Bringing Intelligence to Cyber Physical Systems via Compression and Quantization Techniques for Anomaly Detection in Industry 4.0

Dario Bruneo, Fabrizio De Vita

Università degli Studi di Messina
Messina, Italy

6th Italian Workshop on Embedded Systems, Rome 9-10 December

Introduction - Cyber Physical Systems

Cyber Physical System (CPS) and Internet of Things (IoT) based smart services and applications totally revolutionized the way we interact with the physical world

Today, smart environments represent the maximum expression of CPS:

- Smart cities;
- Smart industry;
- Smart home;
- …

Cloud and Edge computing paradigms play a fundamental role for the realization of these systems:

- large storage;
- high computing capabilities;
- pervasive monitoring;
- early processing of sensors data;
- …

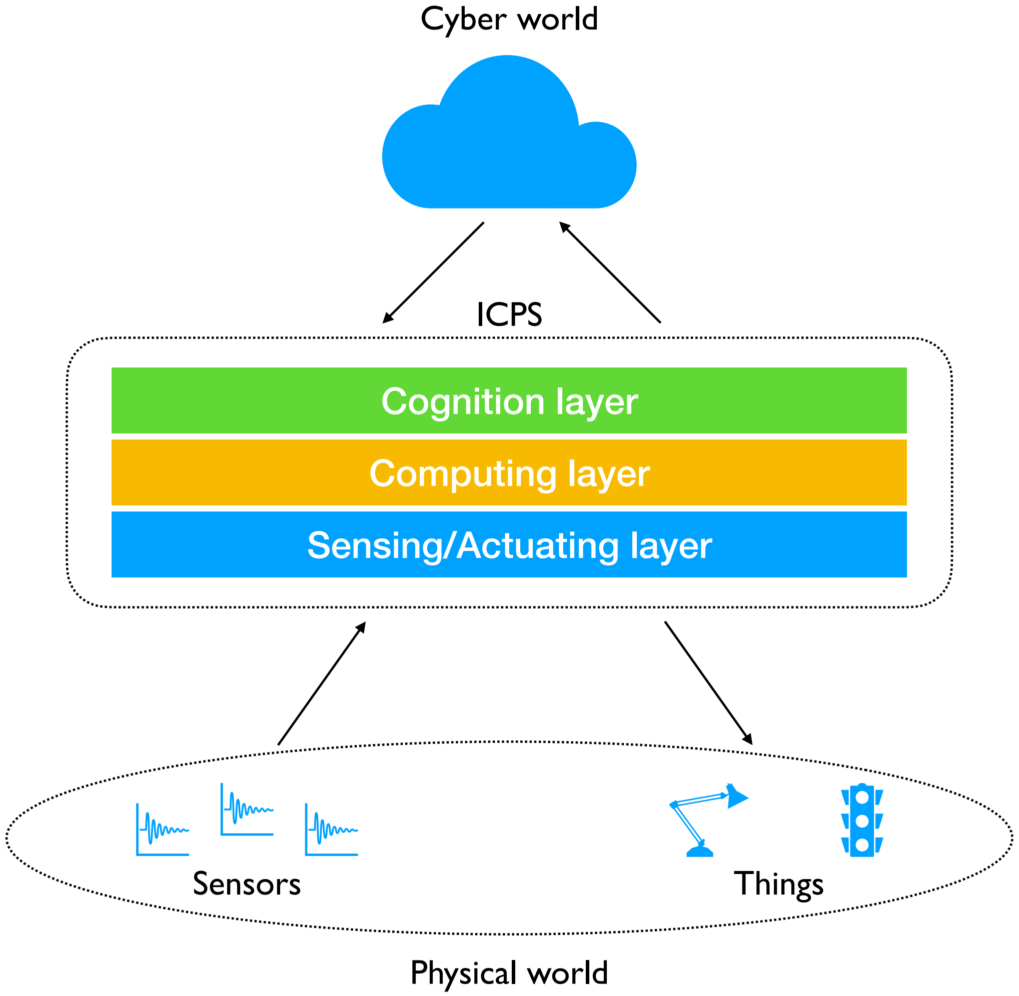
Artificial Intelligence and Intelligent Cyber Physical Systems

Artificial Intelligence is another important player for the realization of smart environments

- Decision making;
- Context awareness;
- Intelligent services;
- …

AI plays a decisive role in bringing the intelligence on CPS

Through the addition of a *Cognition* layer a Cyber Physical System becomes "Intelligent" and able to make "reasonings"



ICPS represent the core of modern frameworks capable of delivering a "reasoned" support to the human being during his daily activities

Smart Industry

In smart industry, aspects related to prognostics and diagnostics are gaining a lot of interest in the recent period

The timely prevention of faults or an anomalies can be crucial

Understand the “health” state of an industrial system is challenging

- strong non-linearities;
- sensors heterogeneity;
- large number of variables.

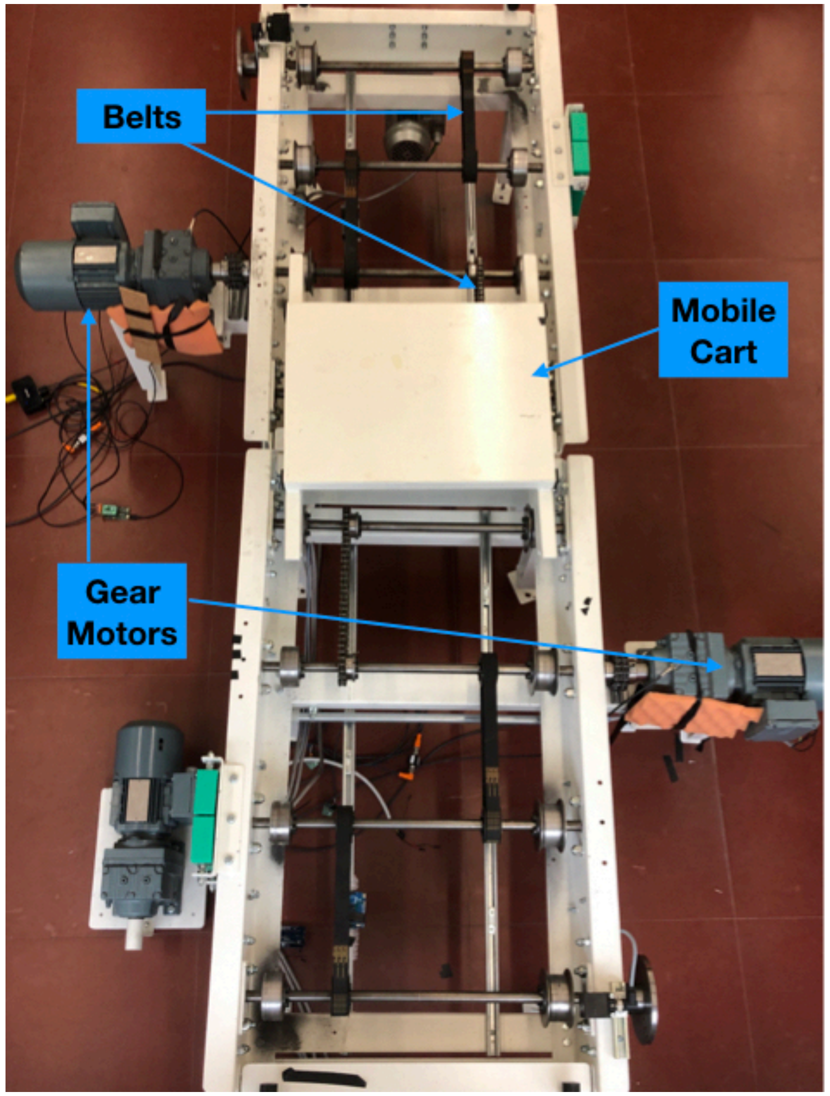
Today the majority of the solutions involve the use of machine and deep learning techniques to address this problem

ICPS became one of the key enabling technologies to perform an on-site anomaly detection

Anomaly Detection in Industrial IoT Systems

Work done in collaboration with LOMA S.r.l. and SmartME.io S.r.l.

