UNIVERSITÄT
PADERBORN

Embedded Machine Learning (EML)

1. Introduction

Andre Diekwisch
Christoph Berganski
Prof. Dr. Marco Platzner

Computer Engineering Group

Overview

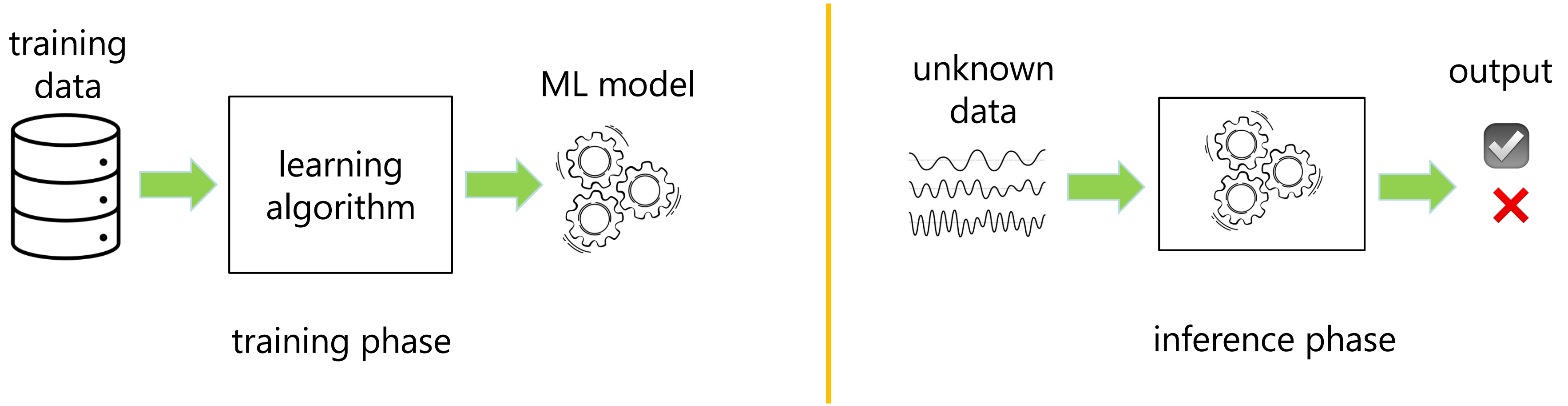
- What is Embedded Machine Learning?
- Course Overview
- Course Organization

What is Machine Learning?

- Machine learning (ML) is a subset of artificial intelligence (AI) that enables systems to learn from data, identify patterns, and make decisions or predictions without explicit programming.

- Uses of ML methods for building “intelligent systems”
  - classification: assign input data to one out of known classes
  - regression: predict output value based on given input data
  - clustering: group input data based on inherent characteristics

- Mostly, there is a training phase and an inference phase



What is an Embedded System (ES)?

- ES = information processing system embedded into a larger product
  - implements a (a few) specific application(s)
  - includes sensors, actors, computing, and communication
  - computing comprises software and hardware
  - wide range of design goals: minimize cost at some performance goal, meet timing requirements, minimize power and energy consumption, guarantee dependable operation, …

