

AI on MCU

Guang Feng Wang
FT RPD FOA ART-CN2

Agenda

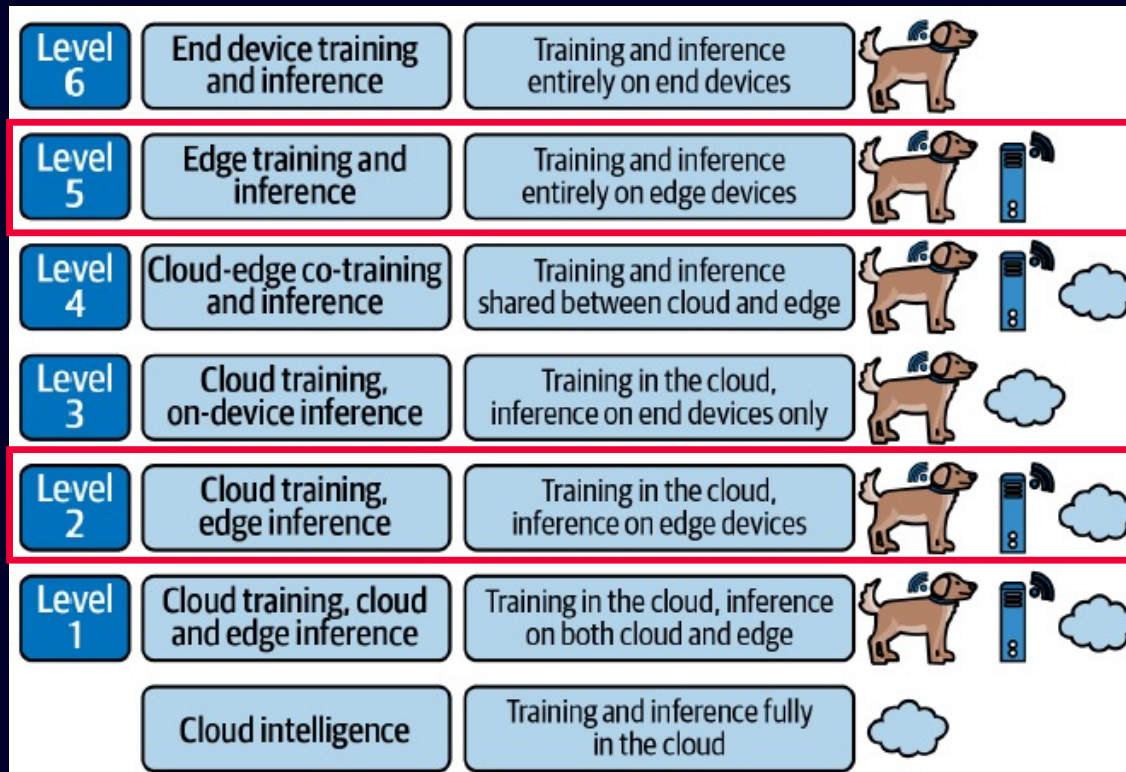
- 01 Edge AI
- 02 Challenges of AI/ML on MCUs
- 03 MCU(Microcontroller Unit) AI/ML
- 04 Two use cases for MCU based AI/ML

01 Edge AI

What is Edge AI(Artificial Intelligence)