The system is a scale replica of an assembly plant for transportation of car pieces equipped with 2 motors and six belts



Sensor Instrumentation:

- Temperature sensor;
- Vibration sensor;
- Distance sensor;
- Absorbed current sensor;
- Noise sensor.

Possibility to inject different types of faults:

- Introduce external vibrations;
- Change belts tension;
- Increase the friction of the gears;
- etc…

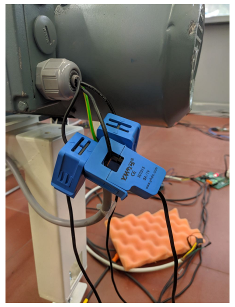
F. De Vita, D. Bruneo, and S. K. Das, "On the use of a full stack hardware/software infrastructure for sensor data fusion and fault prediction in industry 4.0," Pattern Recognition Letters, vol. 138, pp. 30 –37, 2020, issn: 0167-8655. doi: https://doi.org/10.1016/j.patrec.2020.06.028.

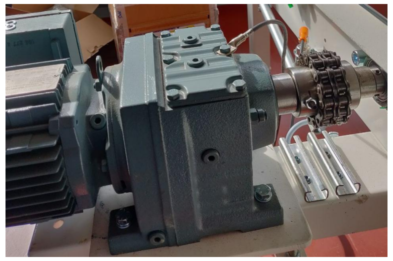
Anomaly Detection in Industrial IoT Systems - Data Collection Framework

Distance

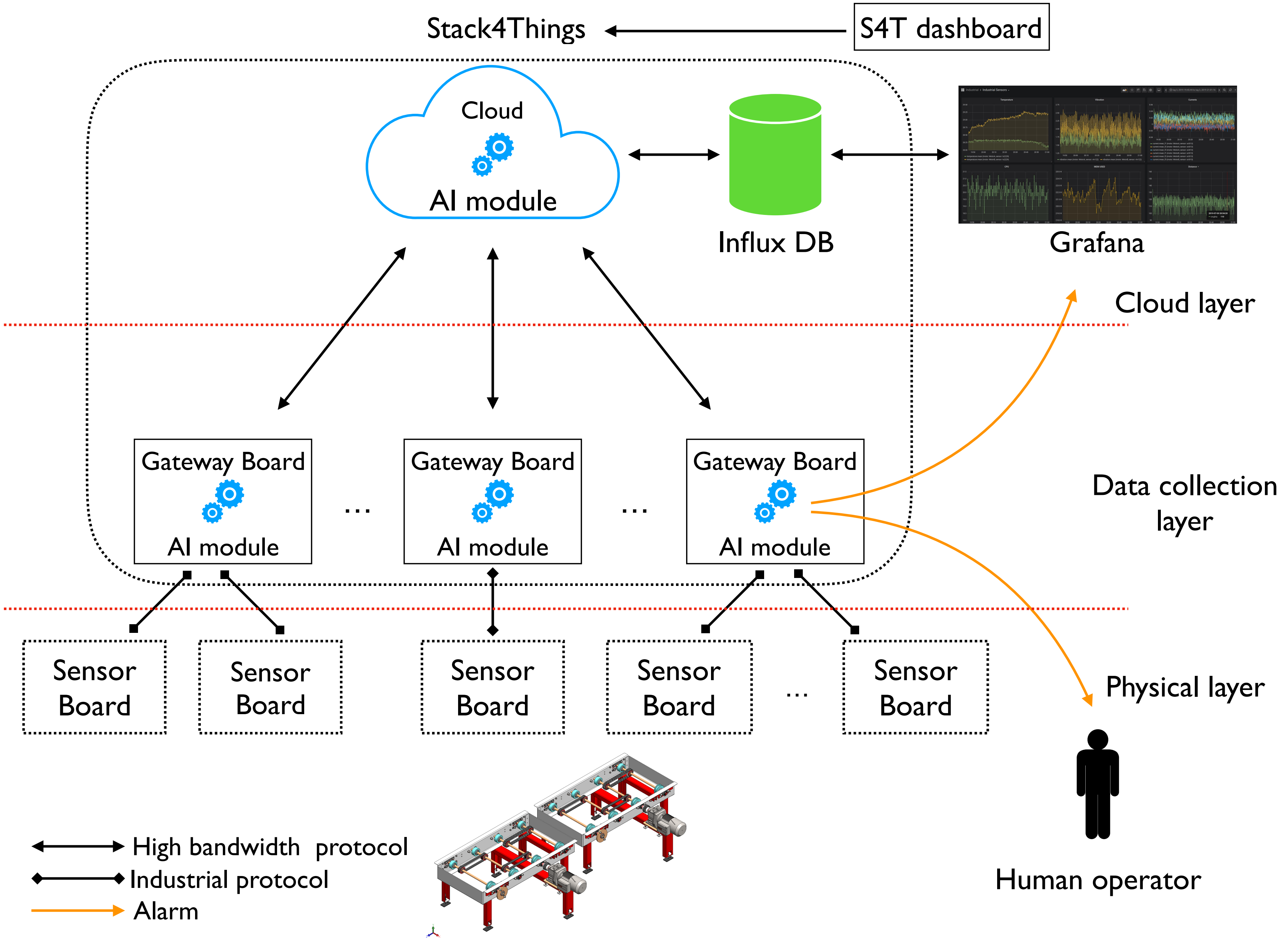
Noise

Vibration


Current


Temperature

Through Stack4Things, the framework can orchestrate IoT devices without caring their location and underlying technology