smart radar sensor

industrial control

cardiac defibrillator

thermostat

hearing aid

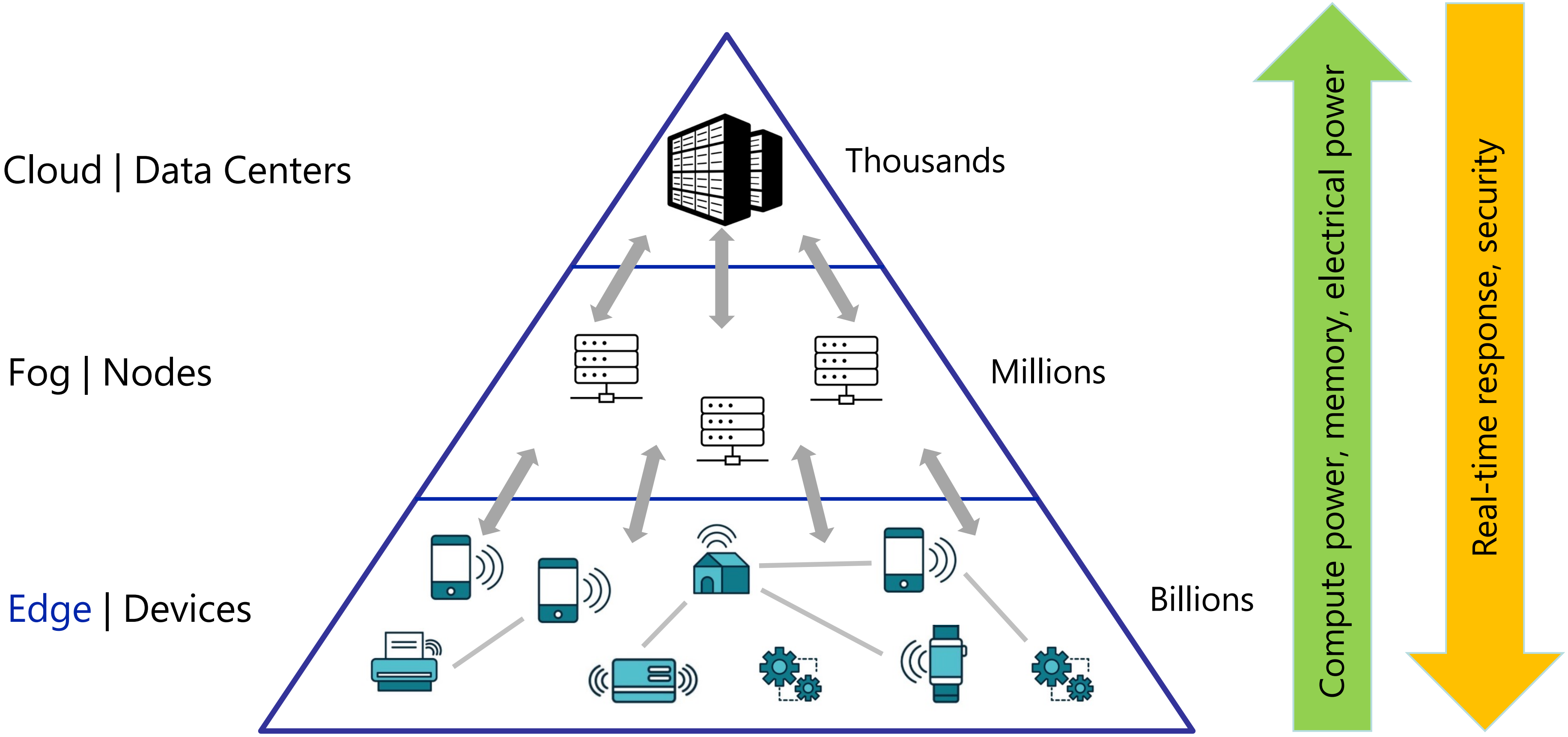
digital tv set

automotive functions

production robot

What is Embedded Machine Learning (EML)?

