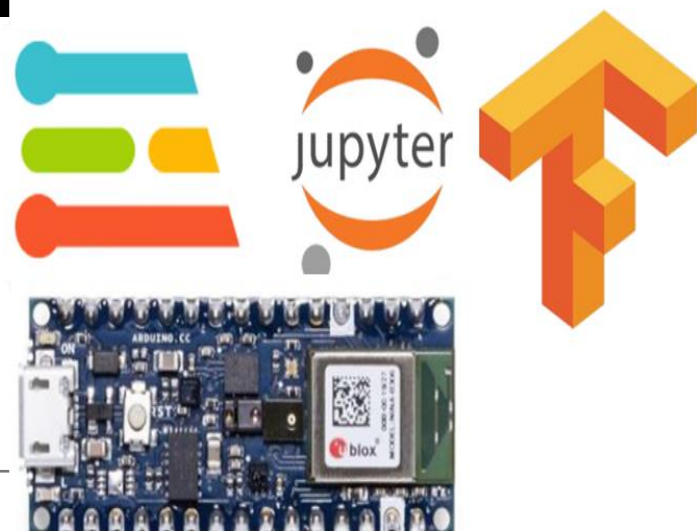


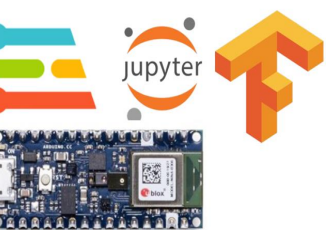


AI

TINYML CHALLENGES

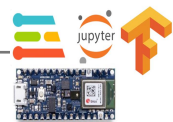
Dennis A. N. Gookyi

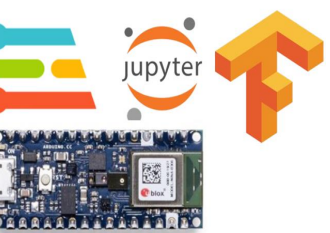




CONTENTS

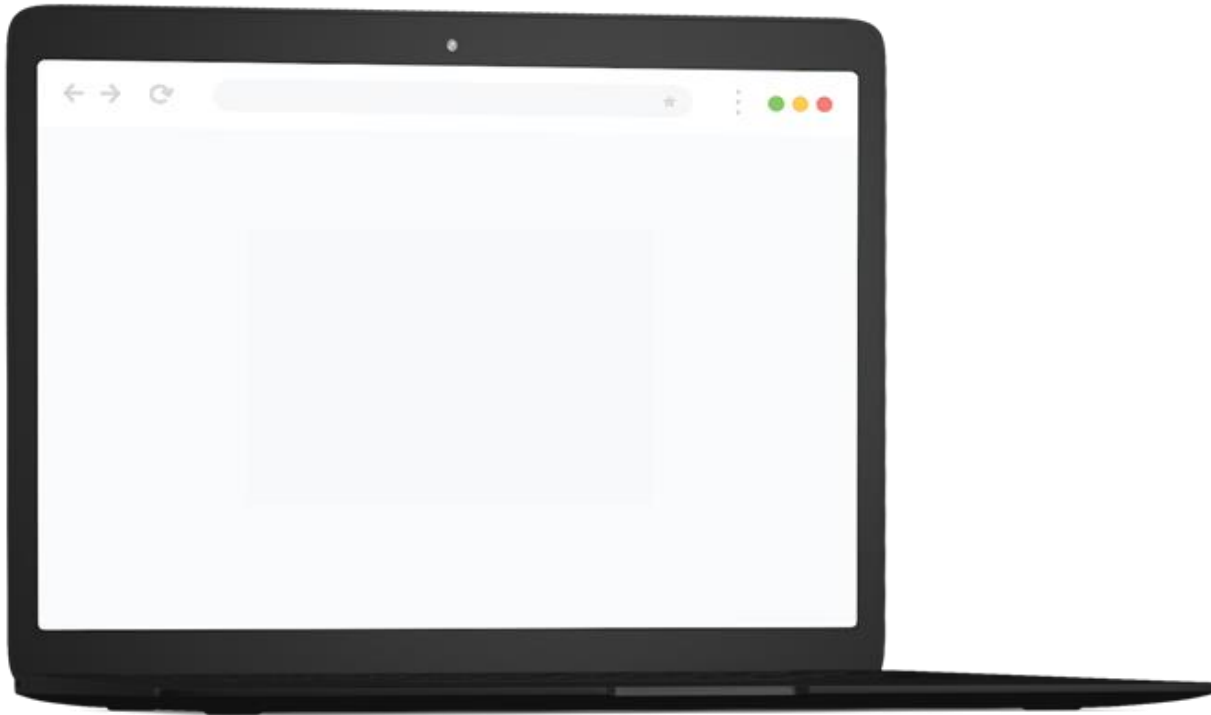
❖ TinyML Challenges

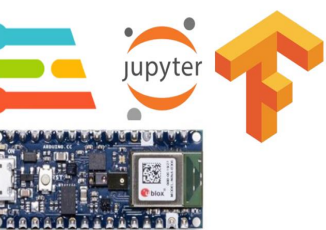




BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware





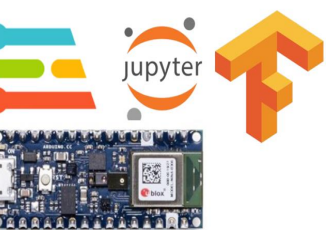
BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