Stack4Things

S4T dashboard

Cloud

AI module

Influx DB

Grafana

Cloud layer

Gateway Board

AI module

Gateway Board

AI module

Gateway Board

AI module

…

…

Data collection layer

Sensor Board

Sensor Board

Sensor Board

Sensor Board

…

Sensor Board

Physical layer

Human operator

High bandwidth protocol

Industrial protocol

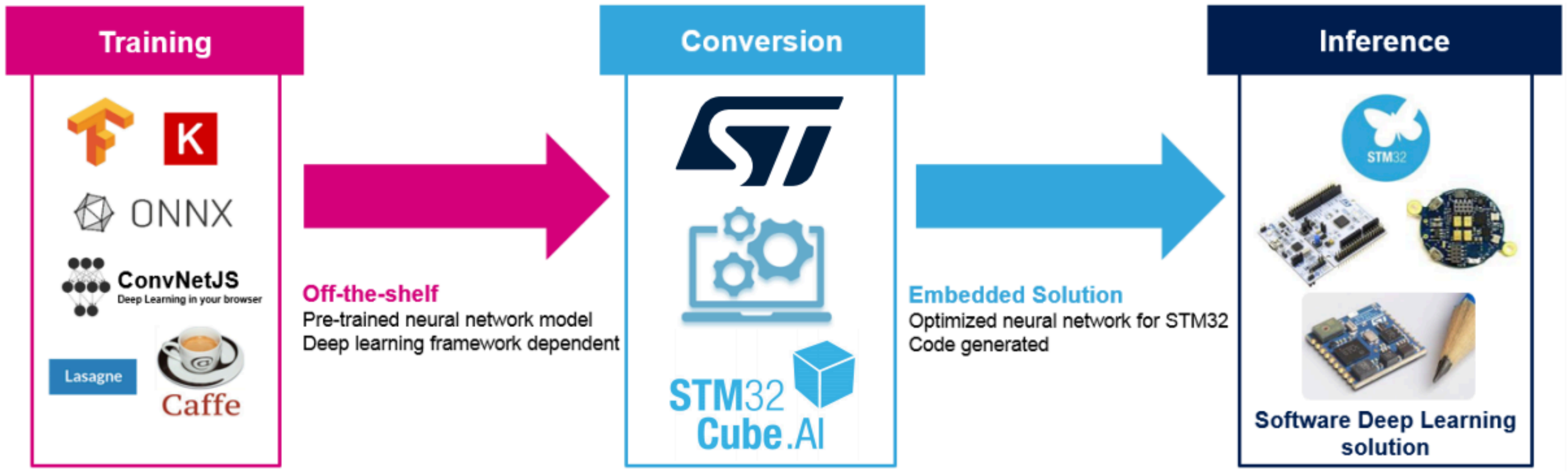
Alarm

Gateway board

Deploy ML models on smart boards with Micro Controlling Unit (MCU)

- low power consumption;
- real-time inference (MCU run algorithms);
- low cost hardware.

STM32CubeMX X-CUBE-AI tool



Training

Off-the-shelf
Pre-trained neural network model
Deep learning framework dependent

Conversion

Embedded Solution
Optimized neural network for STM32
Code generated

Inference

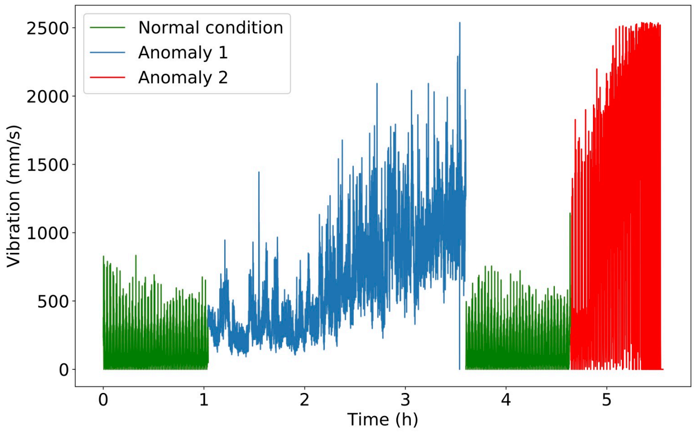
Software Deep Learning solution

Two conversion techniques:

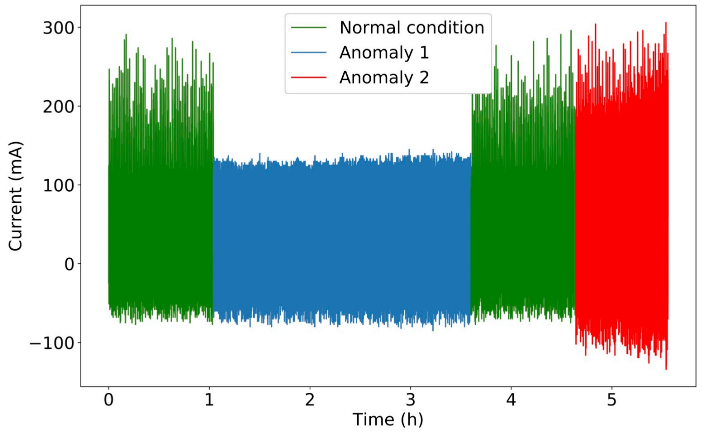
- Weights compression;
- Quantization.

Sensor data and Features Extraction

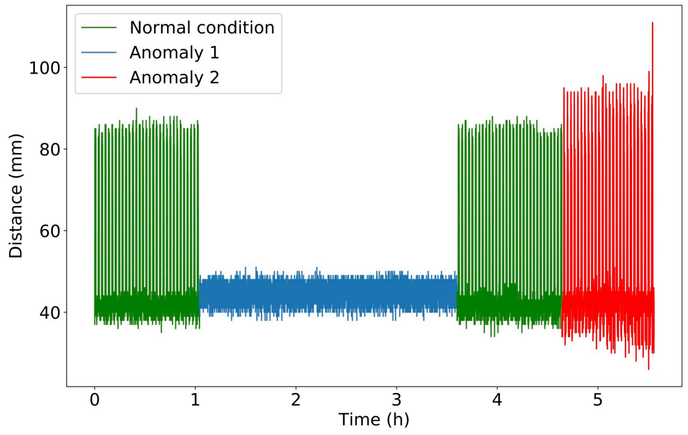
After an exploratory data analysis approach, we were able to extract only those features that allowed a discrimination between a normal and an anomalous condition



Vibration



Current



Distance

The plant has been analyzed under three operating settings:

- Normal condition;
- Anomaly 1 condition (i.e., proximity switch fault);
- Anomaly 2 condition (i.e., brake system fault).

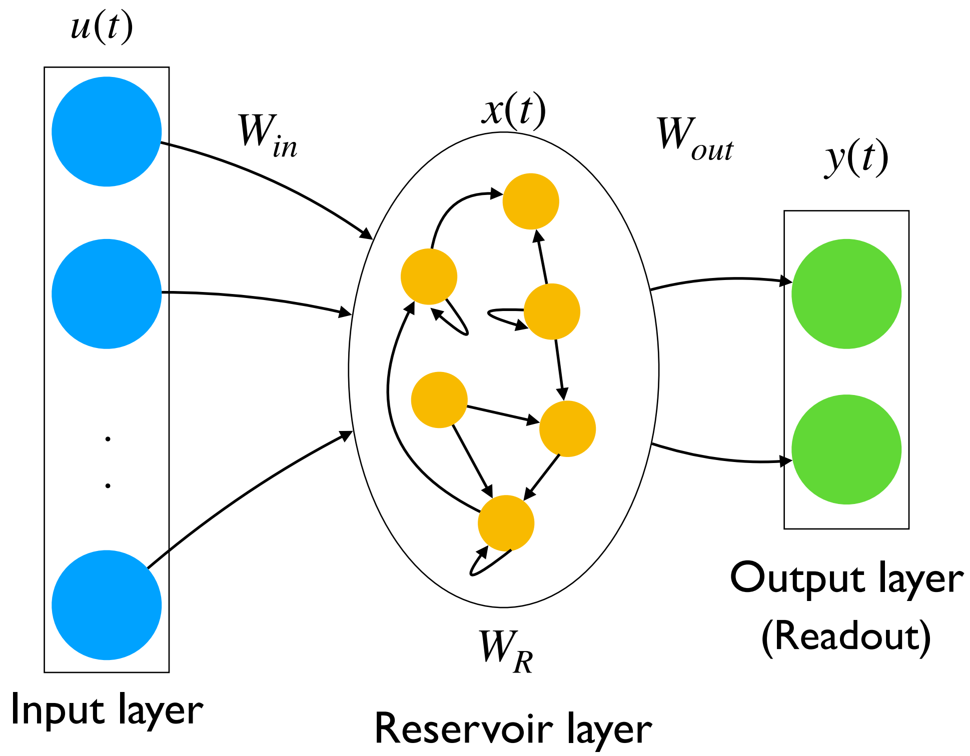
Echo State Networks

Echo State Networks (ESNs) belong to the class of *reservoir models*

An ESN is a Recurrent Neural Network (RNN) with a sparse randomly connected recurrent structure (the reservoir) and an output part called *readout*