Edge Artificial Intelligence

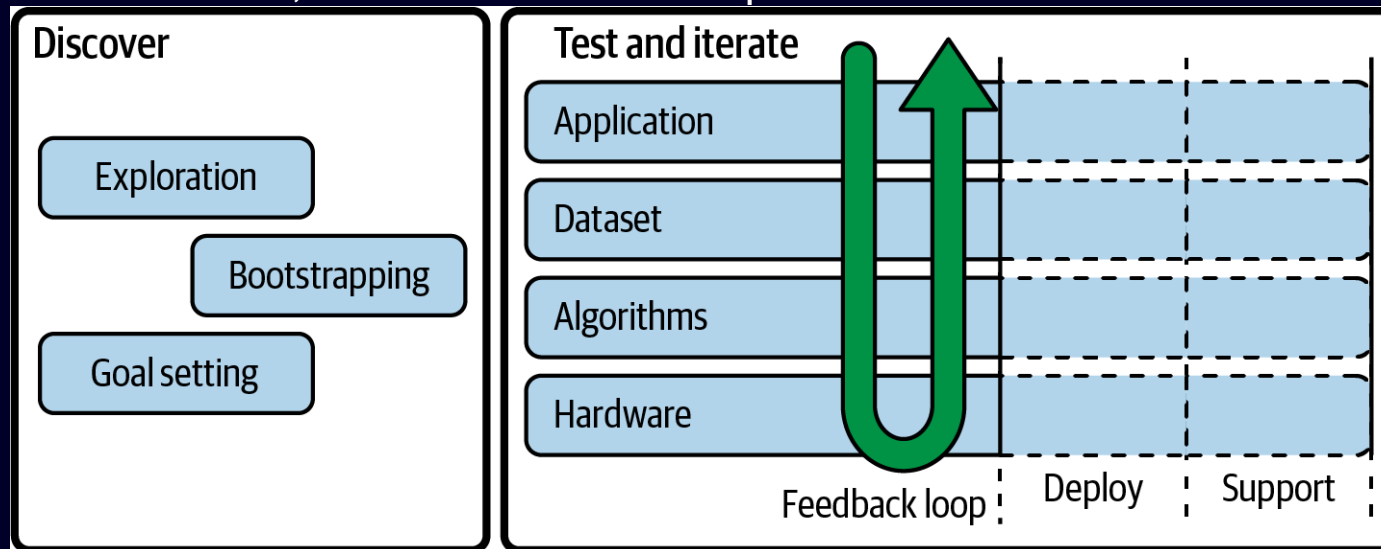
- Edge AI is the combination of edge devices and artificial intelligence.



Building on the benefits of IoT and edge computing to collect and process data locally, on-device ML and DL processing reduces latency, increases data privacy and security, and reduces the need for a continuous cloud connection by providing edge connectivity and analytics solutions. ML and DL edge intelligent processing open opportunities for new, robust, scalable AI systems across the edge continuum (micro- deep-, and meta-edge) and multiple industries.

The Edge AI workflow

Like any sophisticated engineering project, a typical edge AI project involves multiple tracks of work, some of which run in parallel.



- Exploration: A machine learning system should explore more new action strategies or make more use of existing effective action strategies.
- Bootstrapping: In the absence of explicit evaluation functions or large amounts of data, initial feedback can be used to guide the system's actions.
- Goal Setting: Determine long-term and short-term goals and how success will be measured.

AI/ML and MCUs: A Brief Overview

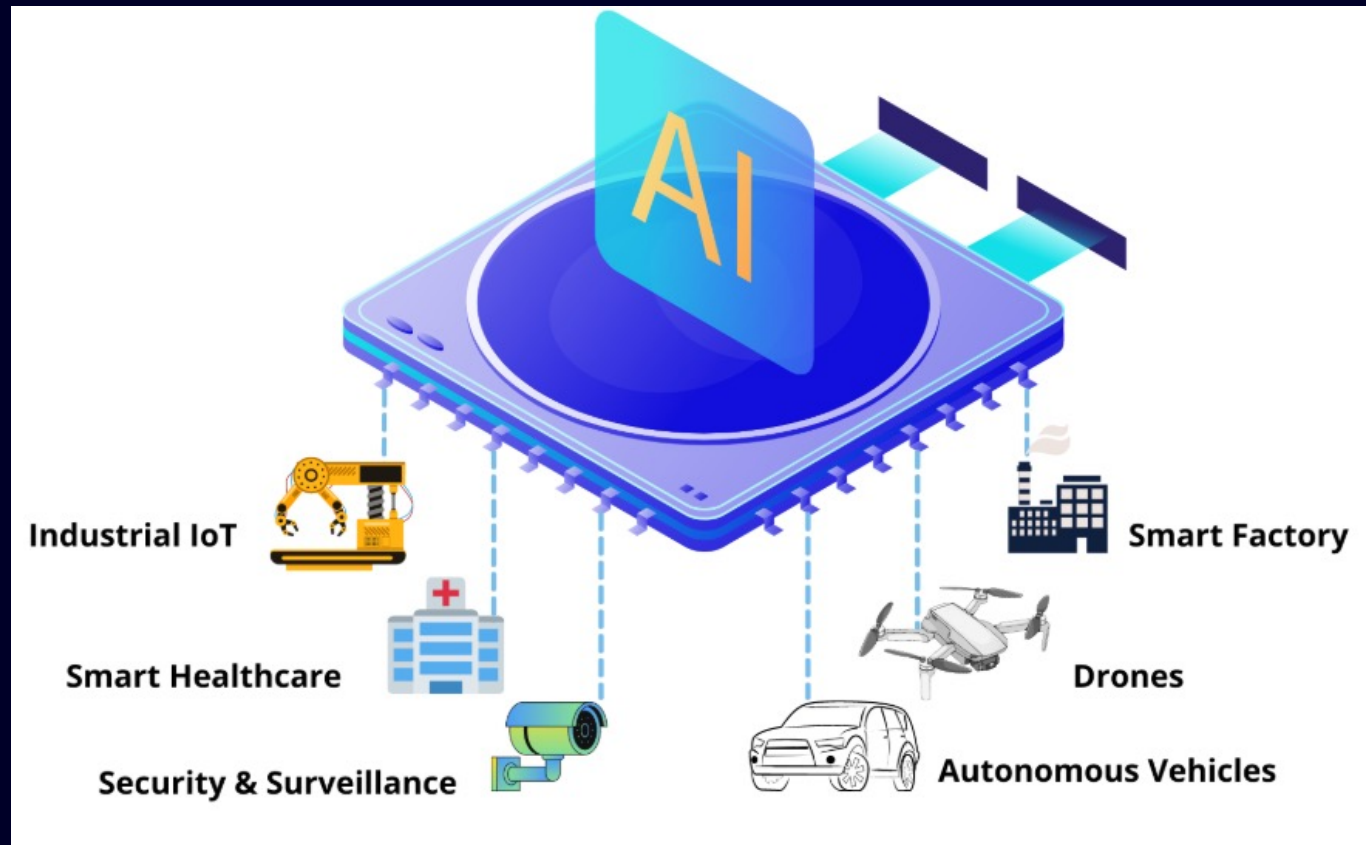
AI creates computer systems that can do human-like tasks, such as understanding language, finding patterns, and deciding. Machine Learning, a subset of AI, involves using algorithms that let computers learn from data and get better over time. ML models can find patterns, sort objects, and predict outcomes from examples.

