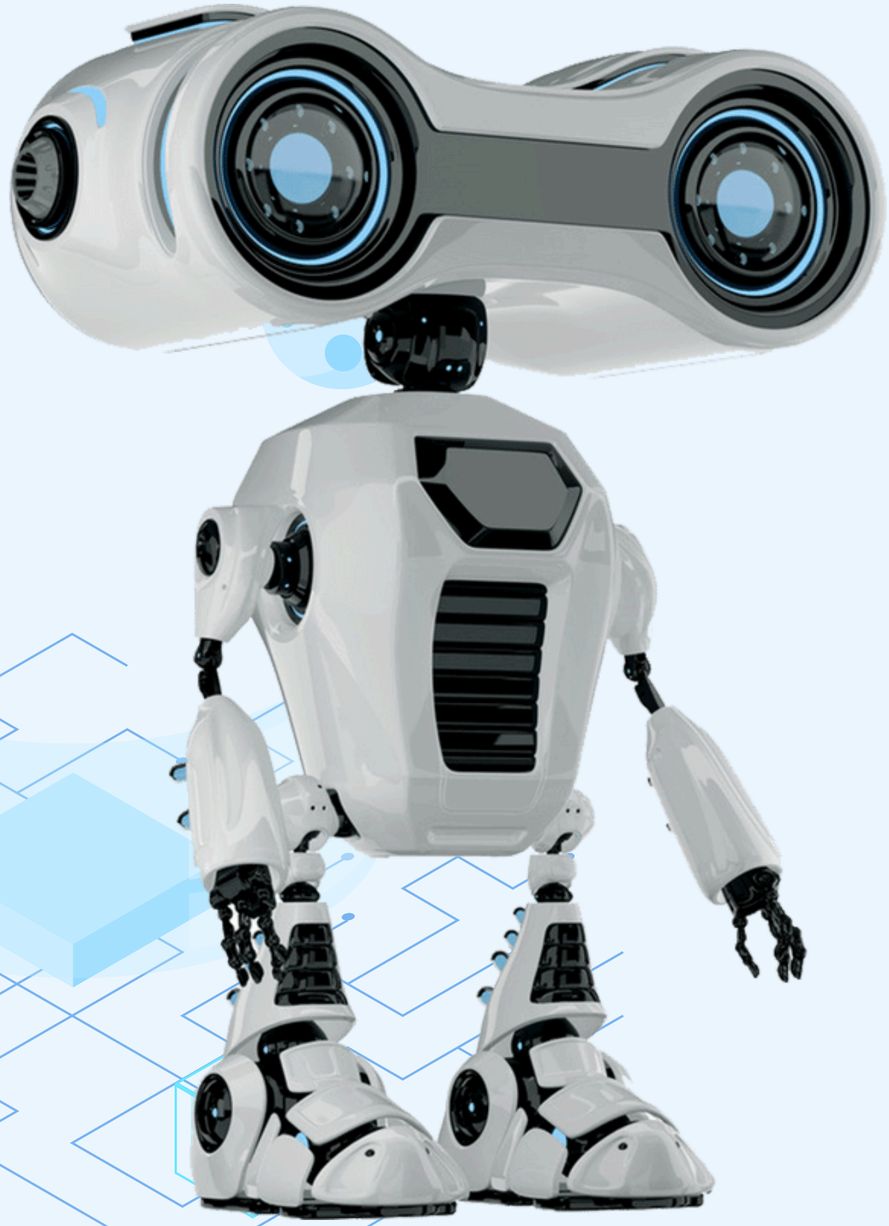


TINYML PROJECT

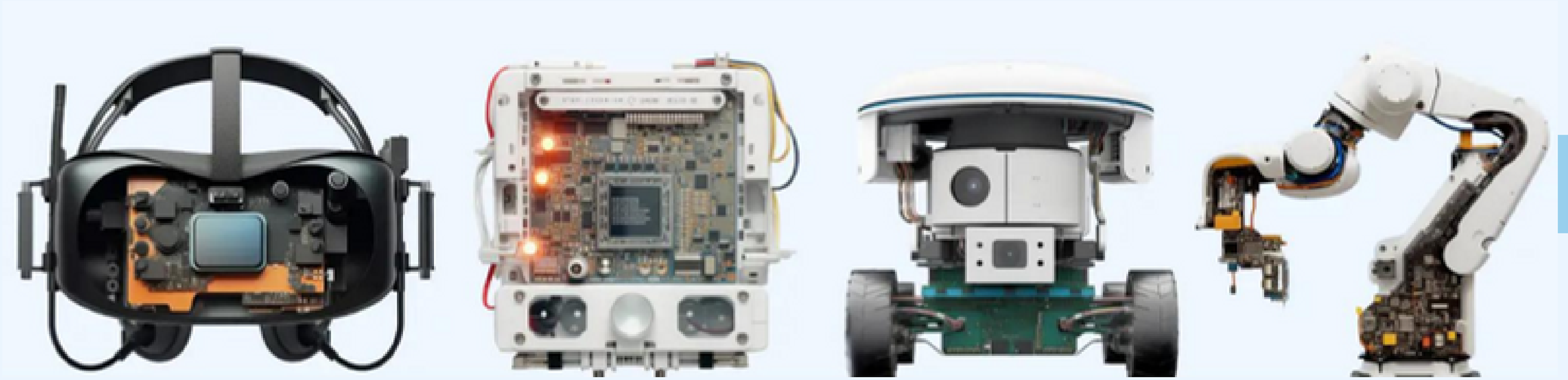
salsabil JABALLAH & Meriem BRAHEM
Supervisor : Prof.SAKRANI

Plan :

- Introduction To TinyML
- Introduction To Edge Impulse
- Project Overview
- Project Workflow
- Code Architecture
- Results and Live Demonstration
- Prespectives & Conclusion

Introduction To TinyML

- **Definition:** TinyML is the deployment of machine learning models on microcontrollers and embedded systems with limited resources.



Introduction To TinyML

Applications:

- Predictive maintenance in industrial equipment.
- Gesture and motion detection in IoT devices.