$x(t+1) = f(W_{in} \cdot u(t+1) + W_R \cdot x(t))$ (State equation)

$y(t+1) = f(W_{out} \cdot x(t+1))$ (Output equation)

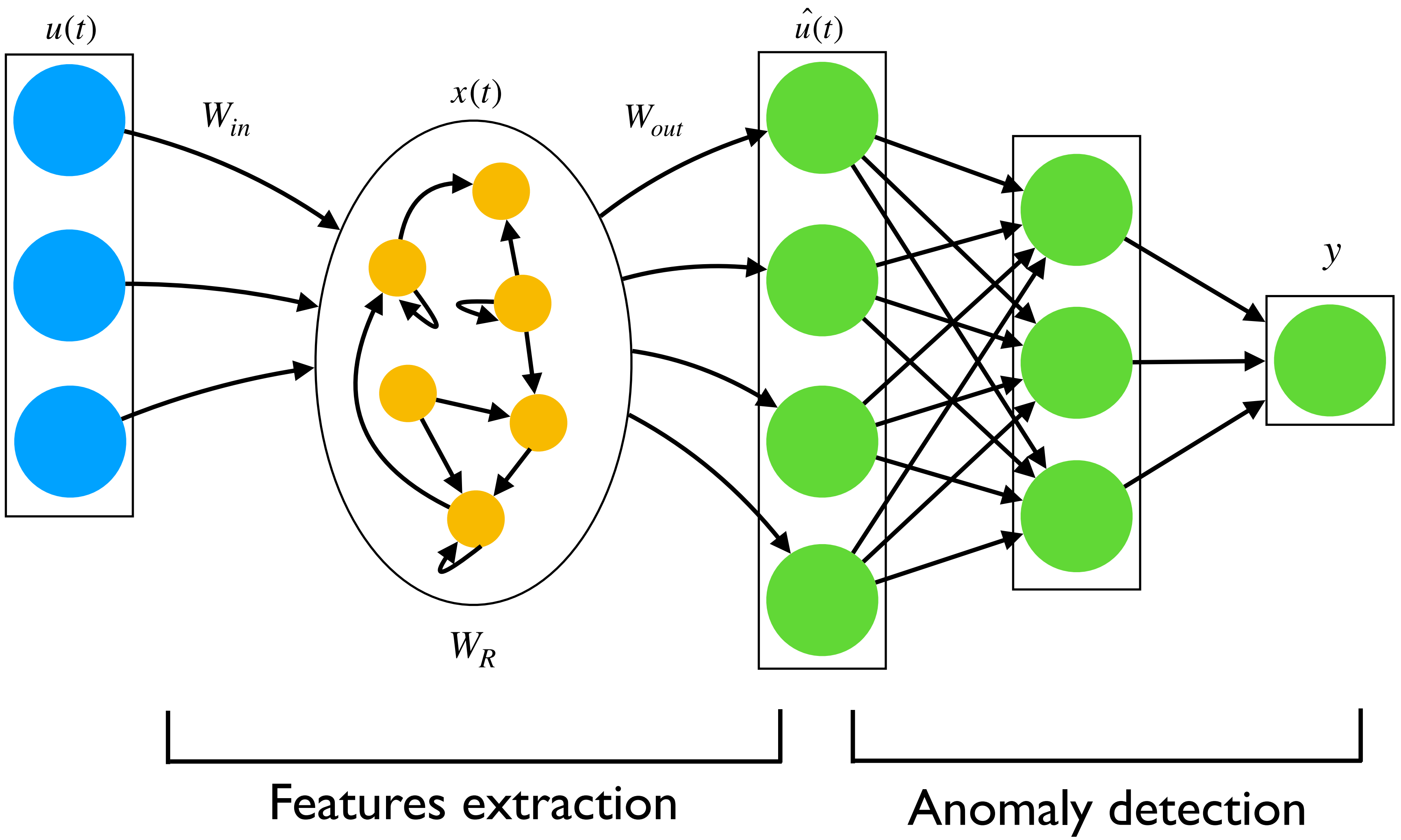


The reservoir weights (i.e., W_R) are randomly set and remain fixed during the entire training procedure

The only trainable weights are those connecting the reservoir and the readout (i.e., W_{out})

