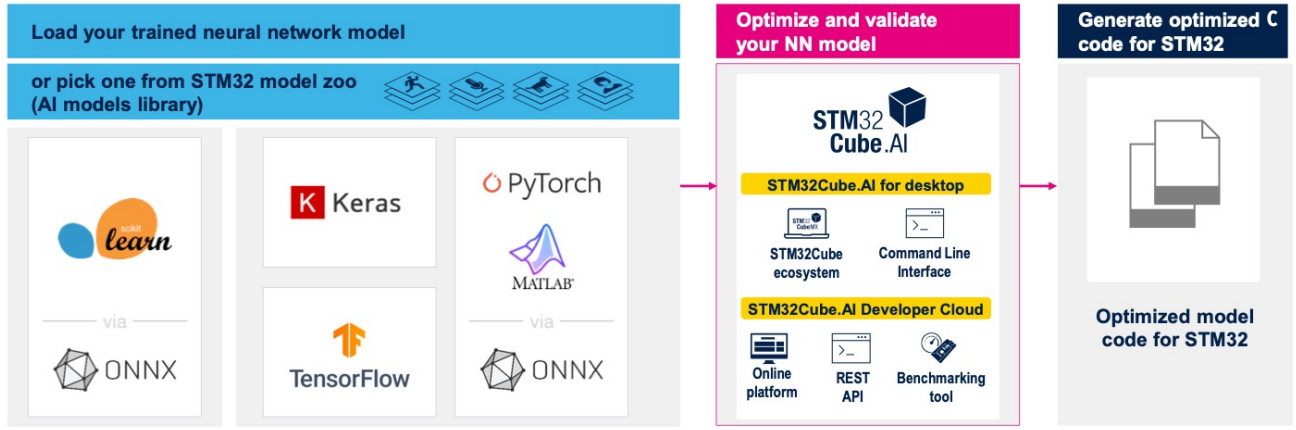


https://www.arm.com/partners/catalog/ai-technology-and-systems

Model deployment toolchain for STM32

STM32Cube.AI

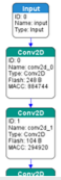
- Allows the optimization and deployment of trained Neural Network models from the most popular AI frameworks on any STM32 microcontroller, including the Neural-ART Accelerator NPU (STM32N6).



source

Graph optimizer

Automatically improve performance through graph simplifications & optimizations that benefit STM32 target HW architectures



- Auto graph rewrite
- Node/operator fusion
- Layout optimization
- Constant-folding…
- Operator-level info to fine-tune memory footprint and computation

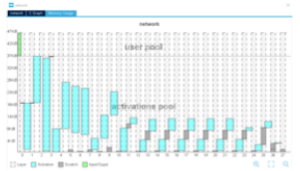
Quantized model support

Import your quantized ANN to be compatible with STM32 embedded architectures while keeping their performance

- From FP32 to Int8 or mixed-precision
- Minimum loss of accuracy
- Code validation on target
  - Latency
  - Accuracy
  - Memory footprint

Memory optimizer

Optimize memory allocation to get the best performance while respecting the constraints of your embedded design



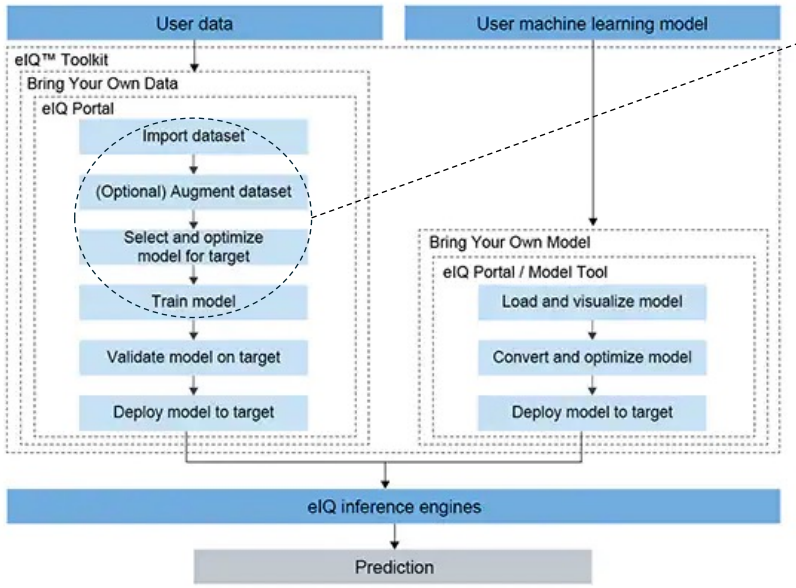
- Memory allocation
- Internal/external memory repartition
- Model-only update option