MCUs play an important role in making AI and ML possible on edge devices. Some use cases for MCU based AI/ML at the edge include:

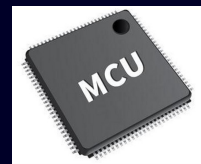
- **Keyword Spotting:** Recognize specific words or phrases (e.g., voice commands) without the need for cloud connectivity
- **Sensor Fusion:** Combining data from multiple sensors to make more informed decisions than with single sensor solutions
- **Anomaly detection:** Detecting outliers or abnormal patterns in sensor data that may indicate faults, errors or threats for predictive maintenance or quality control
- **Object detection:** Identifying and locating objects of interest (e.g., faces, pedestrians, vehicles) in images or videos captured by cameras or other sensors.
- **Gesture recognition:** Interpreting human gestures (e.g., hand movements, facial expressions, body poses) in images or videos captured by cameras or other sensors to improve human computer interaction

02 Challenges of AI/ML on MCUs

Traditional Edge AI



- Edge Artificial Intelligence combines edge computing, Internet of Things (IoT), and Artificial Intelligence (AI) technologies to provide real-time data collection, processing, analytics, and decision-making.
- The advances in edge AI, Machine Learning (ML), and Deep Learning (DL) technologies enable new capabilities at the network's edge, closer to the sensors and actuators that were previously impossible for conventional microcontroller unit (MCU) systems.



Big Challenge --- Can the small gourd hold the sky?



