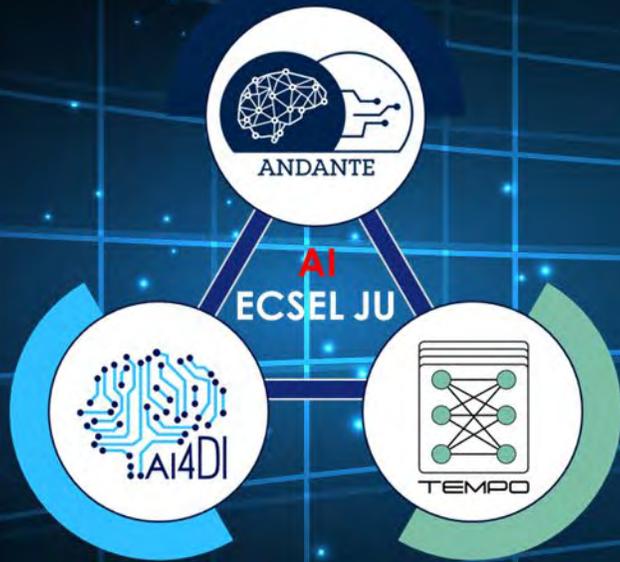
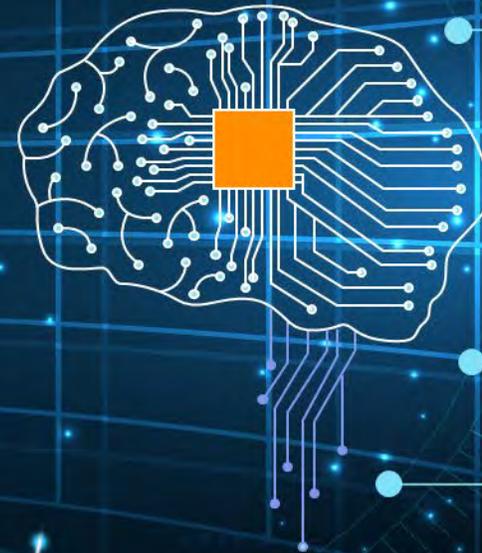


The International Workshop on Edge Artificial Intelligence for Industrial Applications (EAI4IA)

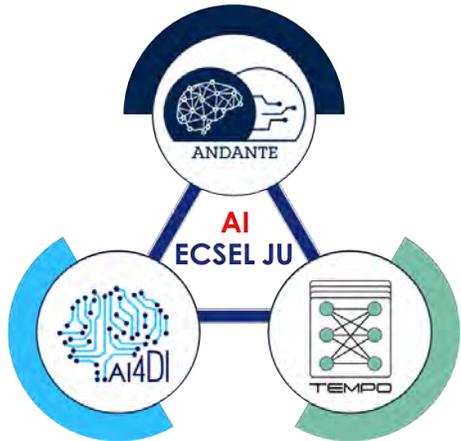


AI



Vienna, Austria
25-26 July 2022

The International Workshop on Edge Artificial Intelligence for Industrial Applications (EAI4IA)



Embedded Edge Intelligent Processing for End-To-End Predictive Maintenance in Industrial Applications

Ovidiu Vermesan, SINTEF, Norway

Vienna, Austria 25-26 July 2022

Presentation Outline



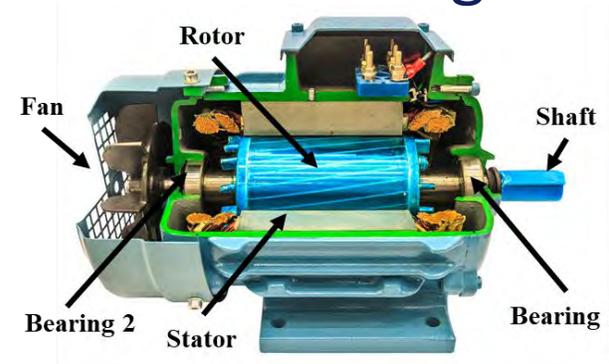
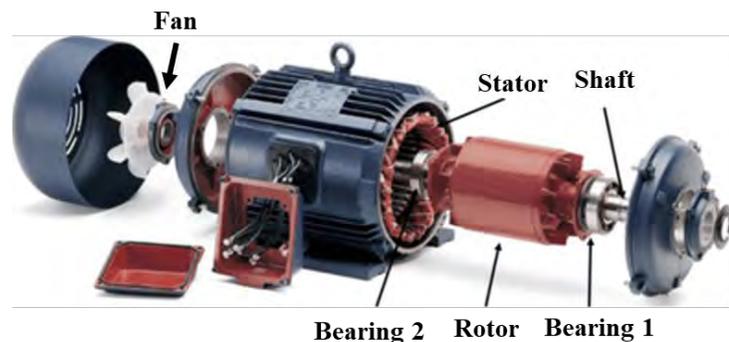
- Introduction
- Intelligent Edge Processing Real-time Maintenance Systems
- ML and DL for Embedded Edge Predictive Maintenance
- Approaches and Frameworks for Integrating AI Mechanisms within MCUs – Use Case Example
- Architecture of the Experimental Results
- Summary
- Discussions and Future Work

Introduction

- Apply AI-enabled methods to edge devices used for motors/equipment real-time conditions monitoring in industrial processes.
- Consider the edge processing continuum including the sensing, processing and communication devices (micro-edge) close to the physical industrial devices, the gateways, intelligent controllers processing devices (deep-edge), and the on-premise multi-use computing devices (meta-edge).
- Investigate different approaches to using ML and DL technologies to bring AI capabilities to micro-edge devices and apply these capabilities for classification to a PdM use case in industrial applications.
- Illustrate how to optimise ML and DL models for resource-constrained micro-edge-embedded devices using different end-to-end AI-based workflows.