- Reduced model complexity
- Faster training

Proposed solution

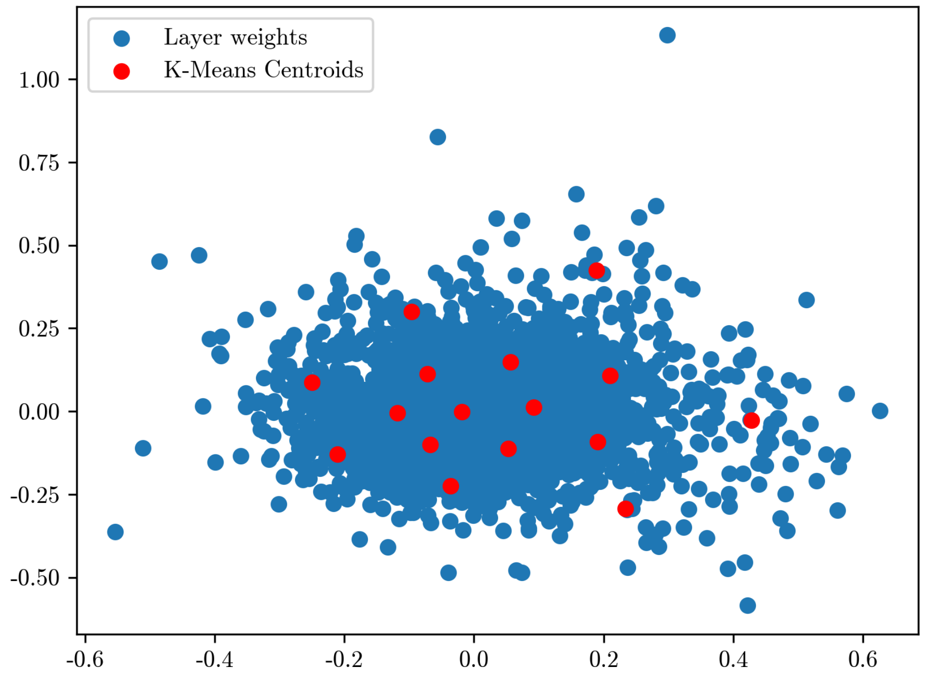


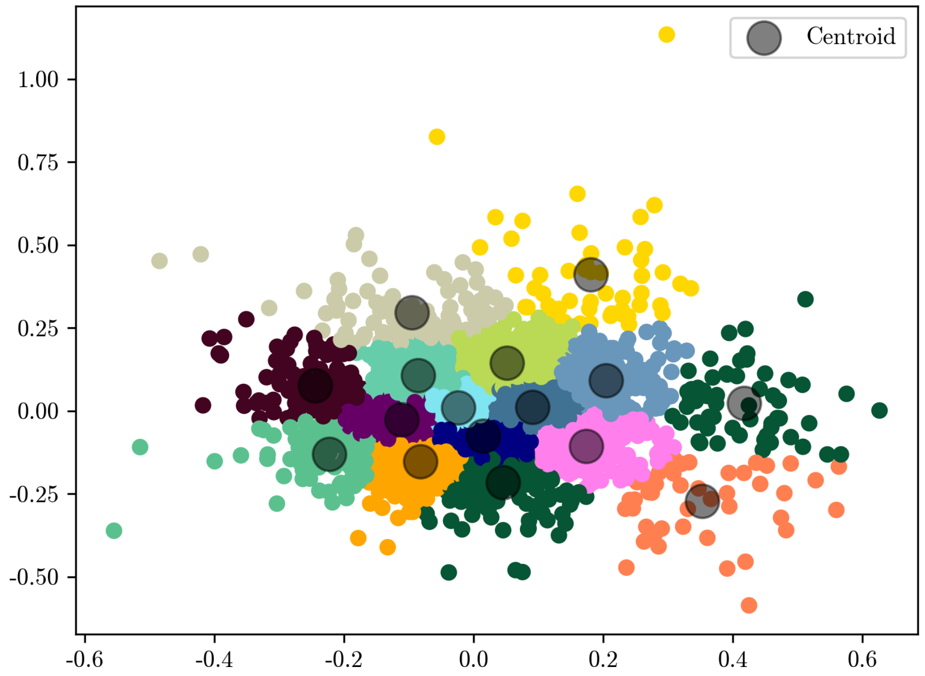
We propose a combination of an ESN and a fully connected network to perform the anomaly detection of the plant

The network is structured in two parts:

- Features extraction
- Anomaly detection

Weights compression





Weights compression is a viable solution to reduce the model memory footprint to fit the hardware constraints

Applicable only to dense layers where the most part of weights is concentrated

K-Means algorithm is used to cluster the layer weights into a reduced number of centroids

The number of centroids depends on the target_factor fixed for the compression as follows:

$$n_{centroids} = 2^{32/target\_factor}$$

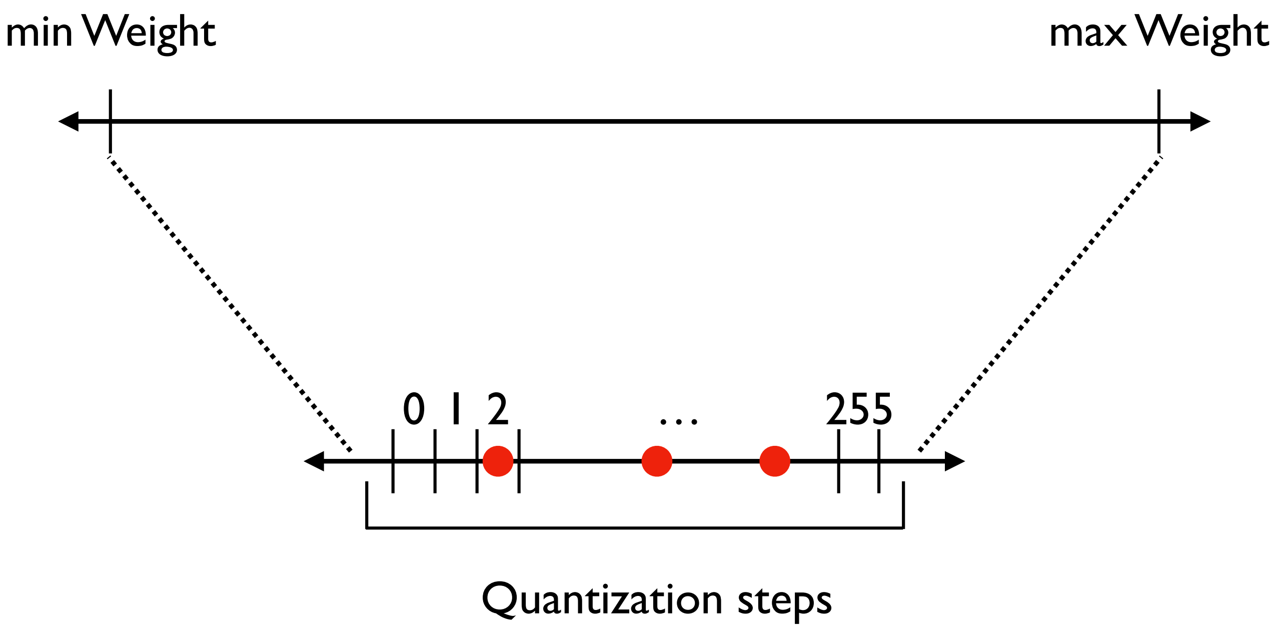
Quantization

Quantization reduces the memory footprint of the model while improving CPU performance

- Weights, Biases, and activation functions are converted from Float to 8 bit precision
- Generation of an optimized C code.

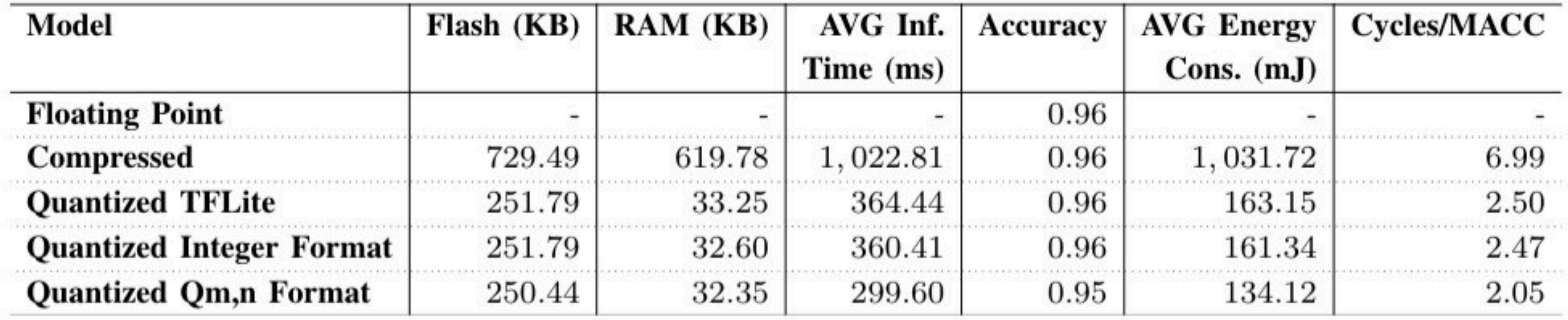
Two types of quantization supported

- Integer quantization
- Fixed-point quantization (Qm,n format)