Journey to the West

.CLOUD

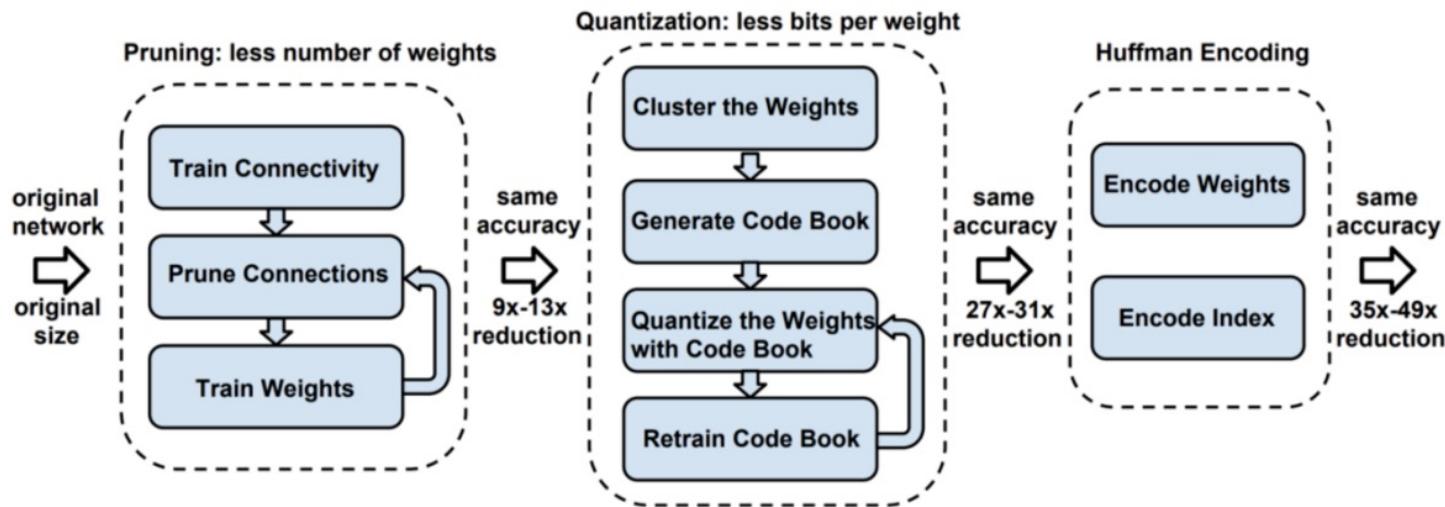
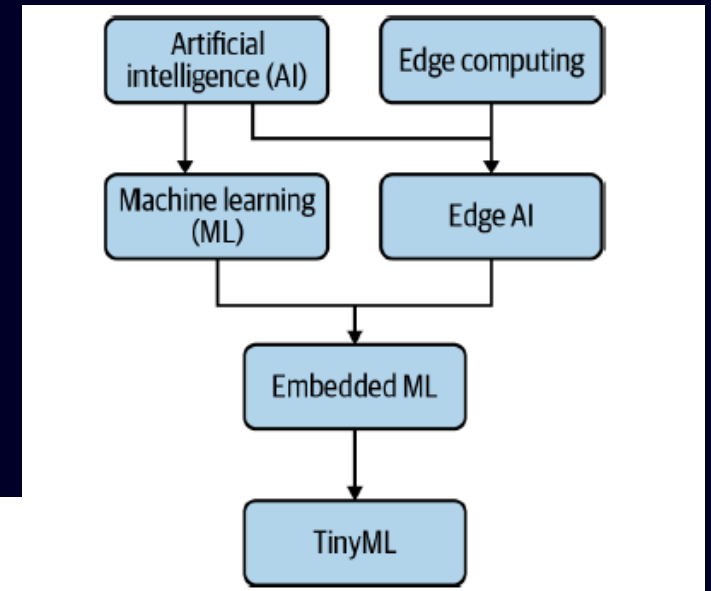


SIEMENS

03 MCU(Microcontroller Unit) AI/ML

What is TinyML?

TinyML refers to machine learning models and techniques optimized for deployment on resource-constrained devices. These devices operate at the edge, where data is generated, and inferencing is performed locally. Typically run on low power MCUs, TinyML systems perform inferences on data collected locally to the node. Inferencing is the moment of truth for an AI model, testing how well it can apply knowledge learned during training. Local inferencing enables MCUs to execute AI models directly, making real-time decisions without relying on external servers or cloud services.



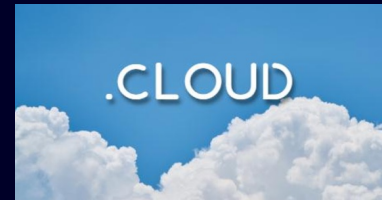
← Deep Compression

Local inferencing in the context of AI/ML is crucial

- **Resource Constraints:** Many embedded devices, especially those running on battery power, have limited resources such as memory, processing capability, and energy efficiency. Traditional general-purpose microcontrollers struggle to perform AI tasks efficiently due to their limited processing power and memory, constrained energy resources, or lack of on-chip acceleration. Local inferencing allows these resource-constrained devices to execute AI workloads without draining excessive power to improve efficiency and performance.
- **User Experience Enhancement:** Consider an example: an AI-enabled electronic face recognition. By training it to distinguish between employees and non-employees, it can open the door only for the employees. Here, local inferencing improves the user experience by ensuring safety and convenience without the need for additional hardware.
- **Efficiency and Performance:** GPUs are commonly used for large-scale AI deployments because they can perform many processes in parallel, essential for effective AI training. However, GPUs are costly and exceed power budgets for small-scale embedded applications. AI-optimized MCUs, with specialized architectures, strike a balance by delivering better performance and power efficiency for AI workloads.