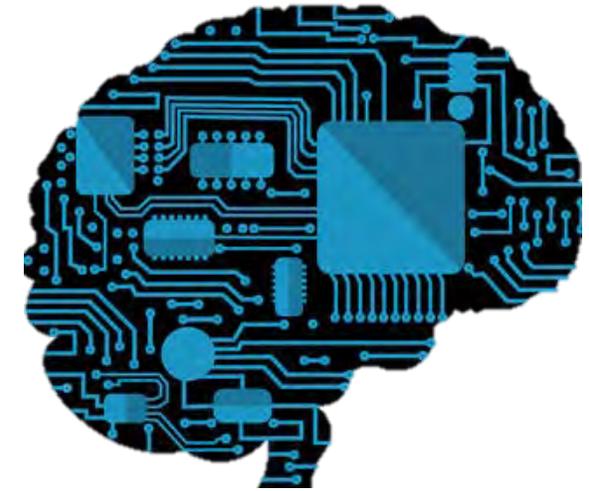
Intelligent Edge Processing Real-time Maintenance Systems

- PdM applies extensive set of input data and analysis to provide a more reliable indicator of the overall health and condition of the motors and an accurate prediction of possible failures and what actions to be considered to prevent it.
- Critical measurements motors: three-axis vibration, temperature, current, etc.
- Various components conditions and operations are possible causes that can generate anomalous behaviour, defining various normal/abnormal states (classes).
- Focus on AI-based PdM approaches, which learn from historical and real-time data employing ML and DL models implemented using micro-edge-embedded devices.



ML and DL for Embedded Edge Predictive Maintenance

- Three different workflows have been implemented to match the PdM application requirements for generating embedded code and performing learning and inference engine optimisations.
- Three existing frameworks and inference engines for integrating AI mechanisms within MCUs have been employed:
 - NanoEdge™ AI (NEAI) Studio,
 - Edge Impulse (EI) and
 - STM32 Cube.AI



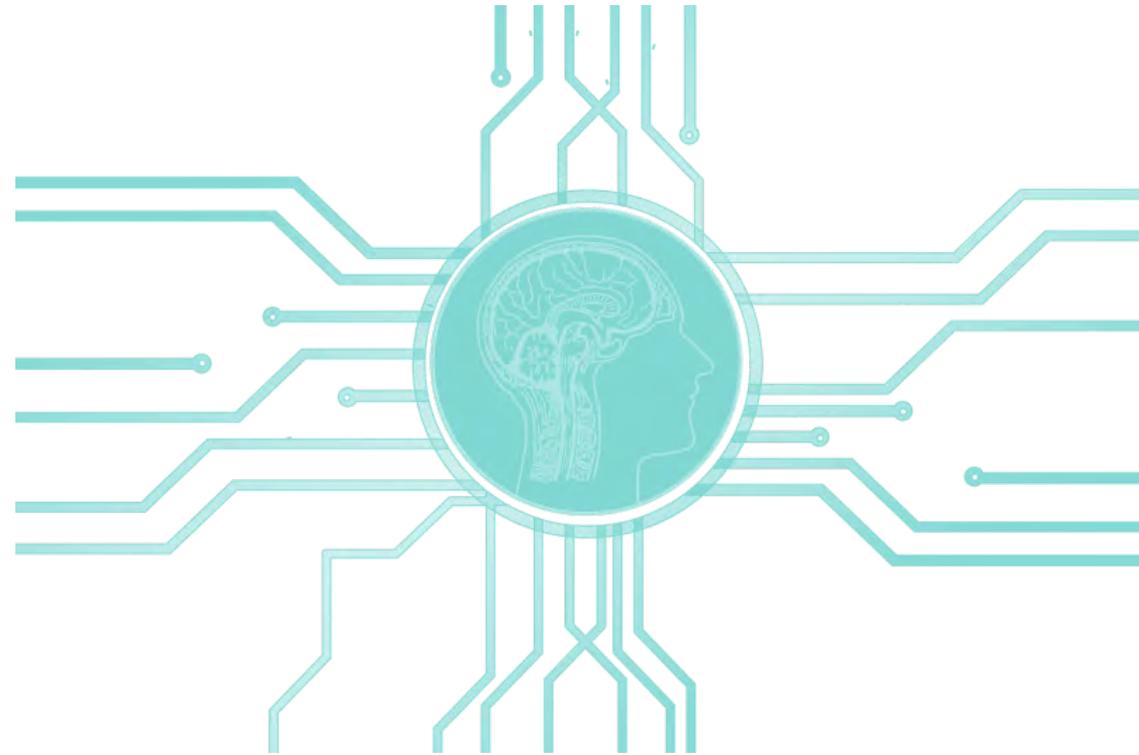
NANOEDGE AI
STUDIO 

 **EDGE IMPULSE**

STM32  
CubeIDE

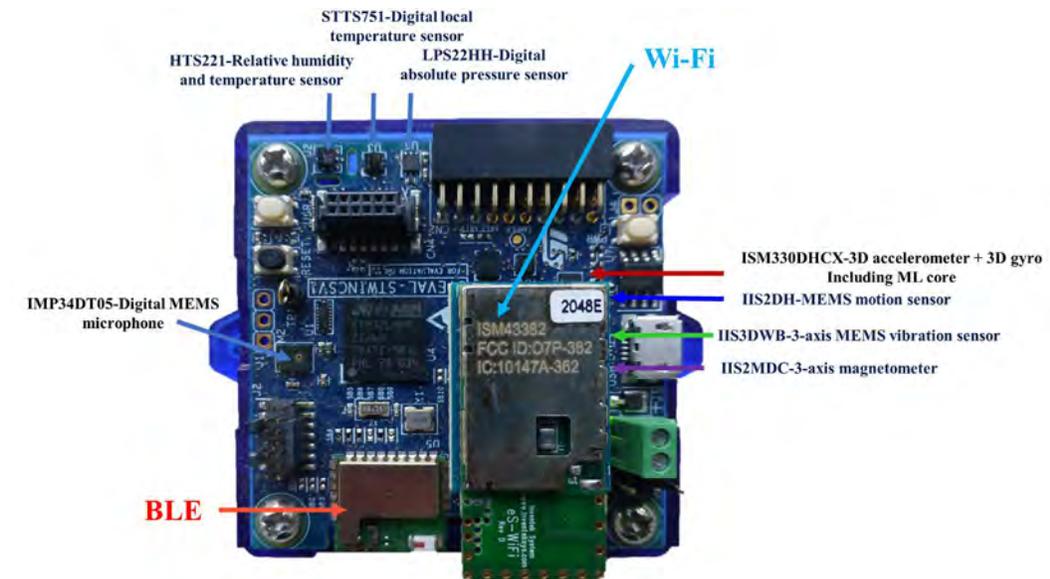
Approaches, Frameworks for Integrating AI Mechanisms