Model deployment toolchain for NXP

eIQ® AI Software Development Environment

- Enables the use of ML algorithms on NXP EdgeVerse™ microcontrollers and microprocessors, including i.MX RT crossover MCUs, and i.MX family application processors.

eIQ Toolkit

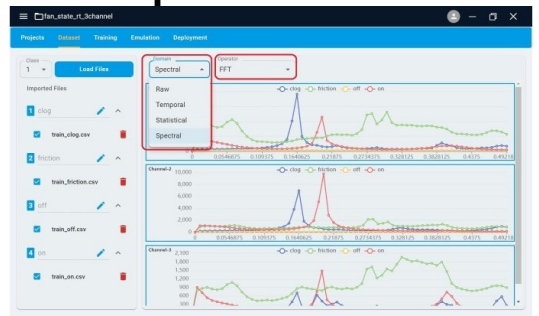


https://www.nxp.com/docs/en/user-guide/EIQTKUG-1.6.9.pdf

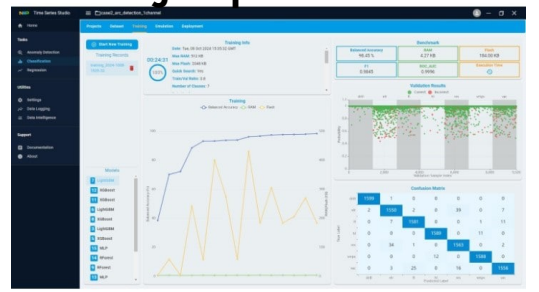
eIQ Time Series Studio

- Features an automated machine-learning workflow that streamlines the development and deployment of time series-based machine learning models across microcontrollers.
- Supports a wide range of sensor input signals, including voltage, current, temperature, vibration, pressure, sound, time of flight, among others, as well as combinations of these for multimodal sensor fusion.

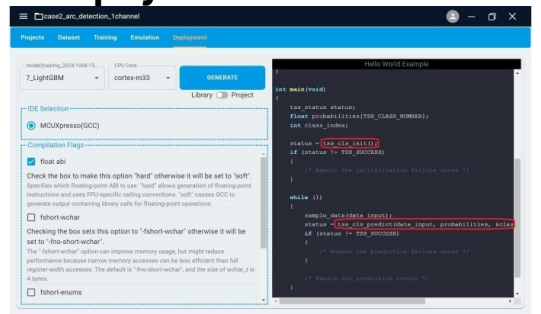
1. Data Input



2. Training & Optimizations



3. Deployment



https://proceedings.mlsys.org/paper_files/paper/2021/file/c4d41d9619462c534b7b61d1f772385e-Paper.pdf

Portable Inference Runtime Library


MicroNets: Neural Network Architectures for Deploying TinyML Applications on Commodity Microcontrollers

Colby Banbury*[1,2] Chuteng Zhou*[1] Igor Fedorov*[1] Ramon Matas Navarro[1] Urmish Thakker[3] Dibakar Gope[1] Vijay Janapa Reddi[2] Matthew Mattina[1] Paul N. Whatmough[1]

Abstract

Executing machine learning workloads locally on resource constrained microcontrollers (MCUs) promises to drastically expand the application space of IoT. However, so-called TinyML presents severe technical challenges, as deep neural network inference demands a large compute and memory budget. To address this challenge, neural architecture search (NAS) promises to help design accurate ML models that meet the tight MCU memory, latency, and energy constraints. A key component of NAS algorithms is their latency/energy model, i.e., the mapping from a given neural network architecture to its inference latency/energy on an MCU. In this paper, we observe an intriguing property of NAS search spaces for MCU model design: on average, model latency varies linearly with model operation (op) count under a uniform prior over models in the search space. Exploiting this insight, we employ differentiable NAS (DNAS) to search for models with low memory usage and low op count, where op count is treated as a viable proxy to latency. Experimental results validate our methodology, yielding our MicroNet models, which we deploy on MCUs using Tensorflow Lite Micro, a standard open-source neural network (NN) inference runtime widely used in the TinyML community. MicroNets demonstrate state-of-the-art results for all three TinyMLperf industry-standard benchmark tasks: visual wake words, audio keyword spotting, and anomaly detection. Models and training scripts can be found at https://github.com/ARM-software/ML-zoo.