- EML = Machine Learning @ Edge



Cloud | Data Centers

Fog | Nodes

Edge | Devices

Thousands

Millions

Billions

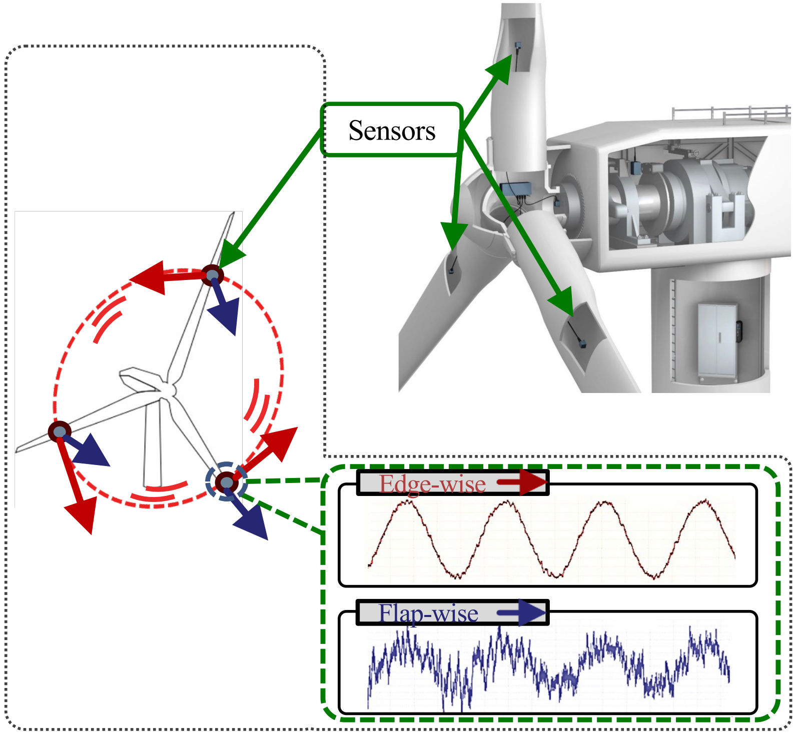
Compute power, memory, electrical power

Real-time response, security

Internet-of-Things (IoT) =
Embedded + connected to Internet

EML Example #1

- Wind turbine condition monitoring
  - harsh working conditions: huge mechanical forces, extreme temperatures, lightning strikes
  - condition monitoring is essential for reliable electricity generation and safe operation



- BLADEcontrol by **Weidmüller**
  - existing product for predictive maintenance, ice detection, operation improvement

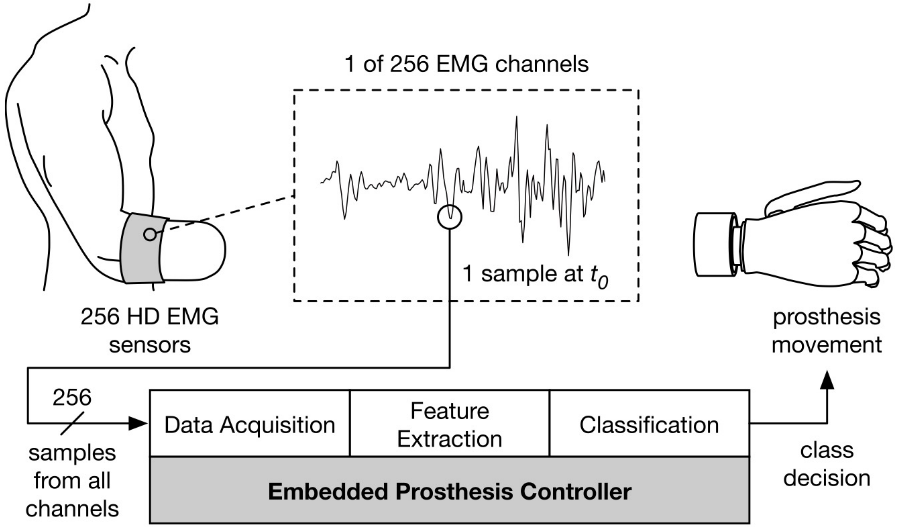
  - apply ML (multi-channel CNN) to identify faulty sensor patterns

[H.G. Mohammadi et al., *DeepWind: An Accurate Wind Turbine Condition Monitoring Framework via Deep Learning on Embedded Platforms*. 25th IEEE Intl. Conf. on Emerging Technologies and Factory Automation (ETFA). 2020]

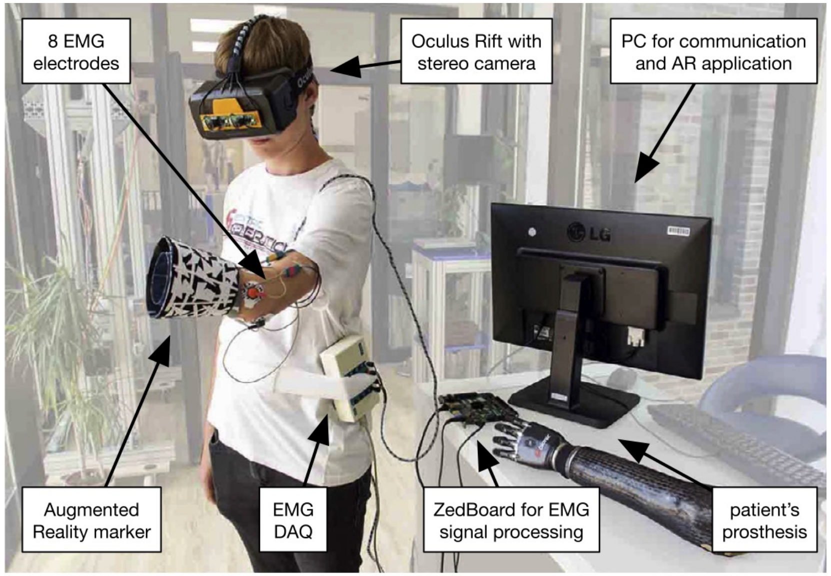
EML Example #2

- Myoelectric signal classification from EMG sensors
  - high-density EMG sensors (up to 256 channels) used for prosthesis control
  - amputees need an environment to train multiple movements (up to 8)