- Environmental monitoring (e.g., detecting anomalies in sensor data).
- Wearables and health monitoring.

Why TinyML?

- Low power consumption.
- Real-time processing.
- Cost-effectiveness.
- Privacy and data security (processing data locally).

Introduction To Edge Impulse

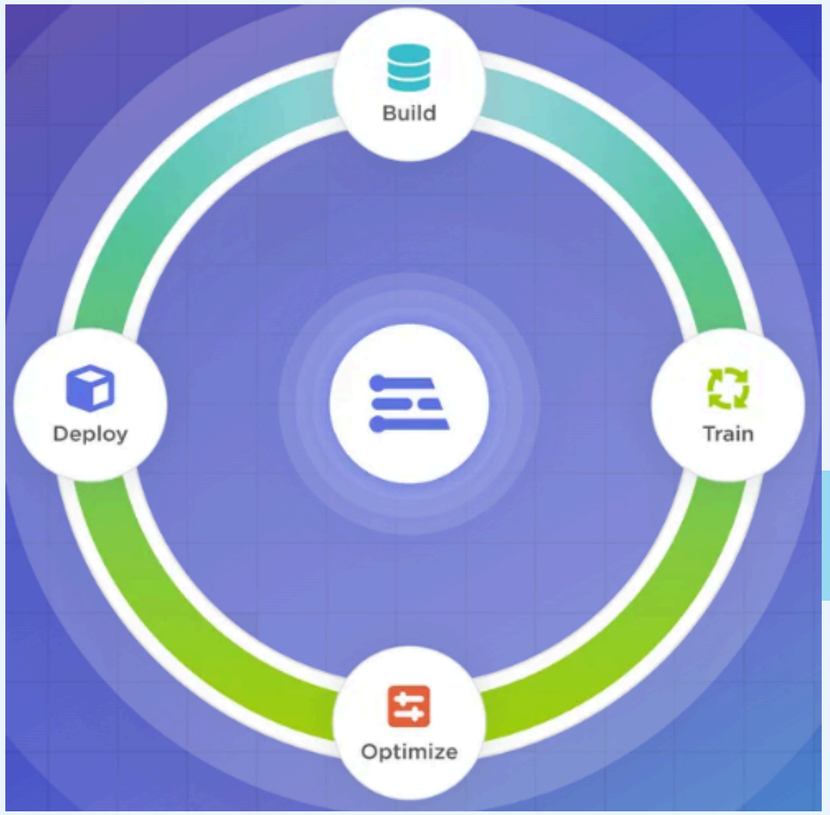
The Leading Development Platform for Edge ML

EDGE IMPULSE

Build datasets, train models, and optimize libraries to run directly on device; from the smallest microcontrollers to gateways with the latest neural accelerators (and anything in between).

Any Data. Any Model. Any Device.