- The workflows offer various degrees of:
 - Automation of various parts of the E2E workflow,
 - Transparency into the ML/DL algorithms and model architecture, and
 - Control over model parameters and hyper parameters
- Therefore, difficult to compare performances. Rather, the aim has been to gain (and present) quantitative and qualitative insights in these complementary workflows with different design and learning components.



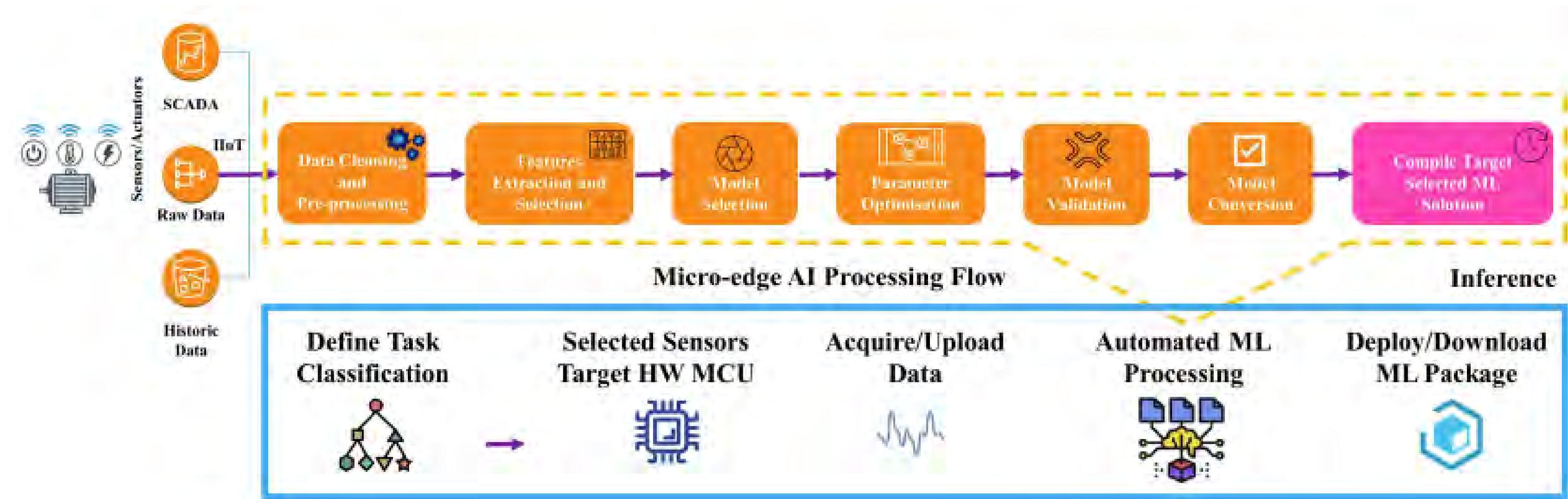
Architecture of the Experimental Results

- The micro-edge IIoT device used for the experiments comprises of:
- A three-axis ultrawide bandwidth (DC to 6 kHz) acceleration sensor (ISM330DHCX), with a 12-bit analog-to-digital converter, a user-configurable digital filter chain, a temperature sensor, and a serial peripheral interface.
- The micro electro mechanical systems (MEMS) vibration sensor has a selectable sensitivity (± 2 , ± 4 , ± 8 , or ± 16 g)
- Processing capabilities ensured by an Arm Cortex-M4 processor (120 MHz, 640 KB RAM, 2 MB Flash).
- The micro-edge device can be powered externally or by an internal lithium-ion battery
- BLE and Wi-Fi connectivity.



Approaches, Frameworks for Integrating AI Mechanisms

- Micro-edge AI processing flow



Classification Task - Design of the Use Case

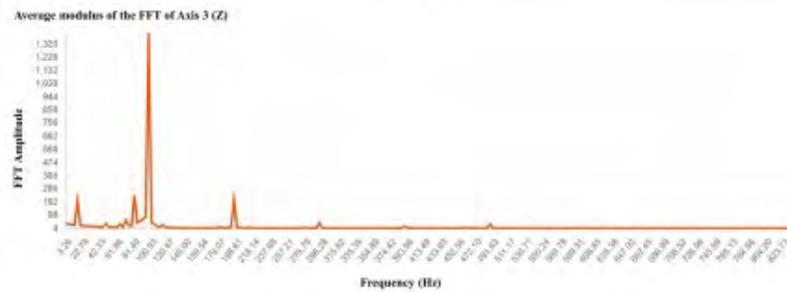
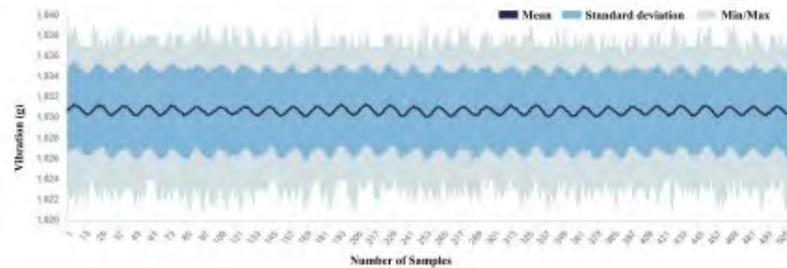
- Classification of the state of a motor based on vibration measurements
- A built-in three-axis accelerometer (ISM330DHCX) measures the accelerations of three orthogonal directions
- Classes defined based on conditions (motor speeds) and sub-conditions (malfunctions):
 - A. *MOTOR_OFF*: just record signals when nothing is happening
 - B. *MOTOR_ON_NORMAL_50*: the motor is running at 50% of the maximum speed
 - C. *MOTOR_ON_NORMAL_75*: the motor is running at 75% of the maximum speed
 - D. *MOTOR_ON_NORMAL_75_B*: the motor fan produces additional trepidations to the motor, while the motor is running at 75% of the maximum speed
 - E. *MOTOR_ON_NORMAL_MAX*: motor is running at maximum speed
- The use case was design with the following goals in mind:
 - The motor behaviour and the classification problem being solved with ML/DL were studied in-depth
 - Classes should be distinguishable for easier classification
 - Data sets should be class-balanced
 - Data sets should be properly split (training, validation, test)