04 Two use cases for MCU based AI/ML

Gold Horn the senior demon king



Tensorflow Lite

What is Tensorflow Lite

- It is a method to increase speed. https://www.tensorflow.org/lite/performance/best_practices
- A small function library that does not train or change the model, but can only run the (inference) model. <https://ubuntu.com/blog/ai-inference-on-edge-with-tensorflow-lite>
- Why such a big name?

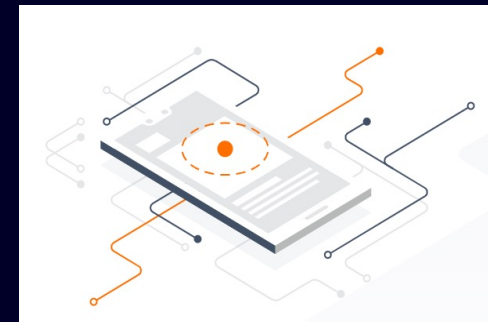
Tensorflow Lite Flow

- Train the model on a PC.
- Use TensorFlow Lite to optimize the model.
- Put the optimized model on a mobile or embedded device for inference.



Tensorflow Lite

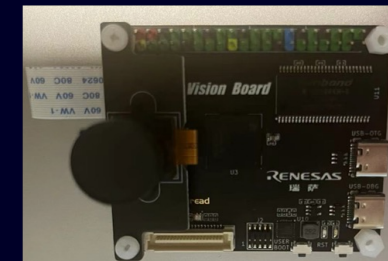
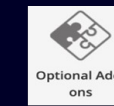
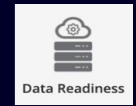
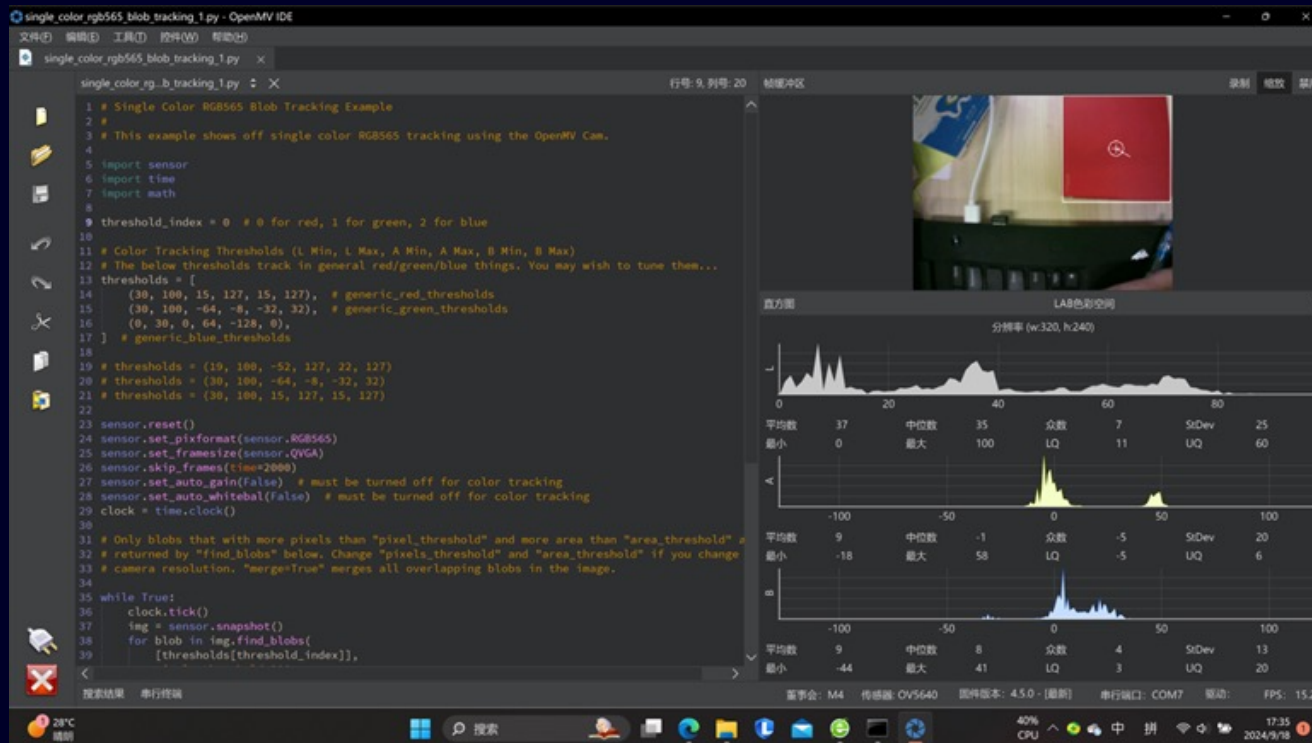
Reasoning on mobile(embedded)
devices