- Apply ML to detect movements
  - time domain features: mean absolute value (MAV), length of the waveform (WFL), amount of zero-crossings (ZC), amount of slope-sign-changes (SSC)
  - classifier: linear discriminant analysis (LDA)



[A. Boschmann, A. Agne, G. Thombansen, L. Witschen, F. Kraus, M. Platzner, *Zynq-based Acceleration of Robust High Density Myoelectric Signal Processing*, J. Parallel Distrib. Comput. 123 (2019) 77–89]



8 EMG electrodes
Oculus Rift with stereo camera
PC for communication and AR application
Augmented Reality marker
EMG DAQ
ZedBoard for EMG signal processing
patient's prosthesis

training setup (augmented reality) with 8 channels

EML Example #3

- Elephant Edge project
  - open-source elephant tracking collar that helps park rangers reduce animal loss from illegal ivory poaching, trophy hunting, human conflict, and environmental degradation
  - use of ML for
    - poaching risk monitoring: sense positions and notify rangers
    - human conflict monitoring: sense mobile phones and WiFi
    - elephant musth monitoring: sense motion and acoustic signals
    - activity monitoring: detect when an elephant is eating, drinking, sleeping
    - elephant communication monitoring: record sounds

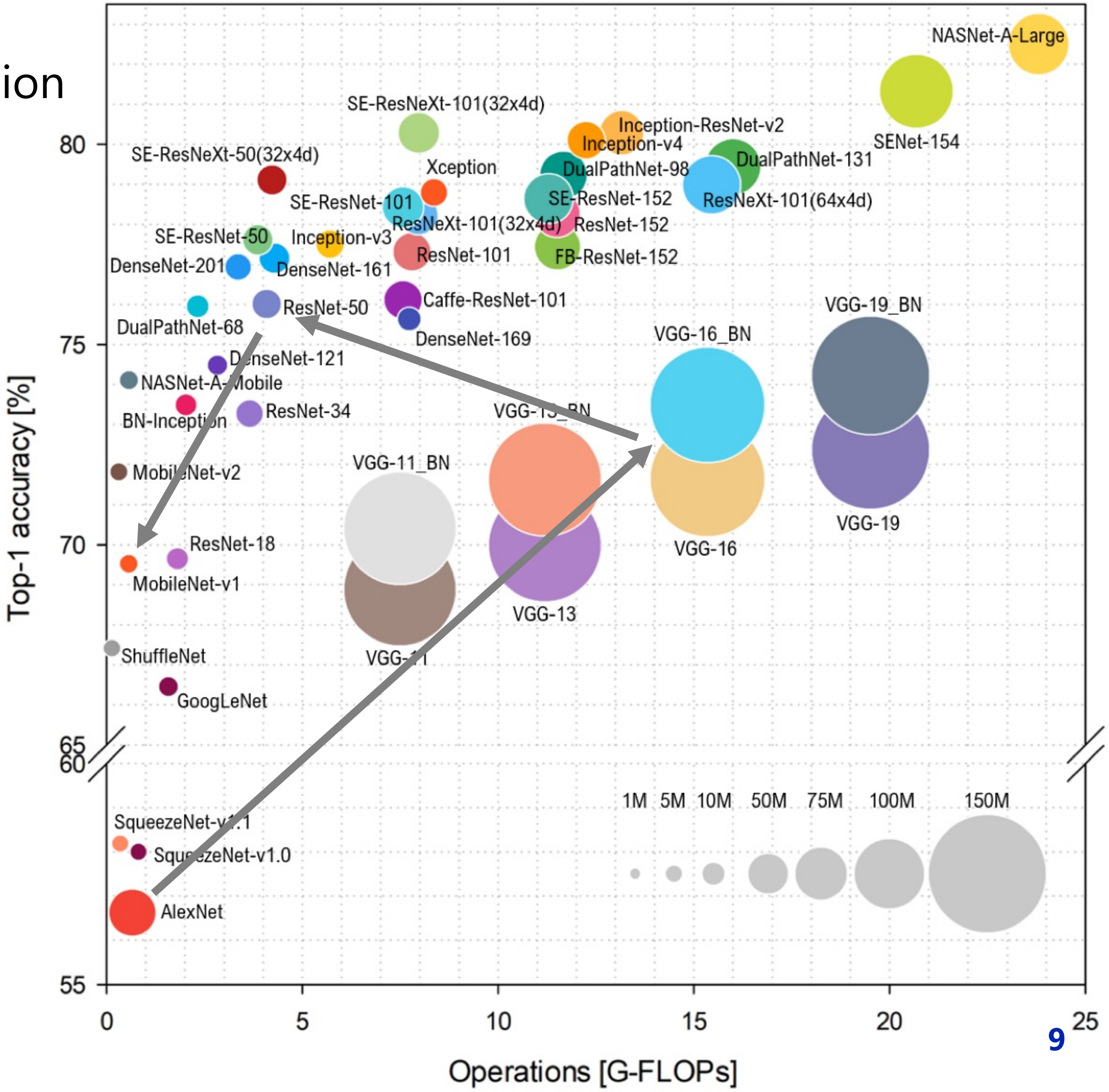
OPENCOLLAR

[https://www.hackster.io/contests/ElephantEdge]