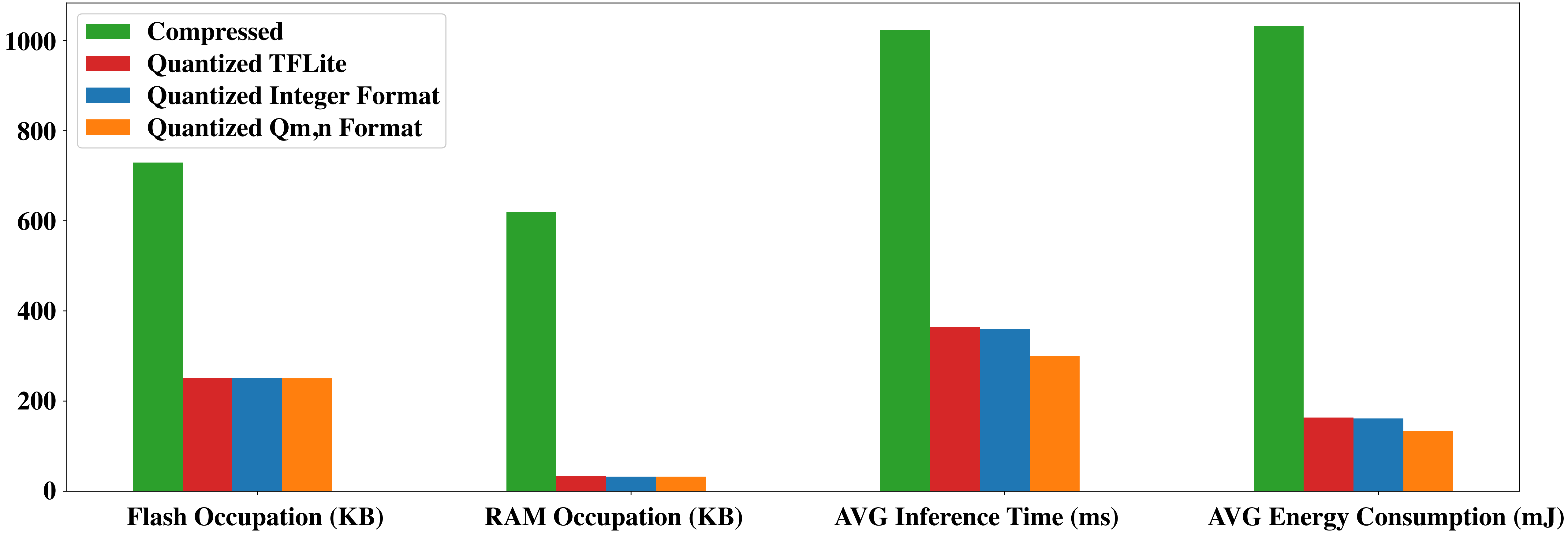


Results

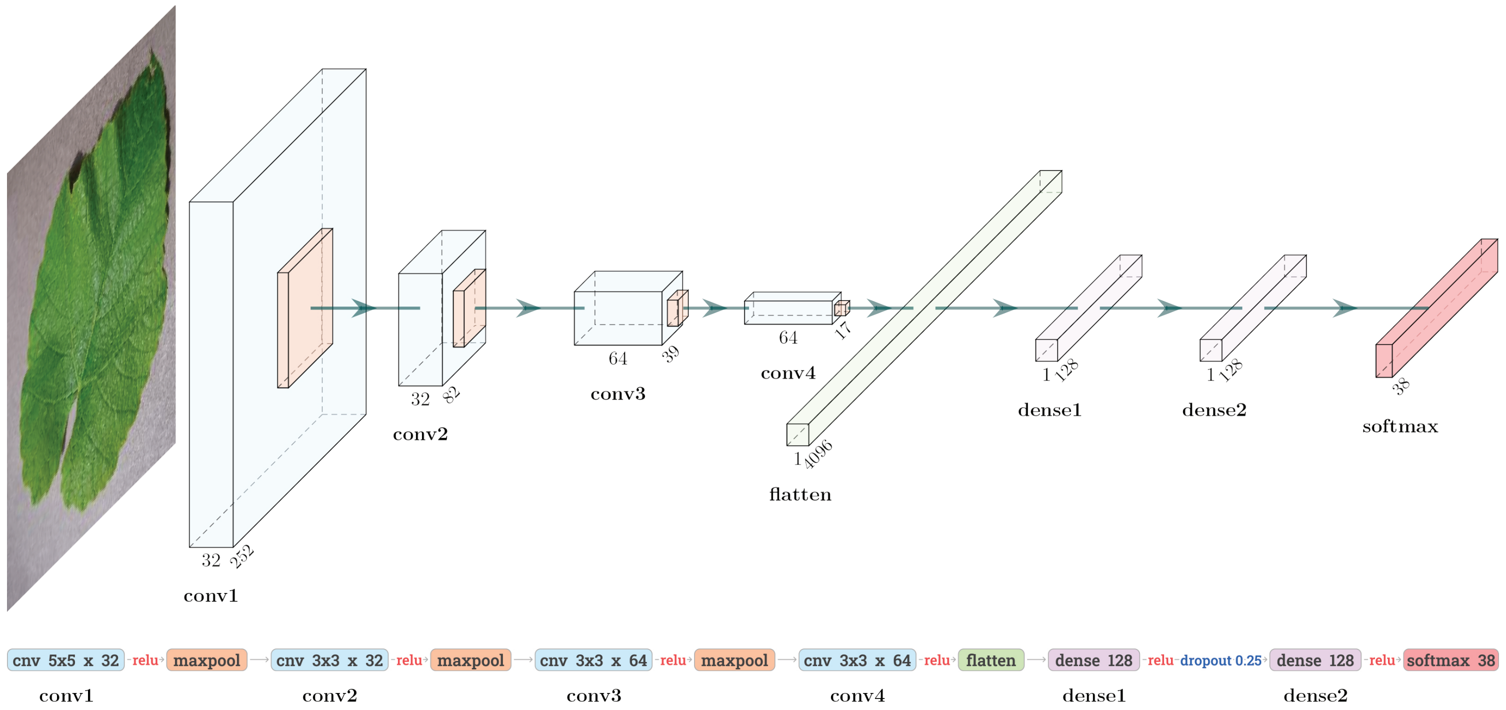
The Qm,n format quantized model resulted in the best trade-off in terms of accuracy, memory occupation, and energy consumption



Model	Flash (KB)	RAM (KB)	AVG Inf. Time (ms)	Accuracy	AVG Energy Cons. (mJ)	Cycles/MACC
Floating Point	-	-	-	0.96	-	-
Compressed	729.49	619.78	1, 022.81	0.96	1, 031.72	6.99
Quantized TFLite	251.79	33.25	364.44	0.96	163.15	2.50
Quantized Integer Format	251.79	32.60	360.41	0.96	161.34	2.47
Quantized Qm,n Format	250.44	32.35	299.60	0.95	134.12	2.05