Signal Data Acquisitions

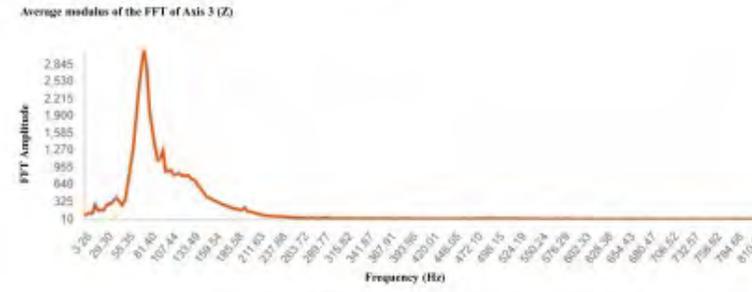
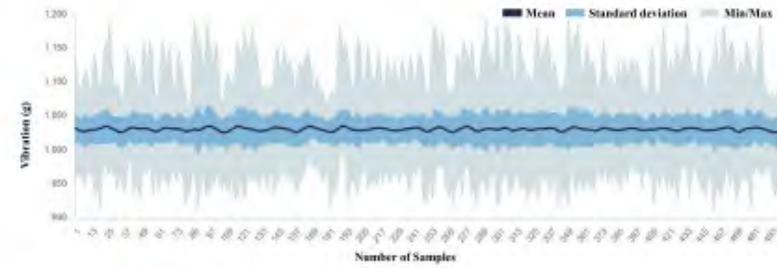
- AI platforms usually offer several ways within their GUIs to acquire signals, either from files or directly from the sensors (provided the IIoT device is supported).
- In the experimental use case, a simple logger application (in C) reads and logs the raw accelerometer sensor data directly on the serial port, so that logs can be retrieved from a computer using serial tools such as Tera Term, from the console of the IDE, and AI platform (NEAI).
- Acquisition parameters for each class:
 - Sampling frequency 1667 Hz,
 - Collection of signals (of approx. 30 seconds),
 - Buffer size of 512 samples on each axis, in total 1536 values per signal,
 - Each buffer is a snapshot of approximately 300 milliseconds ($= 512/1667$) of the accelerometer temporal vibration data

Signal Data Acquisitions

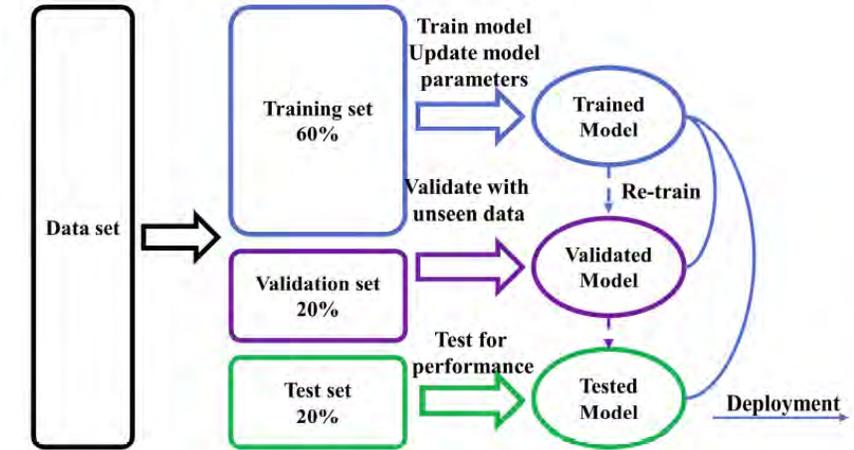
- Visualisation of two selected classes signals (75 and 75B) in both temporal and frequency domain with NEAI:



a) MOTOR_ON_NORMAL_75



b) MOTOR_ON_NORMAL_75_B



Data sets - the collection of signals for each class was split as shown in figure: (60% training, 20% validation and 20% test):

Exploring and visualising features

- Pre-classification phase:
 - Calculate and visualise feature importance, indicating how important the features are for each class
 - Employ dimension reduction algorithms to reduce the computing burden of ML algorithms (deleting the less important or redundant information from the data)
 - Root Mean Square (RMS) and peak values of vibration along the three-axis proved to be the most important features in determining the class (shown by EI and Python)
 - AI frameworks may select different features