Accelerate Innovation with Seamless Collaboration for Production-Ready Models



EDGE IMPULSE

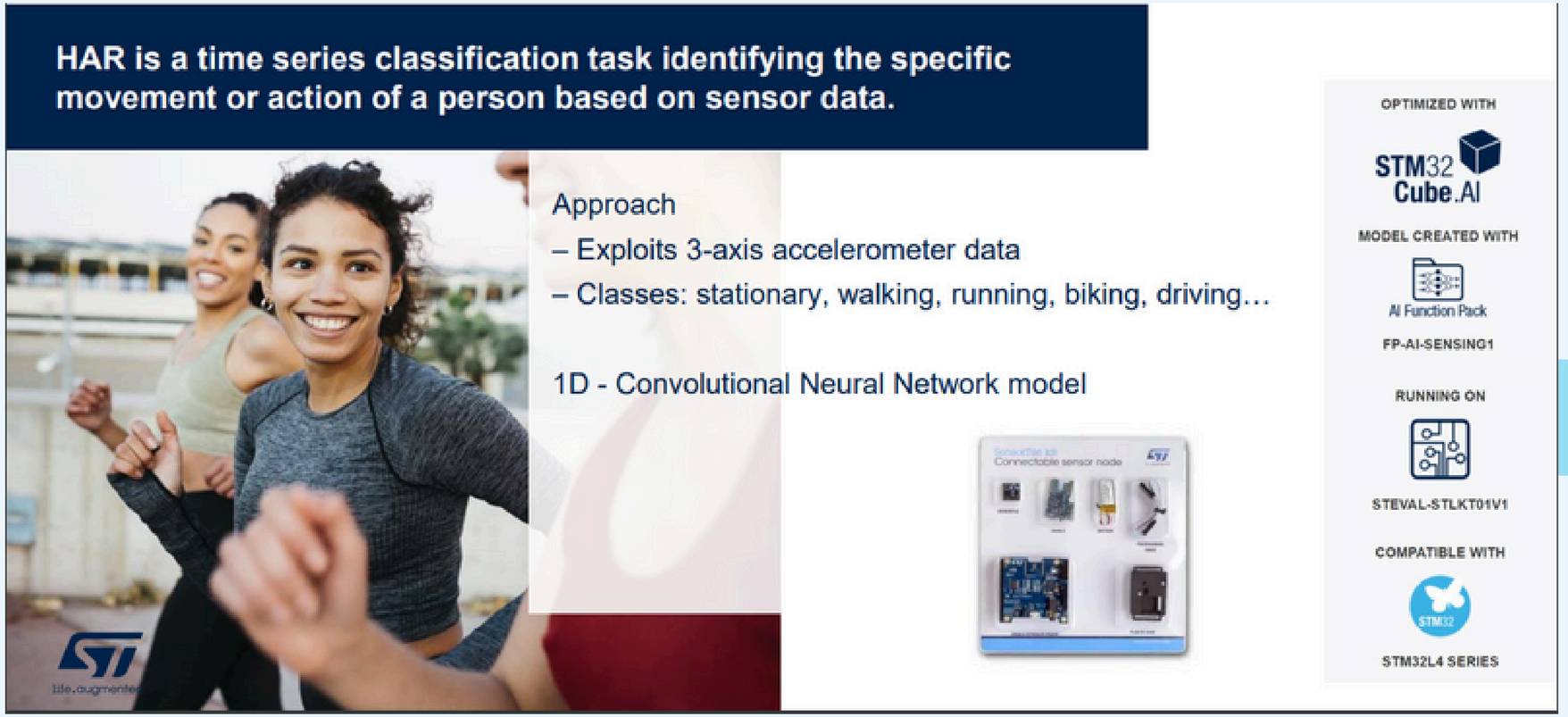
Project Overview

Objective: To implement a motion detection model on the STM32F407VG board using TinyML principles.

Project Overview

Why motion detection?

- Real-world applications in robotics, fitness tracking, and fall detection for elderly people.
- Utilizes the onboard accelerometer (or external MPU6050 sensor).