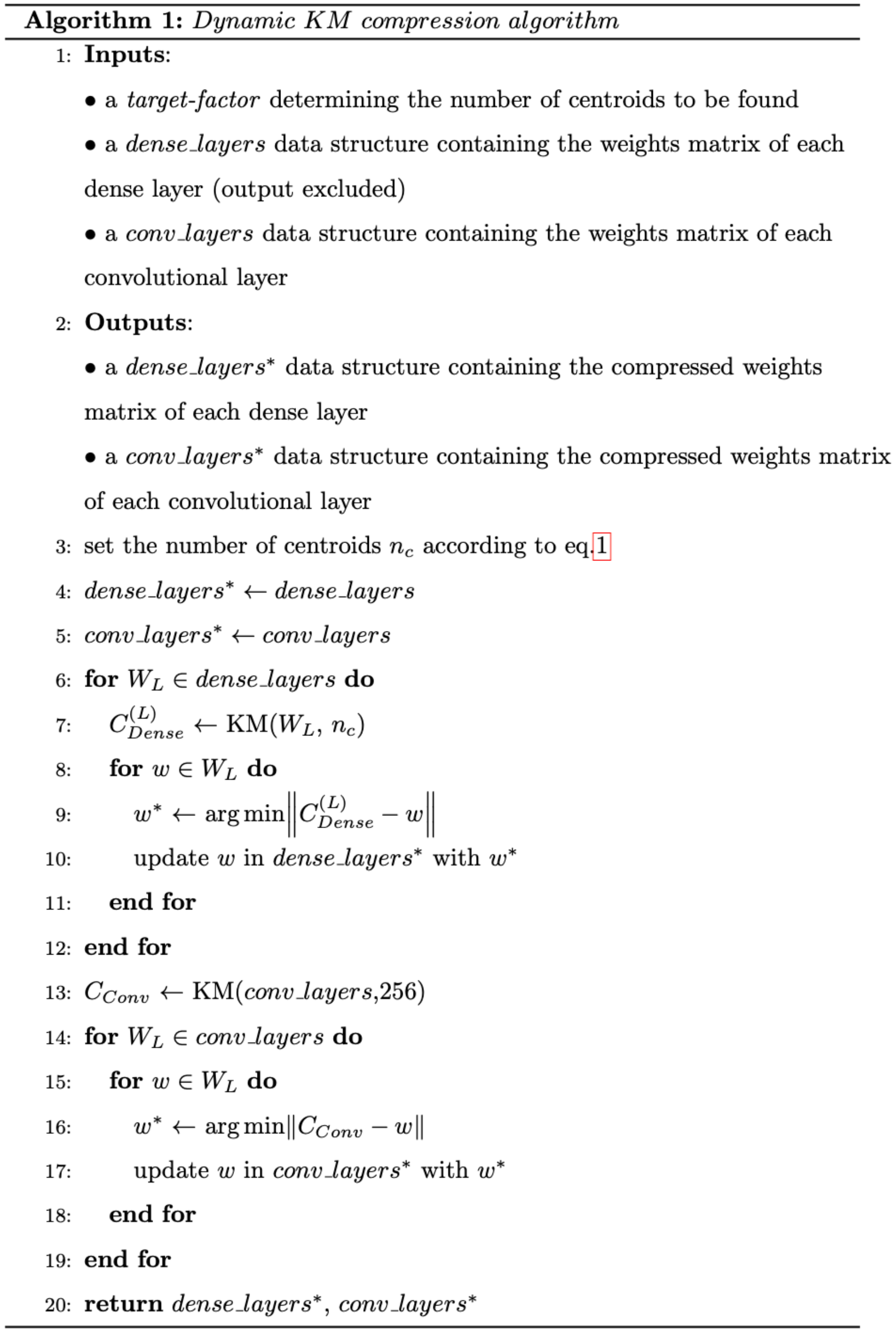
Porting other architectures on MCU



We are currently working on a new compression algorithm to port DNNs on MCU.

We introduce a "shared" compression of convolutional layers which are remapped into a fixed number of 256 K-Means centroids

F. De Vita, G. Nocera, D. Bruneo, V. Tomaselli, D. Giacalone, and S. K. Das, "Porting deep neural networks on the edge via dynamic k-means compression: A case study of plant disease detection," Pervasive and Mobile Computing, vol. 75, p. 101437, 2021, issn: 1574- 1192. doi: https://doi.org/10.1016/j.pmcj.2021.101437. [Online]. Available: https: //www.sciencedirect.com/science/article/pii/S1574119221000821.

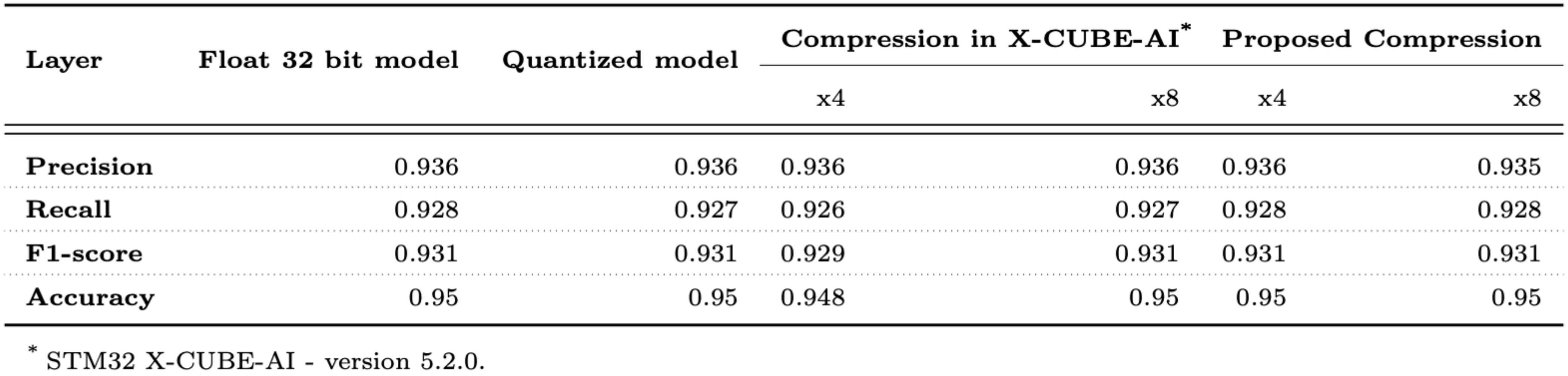


Algorithm 1: *Dynamic KM compression algorithm*

```
Inputs:
  • a target-factor determining the number of centroids to be found
  • a dense_layers data structure containing the weights matrix of each dense layer (output excluded)
  • a conv_layers data structure containing the weights matrix of each convolutional layer
Outputs:
  • a dense_layers* data structure containing the compressed weights matrix of each dense layer
  • a conv_layers* data structure containing the compressed weights matrix of each convolutional layer
set the number of centroids $n_c$ according to eq.1
$dense\_layers^* \leftarrow dense\_layers$
$conv\_layers^* \leftarrow conv\_layers$
for $W_L \in dense\_layers$ do
  $C^{(L)}_{Dense} \leftarrow \mathrm{KM}(W_L, n_c)$
  for $w \in W_L$ do
    $w^* \leftarrow \arg\min \left\| C^{(L)}_{Dense} - w \right\|$
    update $w$ in $dense\_layers^*$ with $w^*$
  end for
end for
$C_{Conv} \leftarrow \mathrm{KM}(conv\_layers, 256)$
for $W_L \in conv\_layers$ do
  for $w \in W_L$ do
    $w^* \leftarrow \arg\min \| C_{Conv} - w \|$
    update $w$ in $conv\_layers^*$ with $w^*$
  end for
end for
return $dense\_layers^*, conv\_layers^*$
```