Parameter	Equation	Description
Peak	$x_p = \max x_i $	Represents the maximum value of a signal
Mean	$\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i$	Represents the average value of a signal
Standard deviation	$\sigma_{std} = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2}$	Statistical metric defining the amount of variation in the signal, that is independent of DC bias. A low value indicates that the values tend to be more concentrated and closer to the mean, while a high value indicates that the values tend to spread out over a wider range
RMS	$x_{rms} = \sqrt{\frac{1}{N} \sum_{i=1}^N x_i^2}$	Root Mean Square (RMS) or the quadratic mean is the square root of the mean square (the arithmetic mean of the squares of a set of values). RMS can be calculated from the power spectral density of a signal (Seryasat et al., 2010)
Kurtosis	$x_{kur} = \frac{\frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^4}{\sigma_{std}^4}$	Quantifies the peak value of the probability density function. Statistical parameter that allows to analyze the distribution of the vibratory amplitudes contained in a time domain signal. It corresponds to the moment of fourth order norm. For a Gaussian distribution, its value is 3. Kurtosis indicates heavy-tailed or light-tailed datasets in comparison to a normal distribution. Datasets with high Kurtosis tend to have outliers (Droni et al., 2004)
Crest factor	$CF = \frac{x_p}{x_{rms}}$	Corresponds to the ratio between the crest value (maximum absolute value reached by the function representative of the signal during the considered period of time) and the RMS value (efficient value) of the signal (Droni et al., 2004)
Skewness	$x_{ske} = \frac{\frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^3}{\sigma_{std}^3}$	Quantifies the asymmetry behavior of a vibration signal through its probability density function. The skewness is a metric of symmetry in a dataset. A dataset is symmetric if it looks the same to the left and right of the center point. A normal distribution has a skewness of 0. Negative skew value illustrates that the distribution concentrates at the right. Positive skew value illustrates that the distribution concentrates at the left (Singh, 2021)

Where x_i is a digital signal series, $i = 1, 2, \dots, N$; N is the number of elements of the digital signal.

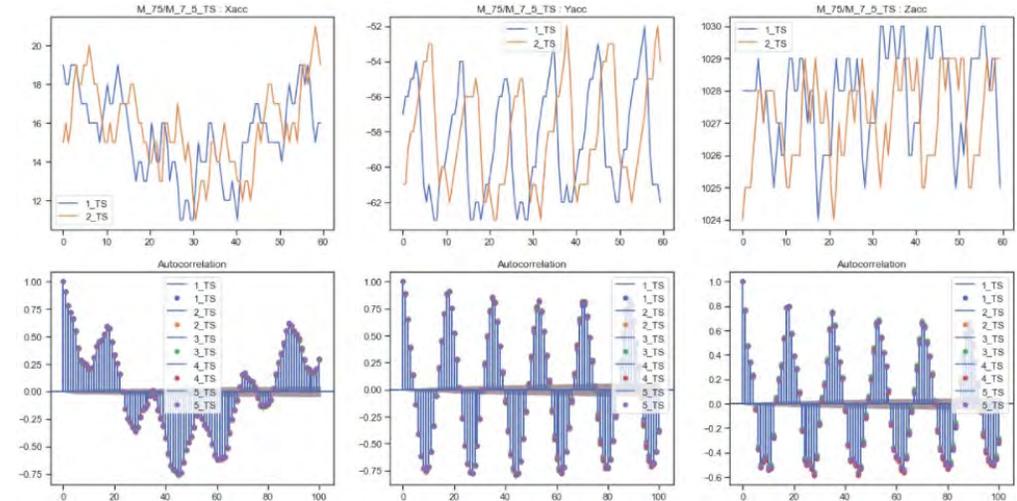
Additional feature:
Auto-correlation

$$r_{xx}[k] = \frac{1}{N} \sum_{n=1}^{N-k} x[n]x[n+k]$$

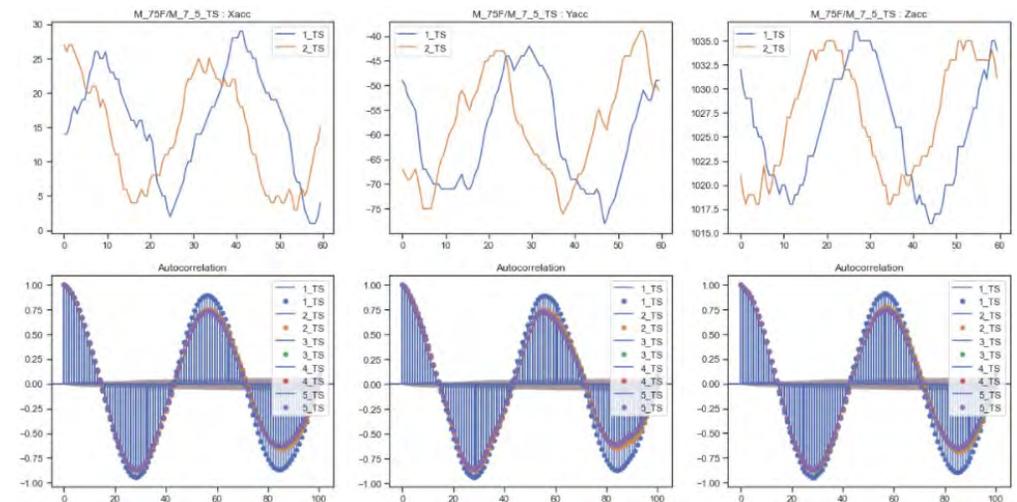
Auto-correlation

- Correlation (when two independent variables are linearly related) vs ACF (when a time series is linearly related to a lagged version of itself).
- From looking at the Xacc, Yacc, Zacc plots, it's not obviously apparent whether or not the data will have any auto-correlation.
- Auto-correlation (ACF) useful to:
 - Uncover patterns in data,
 - Select the best prediction model,
 - Evaluate the effectiveness of the model.