TensorFlow-Lite: A basic demo for AI on MCU

Current Status:

The AI at MCU could recognize colors as the bellowing picture shown.



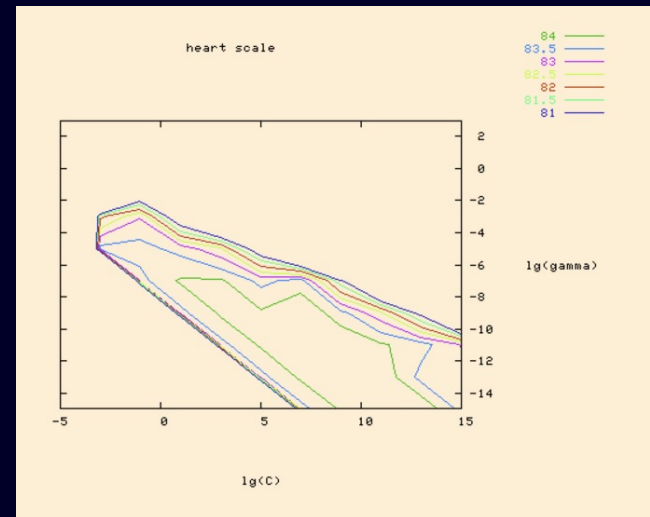
Silver Horn the junior demon king



What is LibSVM

libsvm is a widely used library for support vector machines (SVMs) and has been implemented for various machine learning tasks such as classification, regression, and other pattern recognition problems. It provides both a simple interface for users and a flexible implementation for advanced users.

Indeed, libsvm mainly focuses on the training part of the support vector machine (SVM), especially in terms of model training optimization, parameter tuning, and cross-validation. In contrast, its implementation in the inference (prediction) stage is relatively simple, usually only including the processing of input data and the steps of applying the trained model for classification or regression prediction.



On-device training fan anomaly detection on MCXN947

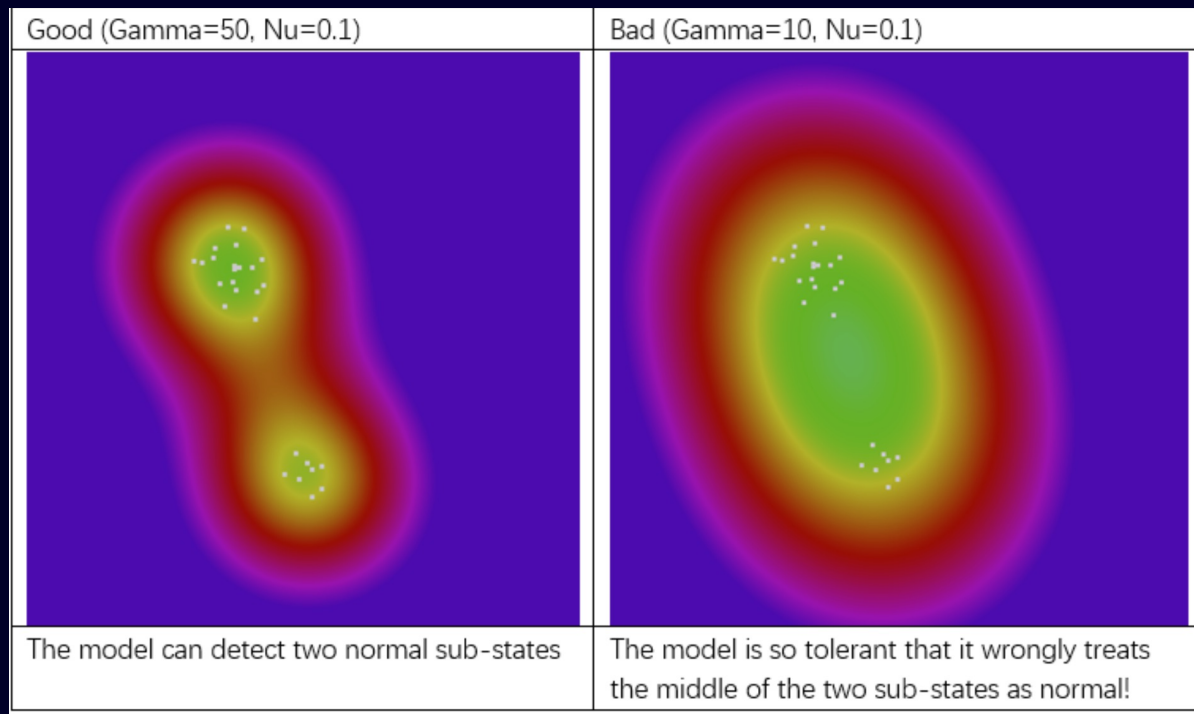
- On-device trainable anomaly detection based on the MCX platform. Train the SVM model on-device with normal accelerometer data and inference SVM model on-device too. Support the Inc-training, which means that we can define one-feature and train the model first, then define second feature and train the model again. The final model can classify both the feature. To achieve this, click the Train, then at the training page, click the IncTrain first, then start. The model will try to learn both the feature.
- Because data patterns are often very different or vary among the device life cycle. During the whole life cycle of a device, learning may need to be done multiple times.
- The GUI is implemented by LVGL.