Challenges for EML

- Many ML methods, especially deep neural networks, need vast amounts of compute power, memory, and energy

- Example: image classification deep neural networks
  - AlexNet (2012)
    - ~57% accuracy
    - ~60M parameters
  - VGG net (2014) VGG-16
    - ~72% accuracy
    - ~138M parameters
  - ResNet (2015) ResNet-50
    - ~76% accuracy
    - ~26M parameters
  - MobileNet (2015) v1
    - ~71% accuracy
    - ~4M parameters



[S. Bianco, R. Cadene, L. Celona, P. Napoletano, *Benchmark Analysis of Representative Deep Neural Network Architectures*, arXiv:1810.00736, 2018]

Challenges for EML

- Large language models (LLMs) are currently the most complex DNNs
  - trained to understand, predict, and generate textual data
  - mostly based on transformer architectures

- Example: evolution of parameters of the GPT LLM family

Year	Model	Parameters
2018	GPT-1	120M
2019	GPT-2	1.5B
2020	GPT-3	175B
2023	GPT-4	1.76T
2025	GPT-5	5-12T (estimated)

[S. Minaee et al., *Large Language Models: A Survey*, arXiv:2402.06196v3, 2025]

Challenges for EML

- Embedded systems are rather constrained in compute power, memory, and available energy

- How can we implement ML on embedded systems?

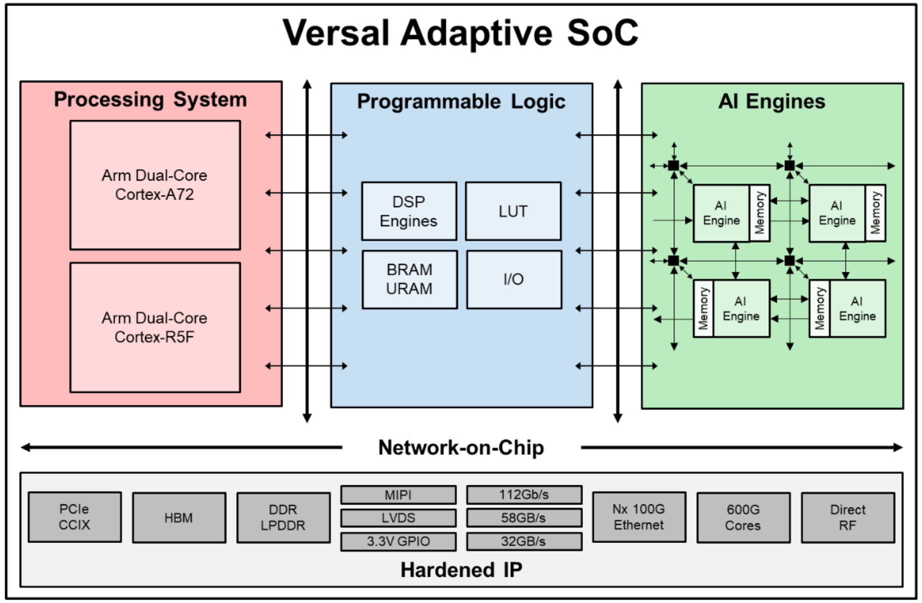
1. Run only inference on the ES, training on more powerful systems