ACF for the M_75 class

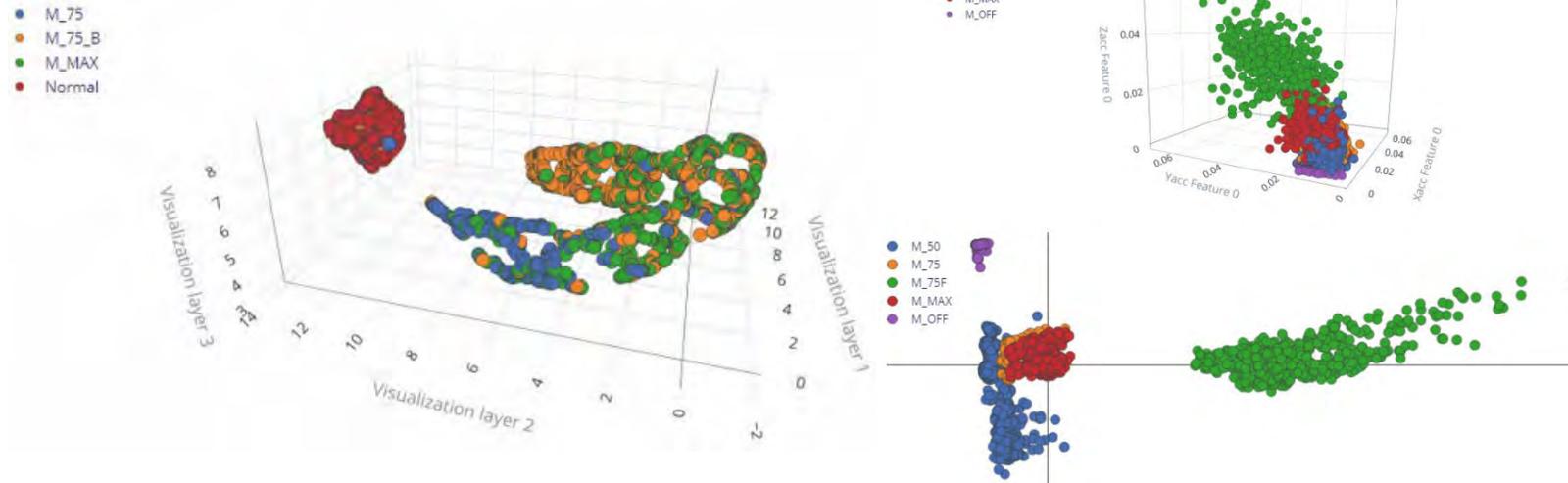


ACF for the M_75F class



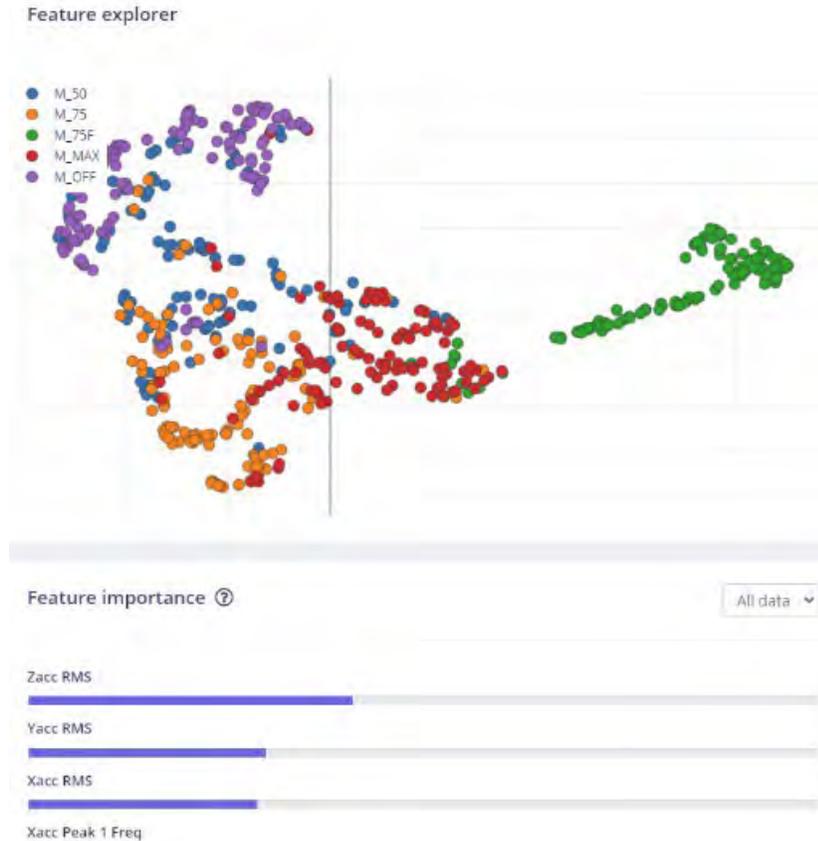
Feature Exploration and Visualisation (EI)

- 33 features (11 per each axis) are generated in total for the input signal.



Classes are not very distinguishable.

The features are visually clustered - good indication that the model can be trained to perform the classification.

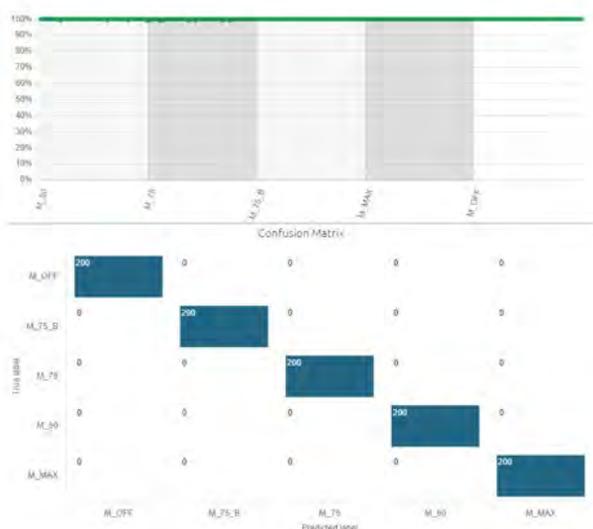


Displaying feature importance useful for dimension reduction.