The 'Param' setting in training windows

The Gam is “Gamma (γ)”, Nu is “Nu (ν)”, they are two parameters to tune the trade-off between sensitivity and tolerance of the trained model.

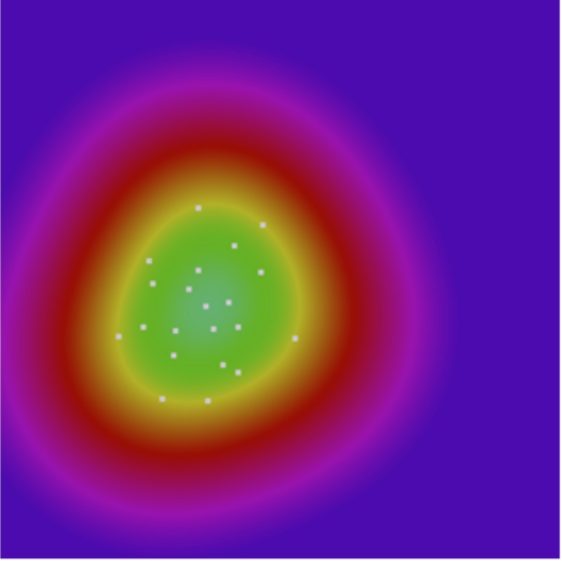
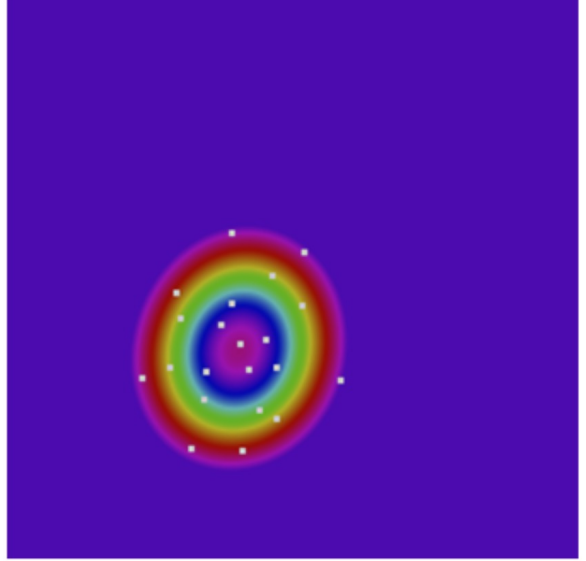
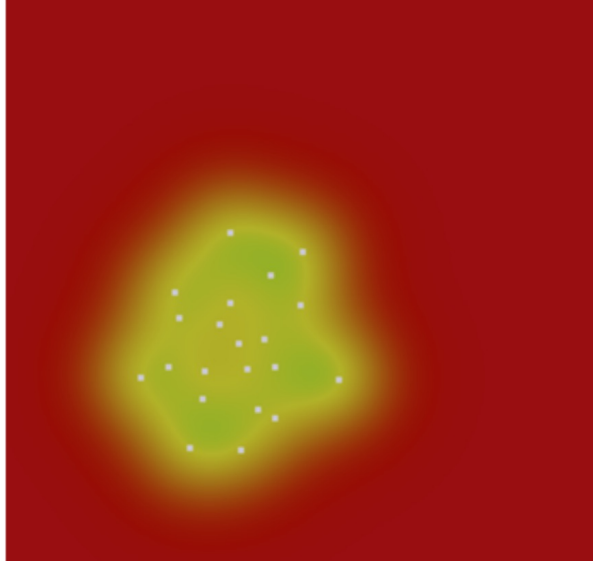
2D color contour visualization of the model’s decision boundary on the LCD, the color in green to blue are normal region, while the color in yellow-red-purple are abnormal regions.