Hardware



Software



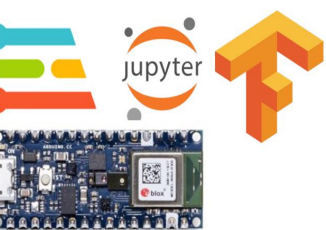
BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

Hardware



Software



BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

□ Hardware

Compute

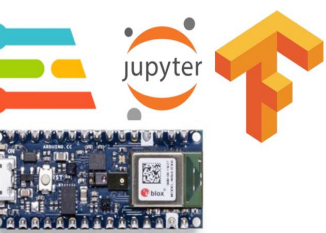


Memory



Storage





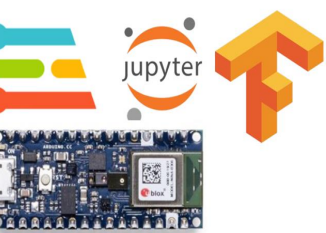
BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

- Hardware

Micro**processor**
v.

Micro**controller**

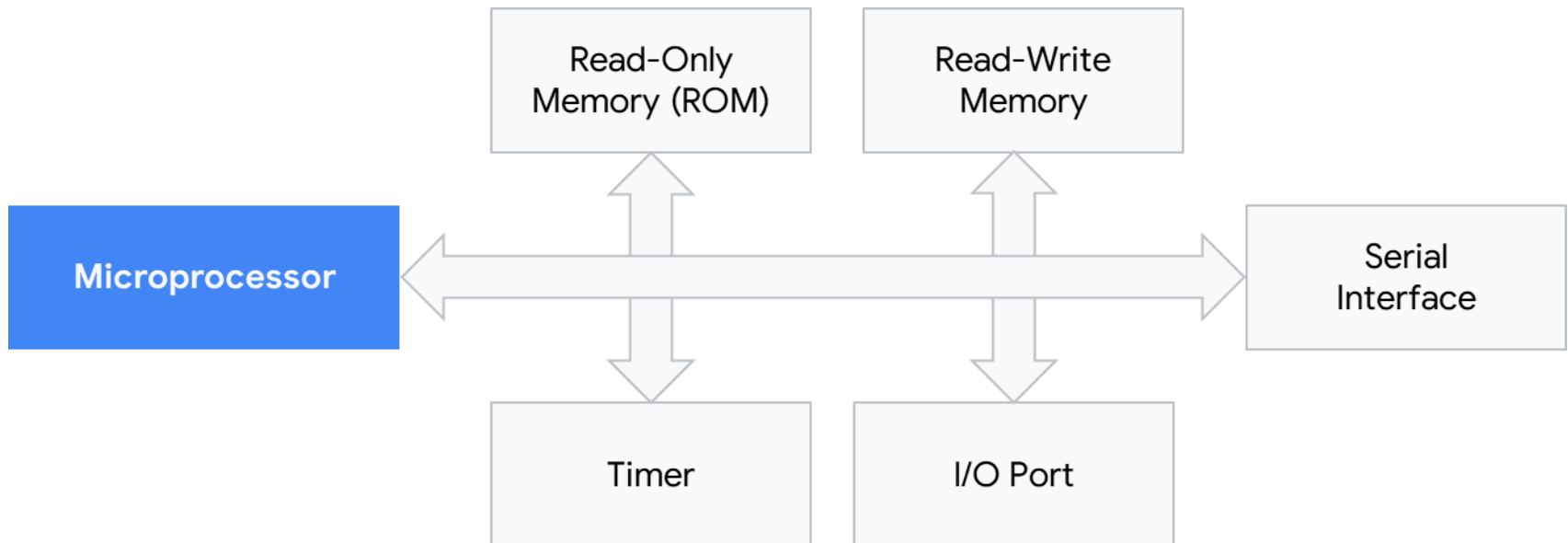


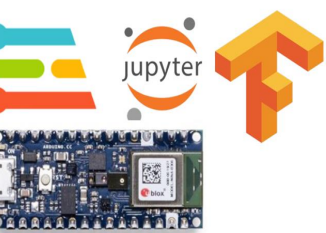
BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