Benchmarking

- Benchmarking allows to compare the performance of different model architectures within the same framework
- Confusion Matrix of the validation data – is a useful evaluation tool

Accuracy %	Confidence %	SMAAB	Flash MB	Model	Lib-selected
100.00	99.98	6.2	24.4	RF	
100.00	99.97	6.2	32.6	RF	
100.00	99.95	6.2	10.3	SEFR	
100.00	99.95	6.2	10.3	SVM	
100.00	99.94	12.4	22.7	SVM	
100.00	99.94	7.3	210.3	MLP	
100.00	99.92	12.4	22.5	SEFR	
100.00	99.92	12.4	22.4	SVM	
100.00	99.91	6.2	9.9	SEFR	
100.00	99.90	6.2	9.7	SVM	



Benchmarking with NEAI. Confusion matrix and data explorer. All correctly classified (green dots)

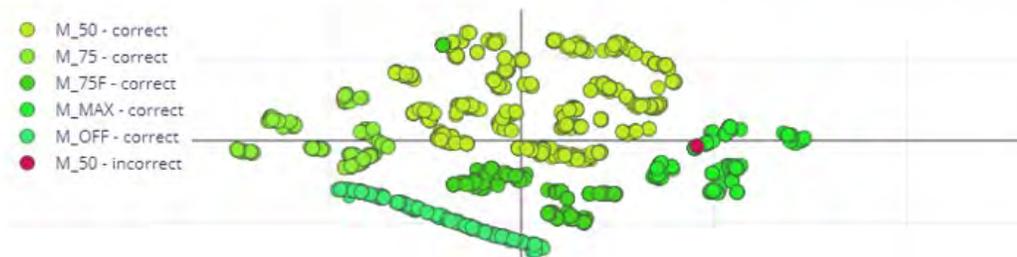
ACCURACY
100.0%

LOSS
0.00

Confusion matrix (validation set)

	M_50	M_75	M_75F	M_MAX	M_OFF
M_50	100%	0%	0%	0%	0%
M_75	0%	100%	0%	0%	0%
M_75F	0%	0%	100%	0%	0%
M_MAX	0%	0%	0%	100%	0%
M_OFF	0%	0%	0%	0%	100%
F1 SCORE	1.00	1.00	1.00	1.00	1.00

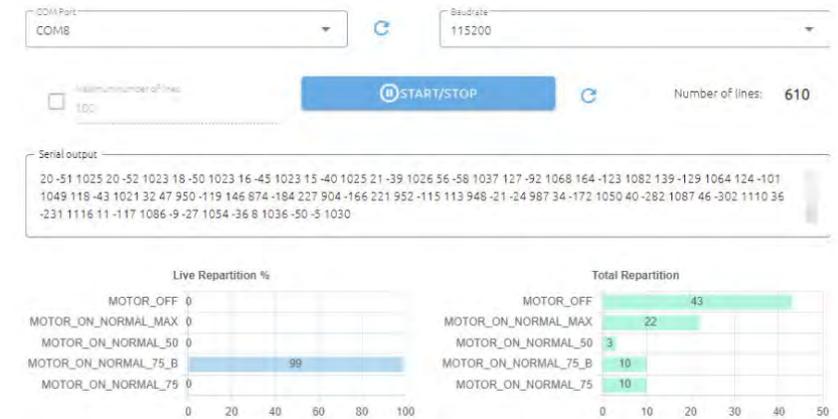
Data explorer (full training set) ?



Benchmarking with EI. Confusion matrix and data explorer based on training set: correctly classified (green dots) and misclassified (red dots).

Testing

- Evaluation of the trained model on the testing dataset to analyse how well the model performs against unseen data.
- Both NEAI and EI platforms provide a microcontroller emulator to test and debug the generated model prior to its deployment on the device
- An advantage of live streaming during testing is that it ensures unbiased evaluation of model effectiveness (completely new signals, not seen before)
- The confusion matrix in the images show that the classifier manages to properly reproduce and detect all classes with reasonable certainty percentages, and these are comparable (NEAI vs EI).



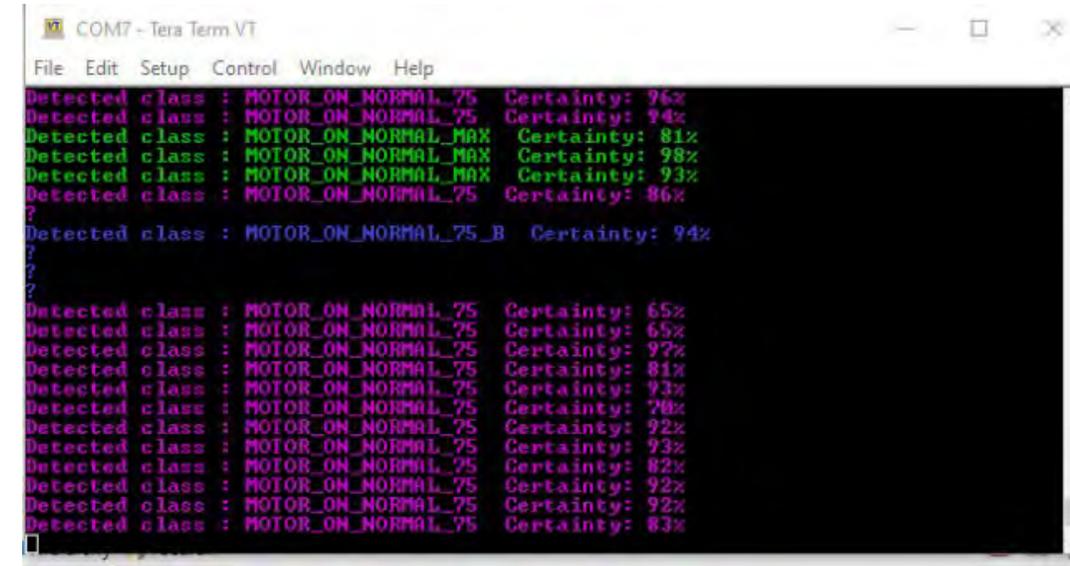
Evaluation of trained model using NEAI Emulator with live streaming (with Arm® Cortex®-M4 MCU STM32L4R9)