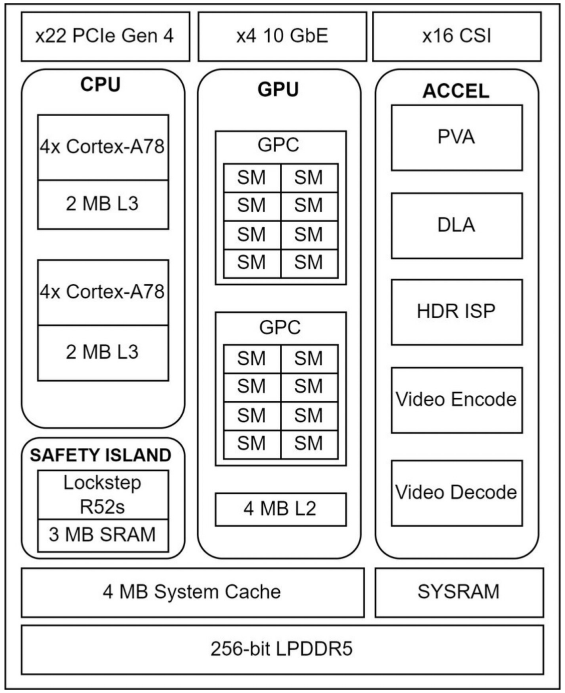
2. Focus on smaller models, that are still useful for embedded applications

3. Make models smaller to fit on the ES – model compression

Compute Elements

- Huge differences in compute power, cost, energy requirements
  - general-purpose microprocessor
  - microcontroller (μC)
  - graphics processing unit (GPU)
  - domain-specific architectures, e.g. tensor processing unit (TPU)
  - field-programmable gate array (FPGA)
- Often, compute element(s) + memory + peripherals (I/O) are integrated on a system-on-chip





NVIDA Jetson embedded GPU

Some Terminology

- Tiny-ML: ML on low-cost, low-power microcontrollers
- Edge-AI: ML on more powerful systems, but still at the edge
- Sometimes: Tiny-ML ⊂ EML ⊂ Edge-AI
  - although often used synonymously

	Edge-AI	EML	Tiny-ML
Scope	Broad (AI @ edge)	ML on embedded system	Tiny ML models
Compute system	Gateways, smart phones, smart cameras	microcontroller or others	low power microcontroller
Memory	MiB to GiB	MiB	KiB
Power consumption	Moderate to high	Low	<1 mW
Application	Autonomous driving, Real-time video analysis	Audio classification, surveillance	Anomaly detection

Learning Objectives

- Lecture
  - describe the structure and challenges of embedded machine learning systems
  - explain how traditional machine learning methods and deep neural networks work
  - select and apply machine learning methods for given use cases
  - select and apply hardware environments and software tools
  - explain methods for model compression
  - assess the significance of parameters such as model accuracy, latency, throughput, memory requirement
- Exercises
  - work on paper & pencil exercises to deepen understanding of the lecture material
- Lab
  - introduction to EML software (libraries and frameworks) and hardware (embedded compute platform)

Responsible AI

- Responsible AI is ethical, fair, transparent, safe, and legally compliant
  - ethical
    - establish clear responsibility for the outcomes of AI systems
    - protect user data and adhere to regulations
    - maintain human control and supervision for AI-based decisions
  - fair
    - ensure AI treats all users fairly and does not create or emphasize biases
  - transparent
    - AI should be understandable / explainable
    - users should know when they interact with AI
    - users should know how AI makes decisions
  - safe
    - AI should not pose risks to human life or well-being