1 Introduction

Machine learning (ML) methods play an increasingly central role in a myriad of internet-of-things (IoT) applications. Using ML, we can interpret the wealth of sensor data that IoT devices generate. Prototypical uses in IoT include tasks such as monitoring environmental conditions such as temperature and atmosphere (e.g., carbon monoxide levels), monitoring mechanical vibrations from machinery to predict failure, or visual tasks such as detecting people or animals. User interfaces based on speech recognition and synthesis are also very common, as many IoT devices have limited user input features and small displays. In mobile applications, ML inference is often offloaded to the cloud, where compute resources are more abundant. However, offloading introduces overheads in terms of latency, energy and privacy. It also requires access to communications, such as WiFi or cellular access. For the proliferating class of IoT devices, offloading can be prohibitively expensive, in terms of both the radio chips which increase the bill of materials, as well as the network access costs.

*Equal contribution [1]Arm ML Research [2]Harvard University [3]SambaNova Systems. Correspondence to: Colby Banbury <cbanbury@g.harvard.edu>.

Proceedings of the 4th MLSys Conference, San Jose, CA, USA, 2021. Copyright 2021 by the author(s).

Table 1. Illustrative comparison of hardware for CloudML, MobileML and TinyML, including the MCUs targeted in this work.

Platform	Architecture	Memory	Storage	Power	Price
CloudML Nvidia V100	GPU Nvidia Volta	HBM 16GB	SSD/Disk TB~PB	250W	$9K
MobileML Cell Phone	CPU Mobile CPU	DRAM 4GB	Flash 64GB	~8W	~$750
TinyML	MCU	SRAM	eFlash		
F446RE	Arm M4	128KB	0.5MB	0.1W	$3
F746ZG	Arm M7	320KB	1MB	0.3W	$5
F767ZI	Arm M7	512KB	2MB	0.3W	$8

TinyML is an alternative paradigm, where we execute ML tasks locally on IoT devices. This allows for real time analysis and interpretation of data at the point of collection, which translates to huge advantages in terms of cost and privacy. Microcontroller units (MCUs) are the ideal hardware platform for TinyML, as they are typically small (~1cm^3), cheap (~$1) and low-power (~1mW) compared to mobile and cloud platforms (Table 1). MCUs typically integrate a CPU, digital and analog peripherals, on-chip embedded flash (eFlash) memory for program storage and Static Random-Access Memory (SRAM) for intermediate data. However, deploying deep neural networks on MCUs is extremely challenging; the most severe limitation being the small and flat memory system (Figure 1) within which the model weights and activations must be stored. Therefore, to

TensorFlow Lite for μControllers (TFLM)

- https://www.tensorflow.org/lite/microcontrollers?hl=fr

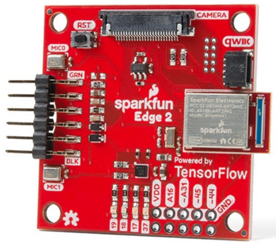
- **Supported μControllers (ARM Cortex-M Series)**
  - Arduino Nano 33 BLE Sense,
  - SparkFun Edge,
  - STM32F746 Discovery kit,
  - Adafruit EdgeBadge,
  - Kit Adafruit TensorFlow Lite for Microcontrollers,
  - Adafruit Circuit Playground Bluefruit,
  - Espressif ESP32-DevKitC,
  - Espressif ESP-EYE,
  - Wio Terminal : ATSAMD51,
  - Himax WE-I Plus EVB Endpoint AI,
  - Synopsys DesignWare ARC EM,
  - Sony Spresense.

✓ What it does

- Interprets models (like standard TFLite) but with no system dependencies (no malloc, no external libraries),
- **Generates C++ code for inference**, however, it does not optimize it for a specific platform (unlike STM32Cube.AI or NXP eIQ).
- Uses pre-optimized kernels for common architectures (CMSIS-NN for ARM Cortex-M, ESP-NN for ESP32).