EI model testing with test datasets (Arm® Cortex®-M4 MCU STM32L4R9 not yet supported)

Deployment and Inference

- Model deployment is dependent on the hardware/software platform and in essence comprises three steps:
 1. Format conversion of the fully trained model,
 2. Weight/model compression to reduce the amount of memory to store the weights in the target hardware platform,
 3. Compiling the model and generating the code to be integrated with the MCUs firmware
- The back-end flow - wrap an STM32CubeIDE project with the generated files from the deployed models, adding functionality on top such as retrieving the accelerometer values to be fed to the classification function and displaying the result, then compile, build, and flash onto the MCU target.
- Inference classification - running directly from the target hardware platform on the micro-edge IIoT devices, producing classification in real-time



```
COM7 - Tera Term VT
File Edit Setup Control Window Help
Detected class : MOTOR_ON_NORMAL_75 Certainty: 96%
Detected class : MOTOR_ON_NORMAL_75 Certainty: 94%
Detected class : MOTOR_ON_NORMAL_MAX Certainty: 81%
Detected class : MOTOR_ON_NORMAL_MAX Certainty: 98%
Detected class : MOTOR_ON_NORMAL_MAX Certainty: 93%
Detected class : MOTOR_ON_NORMAL_75 Certainty: 86%
?
Detected class : MOTOR_ON_NORMAL_75_B Certainty: 94%
?
?
Detected class : MOTOR_ON_NORMAL_75 Certainty: 65%
Detected class : MOTOR_ON_NORMAL_75 Certainty: 65%
Detected class : MOTOR_ON_NORMAL_75 Certainty: 97%
Detected class : MOTOR_ON_NORMAL_75 Certainty: 81%
Detected class : MOTOR_ON_NORMAL_75 Certainty: 93%
Detected class : MOTOR_ON_NORMAL_75 Certainty: 70%
Detected class : MOTOR_ON_NORMAL_75 Certainty: 92%
Detected class : MOTOR_ON_NORMAL_75 Certainty: 93%
Detected class : MOTOR_ON_NORMAL_75 Certainty: 82%
Detected class : MOTOR_ON_NORMAL_75 Certainty: 92%
Detected class : MOTOR_ON_NORMAL_75 Certainty: 92%
Detected class : MOTOR_ON_NORMAL_75 Certainty: 83%
```

Example of live classification streaming with detected state and confidence. Tera Term interacting with STWIN IIoT device (Arm® Cortex®-M4 MCU STM32L4R9)

Summary

- Embedding trained models into the firmware code enables AI/ML capabilities of intelligent edge devices.
- Three different workflows have been implemented to match the PdM application requirements for generating embedded code and performing learning and inference engine optimisations using:
 - NanoEdge™ AI Studio
 - Edge Impulse
 - STM32 Cube.AI
- Various scenarios have been explored allowing to evaluate trade-offs between computational cost and performance on actual classification task (state of a motor based on vibration measurements measured by a built-in three-axis accelerometer).
- The results have been used to lay down the foundation of the PdM strategy

Discussions and Future Work

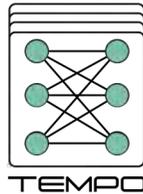
- Findings

- ML and NNs can be efficiently deployed on resource-constrained devices, which enable cost-efficient deployment, widespread availability, and the preservation of sensitive data in PdM applications.
- However, the trade-offs associated with optimisation methods, software frameworks and hardware architecture on performance metrics, such as inference latency and energy consumption, are yet to be studied and researched in depth.

- Future work

- Investigate more complex PdM systems using various AI-based techniques.
- Enlarge comparison and benchmarking by considering more edge ML and DL technologies, workflows, and datasets.
- Aim toward a more generic and complete PdM strategy by including insights from other applications, such as anomaly detection, regression, and forecasting.

Event Organisers



The Key Digital Technologies Joint Undertaking - the Public-Private Partnership for research, development and innovation – funds projects for assuring world-class expertise in these key enabling technologies, essential for Europe's competitive leadership in the era of the digital economy. KDT JU is the successor to the ECSEL JU programme. www.kdt-ju.europa.eu

The AI4DI project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and the national authorities. www.ai4di.eu

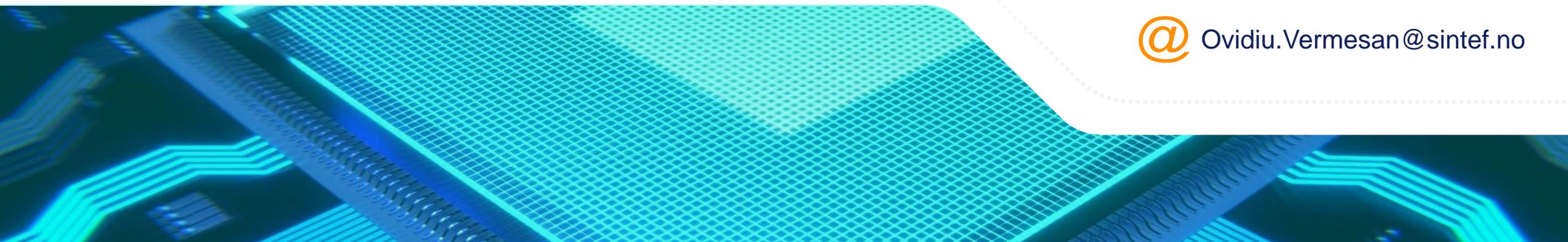
The TEMPO project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826655. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Belgium, France, Germany, The Netherlands, Switzerland. www.tempo-ecsel.eu

The ANDANTE project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 876925. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Belgium, France, Germany, The Netherlands, Portugal, Spain, Switzerland. www.andante-ai.eu



Thank You

For your attention



@ Ovidiu.Vermesan@sintef.no