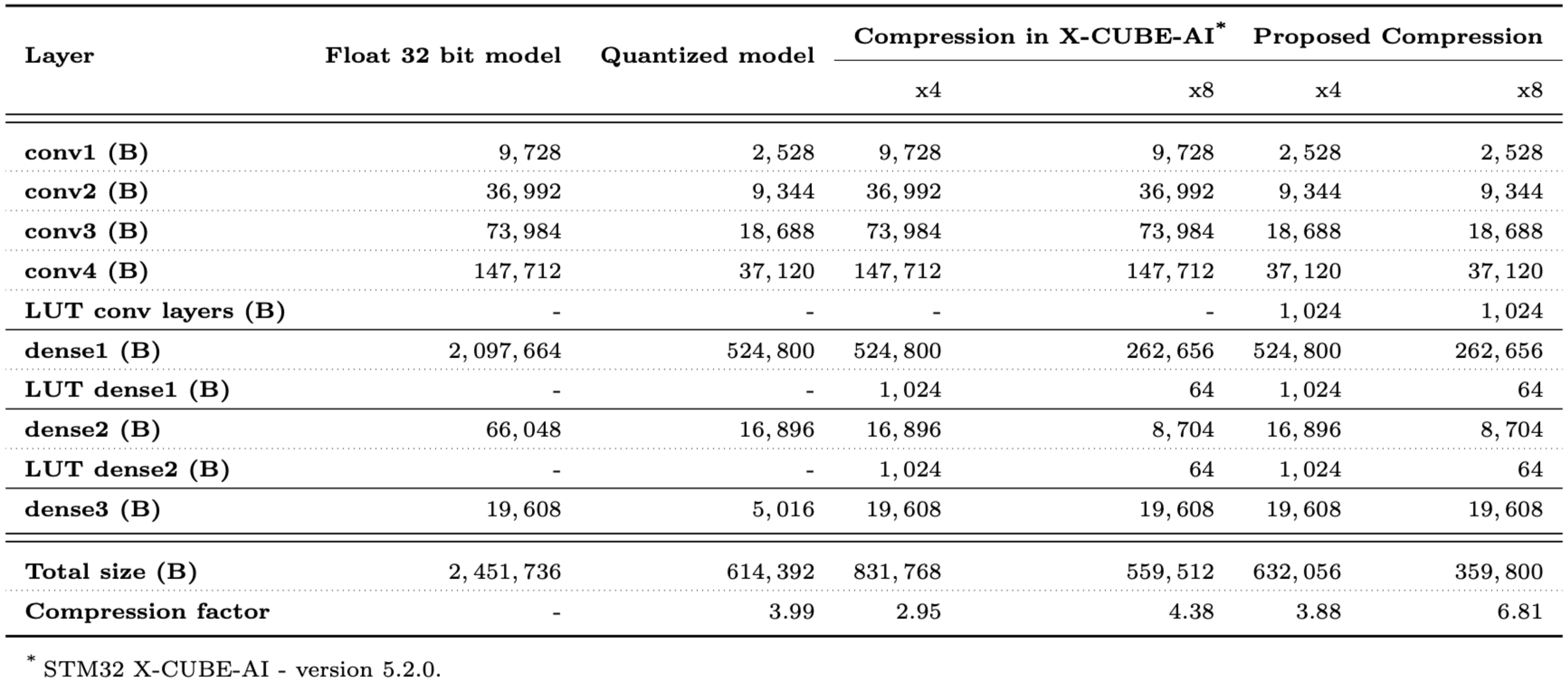
Porting other architectures on MCU - Results

Compared to other approaches such as integer quantization, compression in X-CUBE-AI*, the proposed technique results in the lowest memory footprint while maintaining the same performance of the original Float 32 bit model



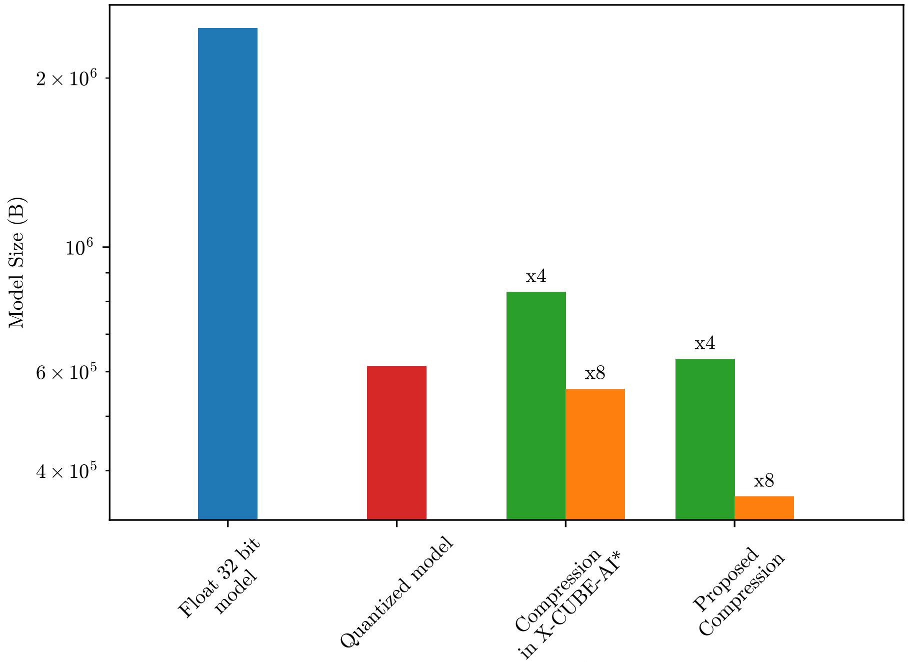
Layer	Float 32 bit model	Quantized model	Compression in X-CUBE-AI*		Proposed Compression	
			x4	x8	x4	x8
Precision	0.936	0.936	0.936	0.936	0.936	0.935
Recall	0.928	0.927	0.926	0.927	0.928	0.928
F1-score	0.931	0.931	0.929	0.931	0.931	0.931
Accuracy	0.95	0.95	0.948	0.95	0.95	0.95

* STM32 X-CUBE-AI - version 5.2.0.



Layer	Float 32 bit model	Quantized model	Compression in X-CUBE-AI*		Proposed Compression	
			x4	x8	x4	x8
conv1 (B)	9,728	2,528	9,728	9,728	2,528	2,528
conv2 (B)	36,992	9,344	36,992	36,992	9,344	9,344
conv3 (B)	73,984	18,688	73,984	73,984	18,688	18,688
conv4 (B)	147,712	37,120	147,712	147,712	37,120	37,120
LUT conv layers (B)	-	-	-	-	1,024	1,024
dense1 (B)	2,097,664	524,800	524,800	262,656	524,800	262,656
LUT dense1 (B)	-	-	1,024	64	1,024	64
dense2 (B)	66,048	16,896	16,896	8,704	16,896	8,704
LUT dense2 (B)	-	-	1,024	64	1,024	64
dense3 (B)	19,608	5,016	19,608	19,608	19,608	19,608
Total size (B)	2,451,736	614,392	831,768	559,512	632,056	359,800
Compression factor	-	3.99	2.95	4.38	3.88	6.81

* STM32 X-CUBE-AI - version 5.2.0.



* STM32 X-CUBE-AI - version 5.2.0.

Thank you !

Dario Bruneo

Department of Engineering

University of Messina (Italy)

✉ dbruneo@unime.it

✉ dario@smartme.io

Fabrizio De Vita

Department of Engineering

University of Messina (Italy)

✉ fdevita@unime.it