✗ What it does not do

- No "bare-metal" code generation (unlike STM32Cube.AI),
- Manual integration of the TFLite Micro runtime into your project,
- No advanced hardware optimization (quantization, pruning, etc.) at compile time, these steps must be done beforehand (via TensorFlow or tools like Edge Impulse).



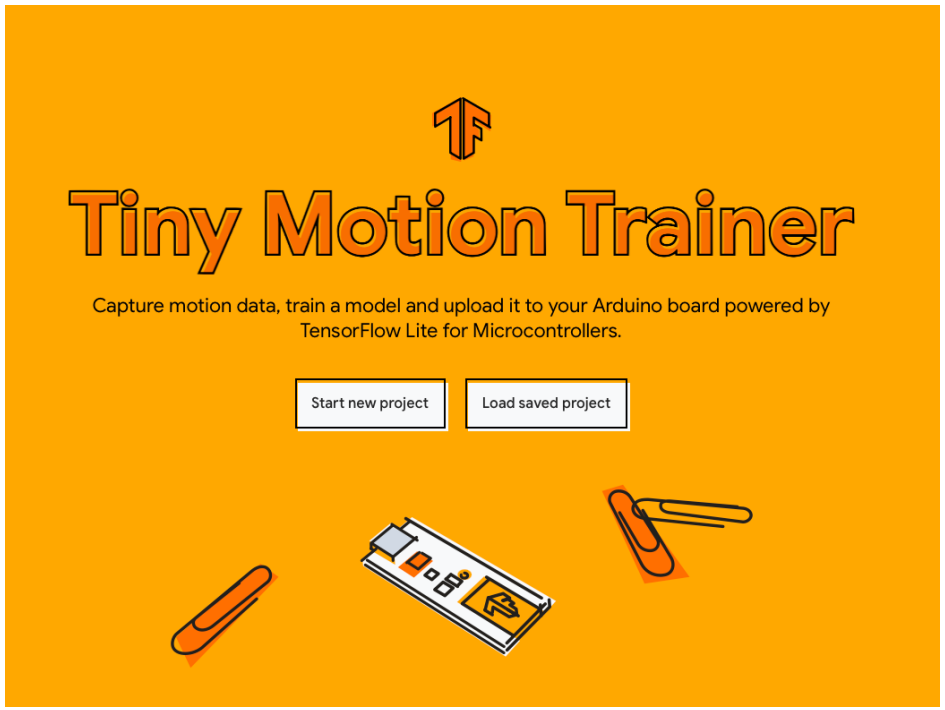
https://github.com/tensorflow/tflite-micro/tree/main/tensorflow/lite/micro/examples

Portable Inference Runtime Library

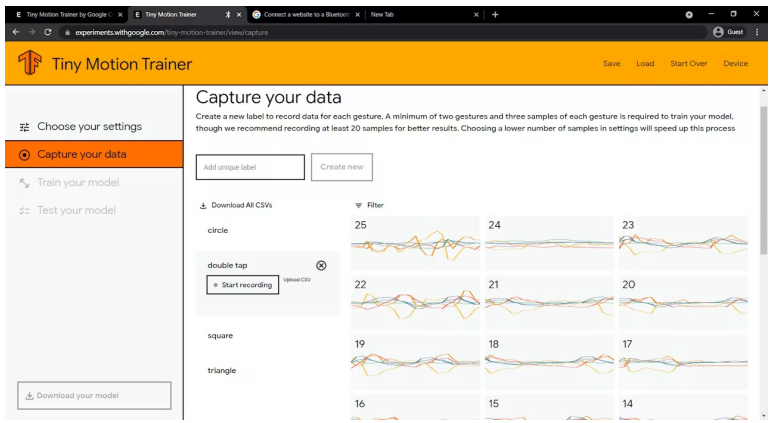
TensorFlow Lite for μControllers (TFLM)