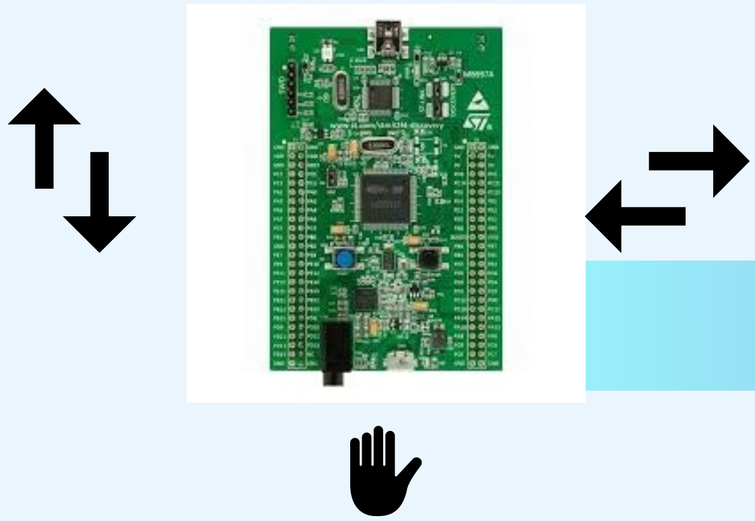
Project Overview

Optimization:
Simplifying the idea of motion detection to fit in our board with Cortex-M4 microcontroller using onboard accelerometer.
Our application consist of motion detection system to classify three motions:

1. Up-Down
2. Right-Left
3. Idle (no movement)

Problem Breakdown

- How Motion is Detected:
  - Accelerometer provides raw data for X, Y, and Z axes.
  - Patterns in this data are used to classify movements.
- Challenge:
  - Efficiently process data on limited hardware while maintaining real-time classification.



Project Workflow

HARDWAR E SETUP | DATA COLLECTION | MODEL TRAINING | MODEL DEPLOYMENT

Hardware Setup:

- Use STM32F407VG Discovery Board with onboard accelerometer.
- Connect to a computer via USB for data transfer and debugging.

Data Collection:

- Record accelerometer data for each motion:
  - Up-down: Varying Z-axis values.
  - Right-left: Varying X-axis values.
  - Idle: Stable X, Y, and Z values.
- Use the Edge Impulse Data Forwarder to stream data.

Model Training:

- Train a Neural Network Classifier in Edge Impulse using collected data.
- Validate and test the model for accuracy.

Model Deployment:

- Download the trained model as a C++ library.
- Integrate it with the STM32 project using the Edge Impulse SDK.
  -

Code Architecture

- **Header Section:**
- Includes libraries for STM32 HAL, Edge Impulse SDK, and accelerometer.
- **Key Components:**
- features[]: Stores raw accelerometer data.
- get_feature_data(): Feeds data into the classifier.
- run_classifier(): Runs the Edge Impulse model for classification.

How It Works:

1. **Inference Phase:**
    - Captures raw data from the accelerometer (BSP_ACCELERO_GetXYZ).
    - Buffers data and passes it to the run_classifier() function.
2. **Classification Output:**
    - Up-Down → Based on Z-axis variations.
    - Right-Left → Based on X-axis variations.
    - Idle → Minimal changes in all axes.

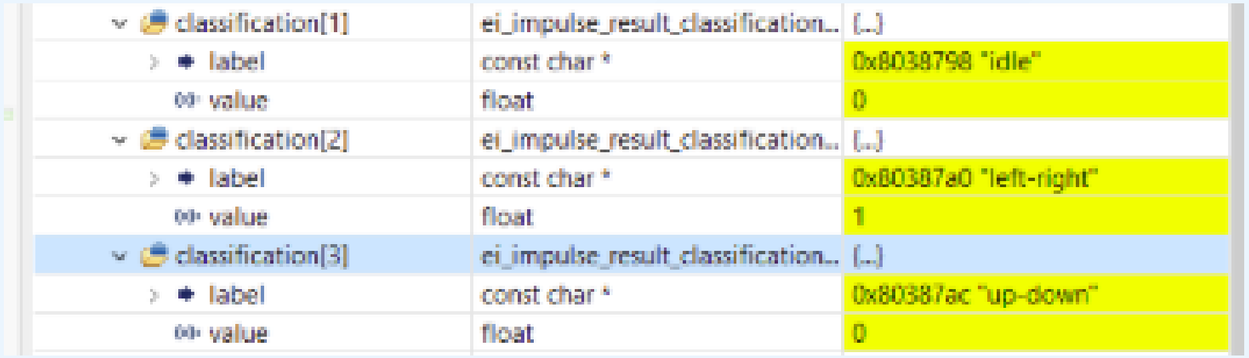
Real-Time Behavior:

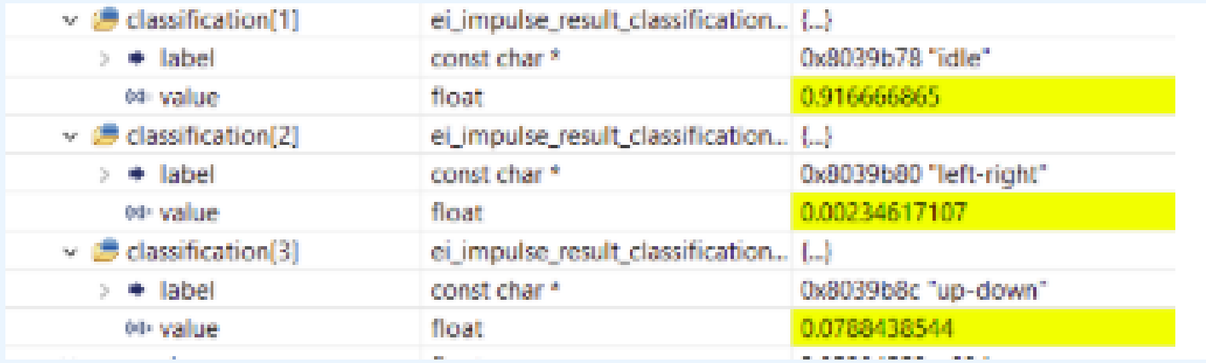
- The result is stored in the variable result.
- LED indicators or UART output can be used to display the classified motion. (we faced some problems implementing this point so we opted for just checking the variable result in debug mode in live)

Results and Live Demonstration

Expected Results

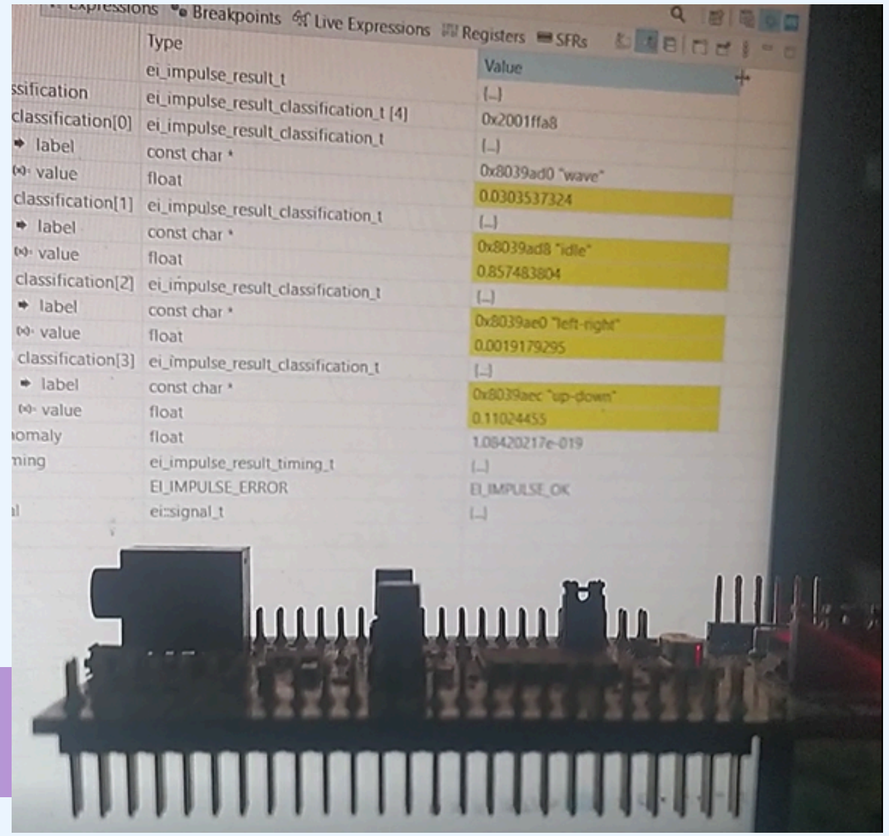
- Real-Time Testing:
- Flash the STM32 board with the model in Inference Phase.
- Perform the motions:
- Move the board up-down → Classified as Up-Down.
- Move the board left-right → Classified as Right-Left.
- Keep the board idle → Classified as Idle.
- Observe accurate classification of motions in real time.





=> **In debug mode, the result variable was added to the live variable watch, allowing us to track its changes in real time as the direction movements of the STM32 were adjusted.**

Prespectives & Conclusion



- TinyML allows us to bridge the gap between machine learning and resource-constrained devices (low cost MCUs).
- As Students, STM32 and Edge Impulse provided us with an excellent ecosystem to learn more about embedded AI development and practice it on real projects.

THANK YOU !