- Human-centered AI
  - AI should be used to augment and enhance humans' lives

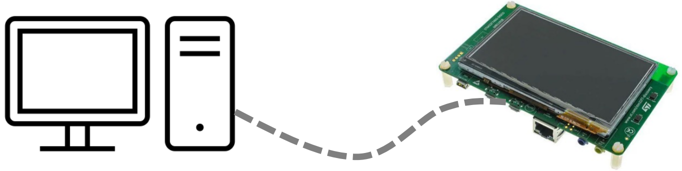
[Ben Shneiderman, Human-Centered AI, Oxford University Press, 2022]

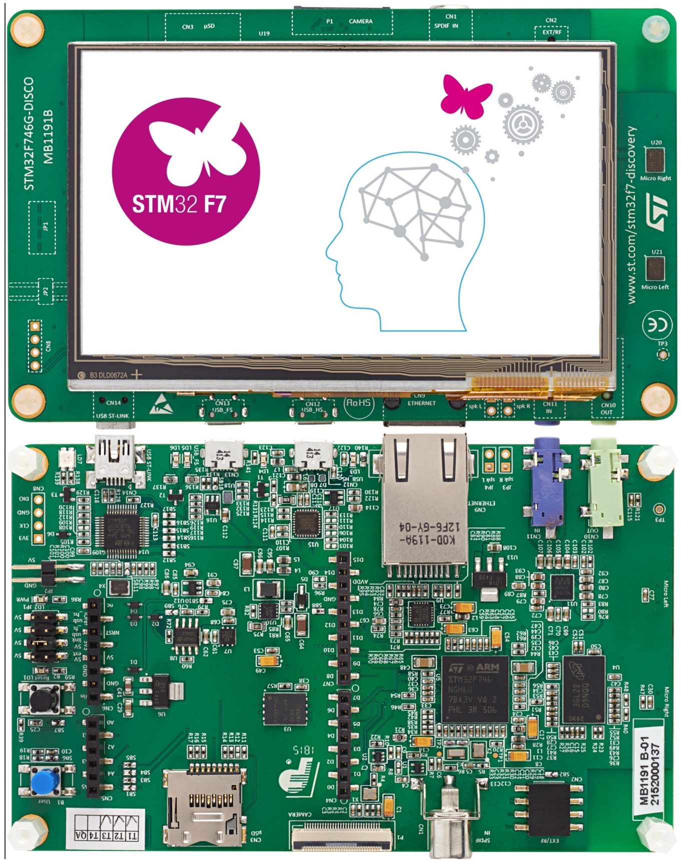
Lecture – Topics

- Introduction
- Sensors, data acquisition, and signal preprocessing
- Traditional machine learning methods
- Deep neural networks
- Model compression
- Processors and hardware accelerators

Lab – Topics (1)

- Development platforms
  - PC connected to µC board
  - 32F746GDISCOVERY board
    - Arm Cortex-M7 processor @ 216MHz
    - 340 KiB RAM, 1 MiB flash, 64 MiBit SDRAM
  - sensors
    - stereo microphones (on board)
    - inertial measurement unit (IMU)
    - camera module