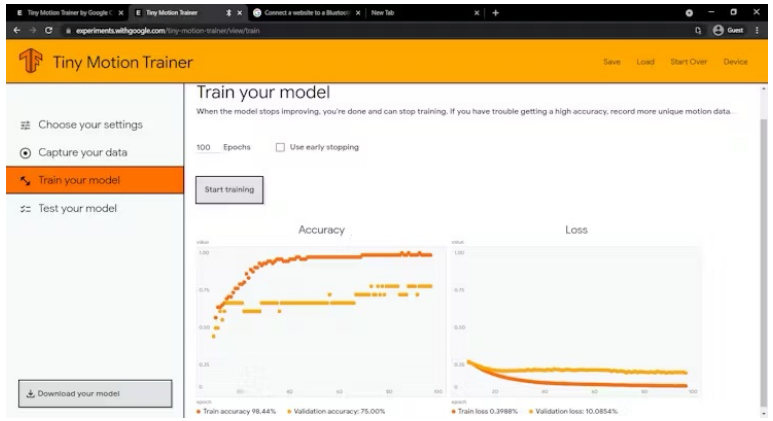
- https://www.tensorflow.org/lite/microcontrollers?hl=fr

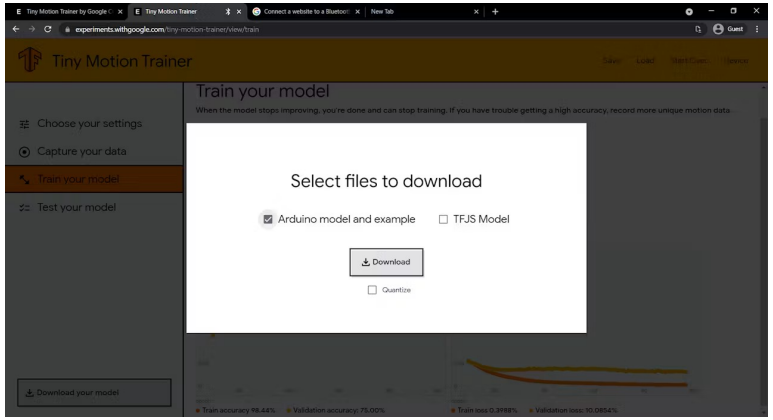
https://experiments.withgoogle.com/tiny-motion-trainer/view/ https://www.youtube.com/watch?v=jqVCR2MUJEs

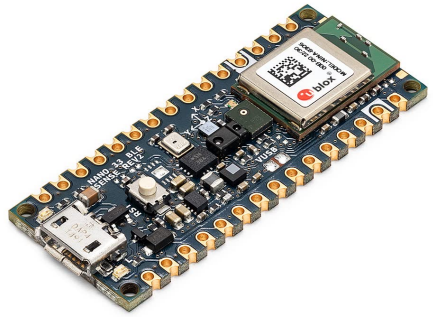


https://github.com/googlecreativelab/tiny-motion-trainer









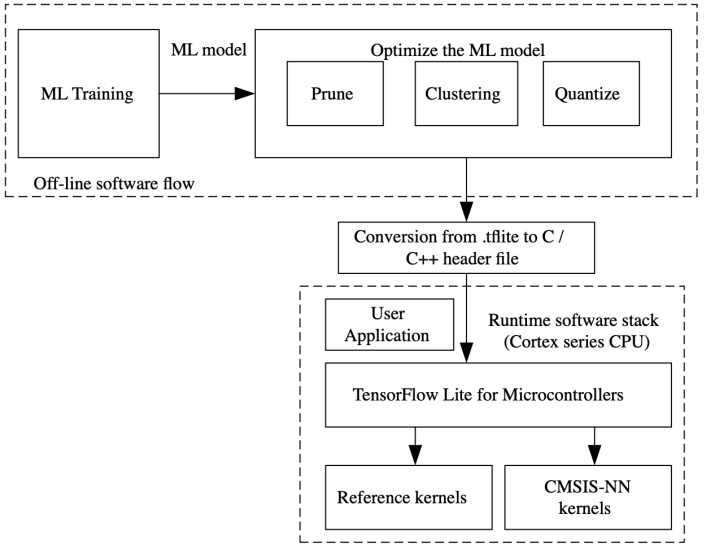
Arduino Nano 33 BLE Sense

Portable Inference Runtime Library

TensorFlow Lite for μControllers (TFLM)

- https://www.tensorflow.org/lite/microcontrollers?hl=fr
- TFLM supports a number of TensorFlow AI operators (around 100 ops), but not all of them yet (see https://medium.com/@jahiz.ahmed/layers-current-supported-on-tensorflow-lite-for-micro-1d79c31e8088).

- **Software Flow on ARM Cortex-M processors and Ethos-U NPU**



- **Software Flow on X-CUBE-AI (STM32)**