Two normal sub-states that are significantly different



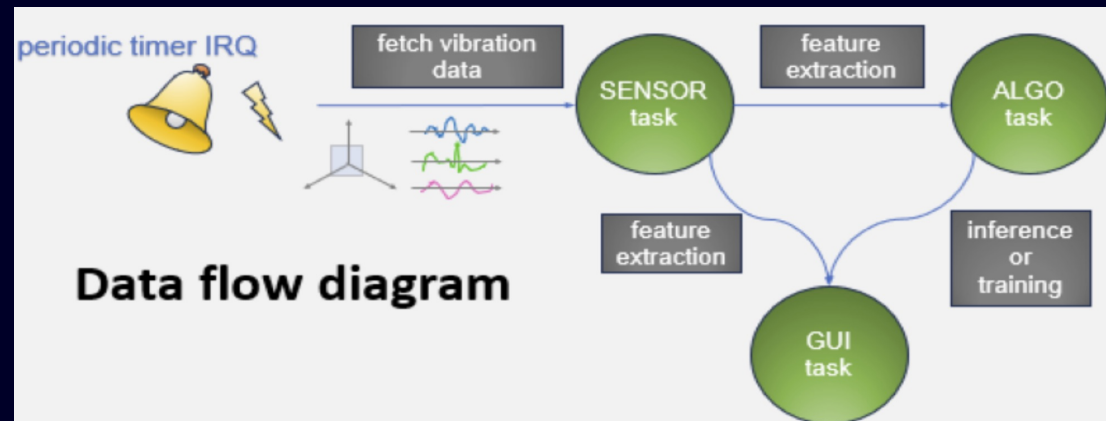
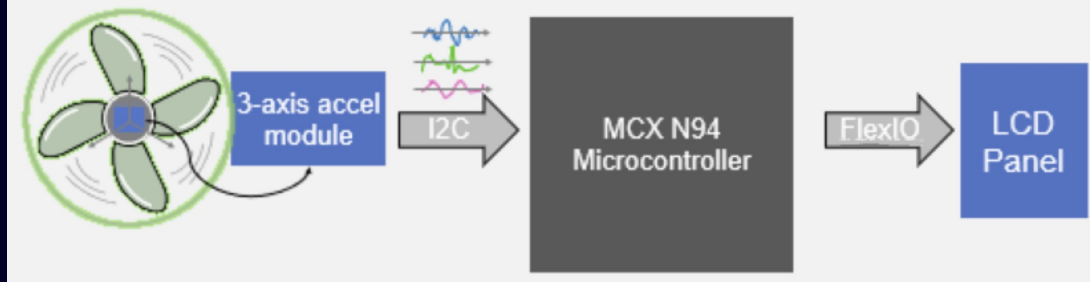
The 'Param' setting in training windows --- Cont.

One normal sub-state, but data variance is large (due to disturbances)

| Good (Gamma=50, Nu=0.1) | Bad 1 (Gamma=50, Nu=0.4) | Bad 2 (Gamma=300, Nu=0.1) |
|--|--|---|
|  |  |  |
| <p>The model can tolerate high disturbance.</p> | <p>Though the model can recover health very fast with samples that are near to average, the model wrongly treats the samples that are far from average to abnormal</p> | <p>The model clusterizes the samples too much (samples' effective range is too small), and even the region near the average can be treated as abnormal due to lack of training samples!</p> |

LibSVM: a basic demo for AI on MCU

Block diagram



**Thank you for your listening !
See you next time !**

A 3D rendering of the letters 'Q&A' in a blue, sans-serif font. The letters are thick and have a slight shadow underneath, giving them a three-dimensional appearance. They are set against a plain white background.