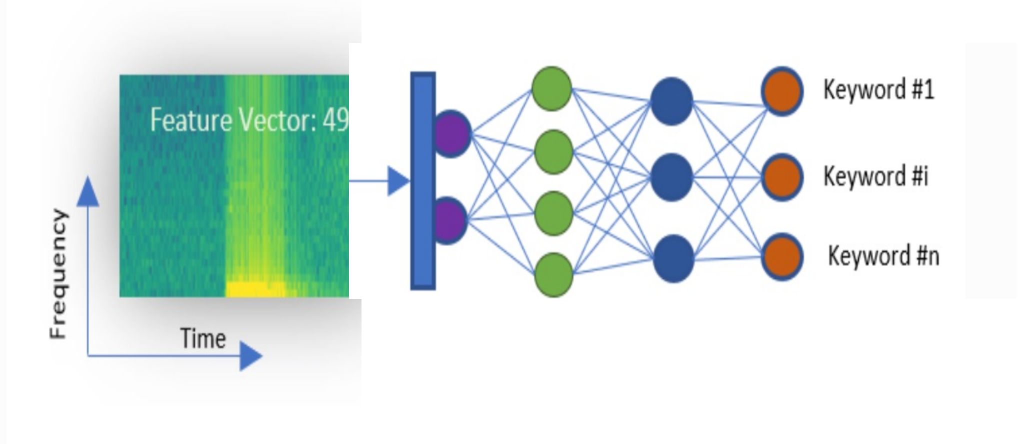




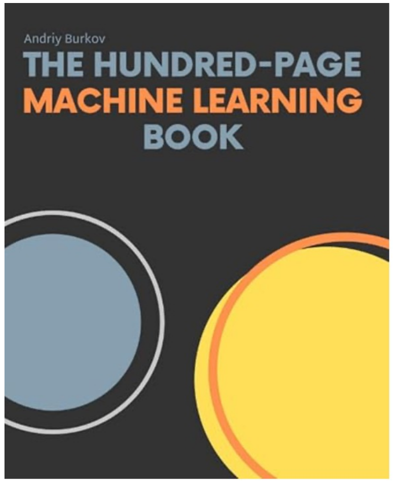
- Tools on PC
  - VS code
  - Python
  - scikit-learn, TensorFlow
  - platformIO
- Tools on µC
  - C/C++
  - TensorFlow runtime

Lab – Topics (2)

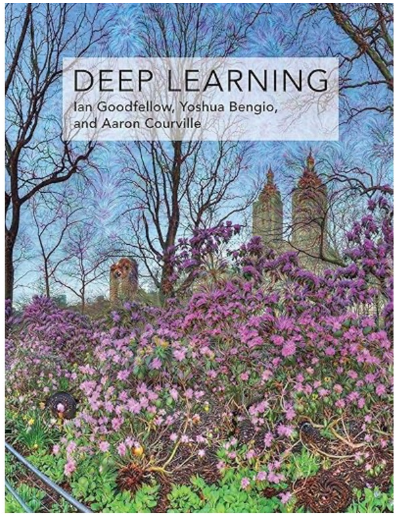
- Development tasks
  - module #1
    - sensor-specific data preprocessing
    - µC to PC communication
    - feature extraction and visualization
  - module #2
    - training and deployment of traditional ML techniques, e.g. linear regression, decision tree, support vector machine, clustering, …
  - module #3
    - training and deployment of deep neural networks, e.g. multilayer perceptron (MLP), convolutional neural network (CNN), …
    - model optimization/compression



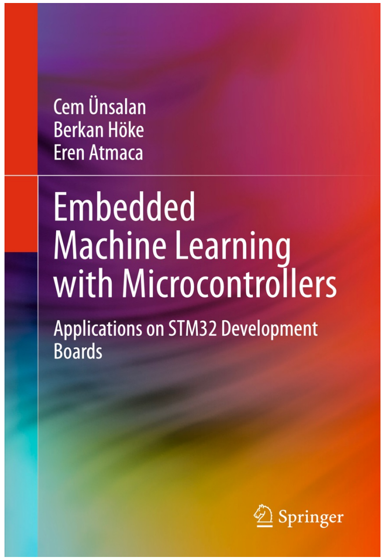
Literature (Selection)



A. Burkov: *The Hundred-Page Machine Learning Book.* 2019 (Web Link)



I. Goodfellow, Y. Bengio, A. Courville: *Deep Learning*, MIT Press, 2016 (https://www.deeplearningbook.org)


Cem Ünsalan
Berkan Höke
Eren Atmaca
Embedded Machine Learning with Microcontrollers
Applications on STM32 Development Boards
Springer

C. Ünsalan, B. Hoke, E. Atmaca: *Embedded Machine Learning with Microcontrollers – Application on STM32 Development Boards*, Springer, 2025 (Open Access)

Lecture Organization

- Materials and information in PANDA

- Lecture & Exercises
  - Wednesday 16:15 – 17:45
  - exercise sheets provided, try to solve the problems on your own, discussion of solutions in class

- Lab
  - announced in PANDA

- Contact
  - Marco Platzner, platzner@upb.de, 60-5250
  - Christoph Berganski, christoph.berganski@uni-paderborn.de, 60-4343
  - Andre Diekwisch, andre.diekwisch@uni-paderborn.de, 60-1744

- Study achievement
  - successfully complete one task in each of the three modules

- Grading
  - written exam, covering material from the lecture, exercises and lab
  - successful lab participation earns a bonus of one or two grade steps (if exam has been passed)