□ Hardware

- Microprocessor: only one part of the puzzle



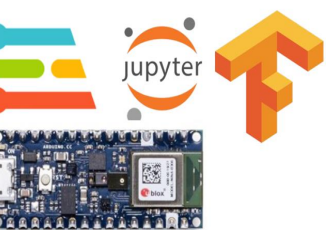


BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

- Hardware
 - Microcontroller

CPU	Read-Only Memory (ROM)	Read-Write Memory
Timer	I/O Port	Serial Interface

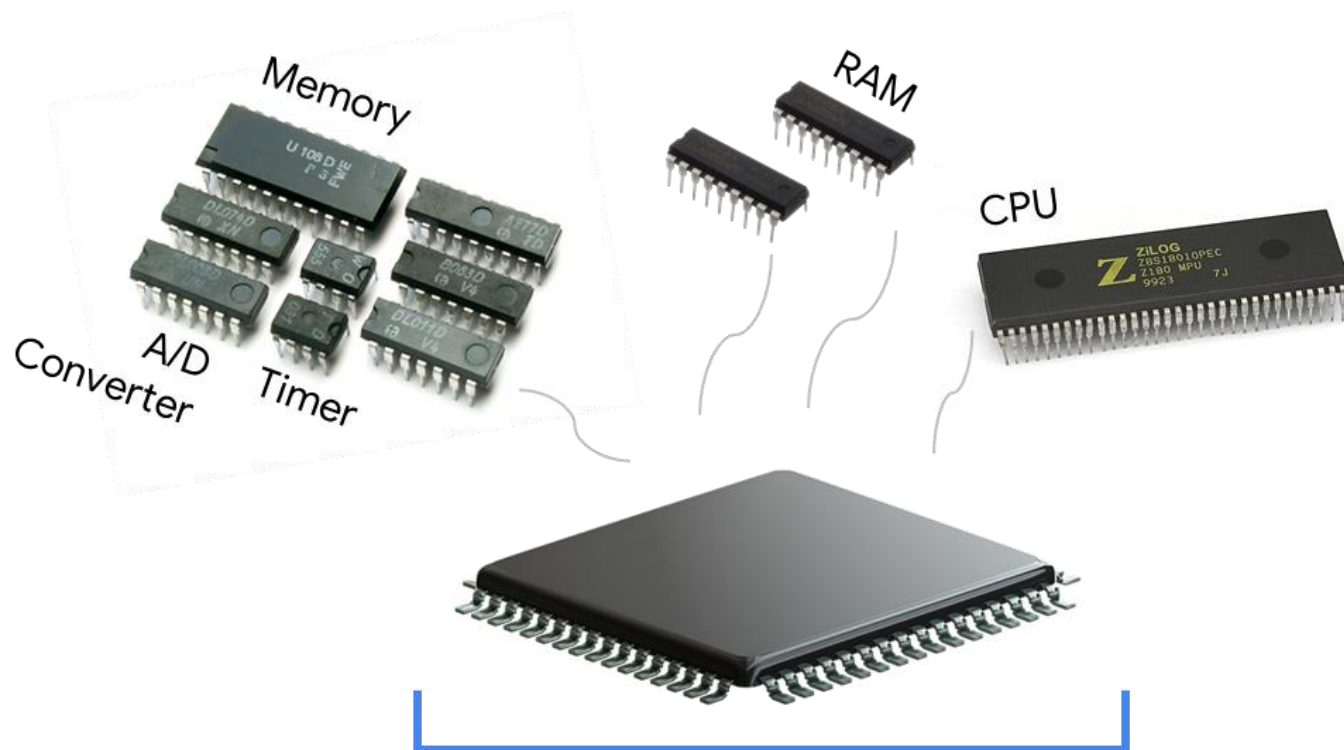


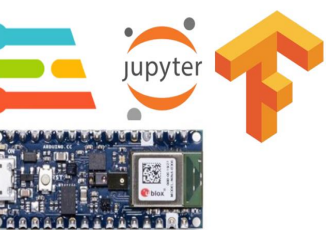
BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

□ Hardware

- Microcontroller: a complete package





BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

□ Hardware

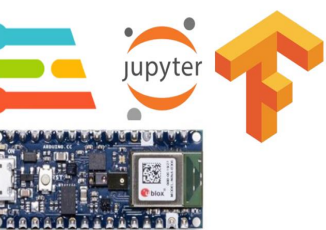
- Microcontroller vs Microprocessor

Microprocessor

- Heart of a **computer system**
- Just the processor, memory and storage are **external**
- Mainly used in **general purpose systems** like laptops, desktops and servers
- **Offers flexibility** in design
- System size is **big**

Microcontroller



- Heart of an **embedded system**
- Memory and storage are all **internal** to the system
- Mainly used in **specialized, fixed function systems** like phones, MP3 players, etc.
- **Limited flexibility** in design
- System size is **tiny**

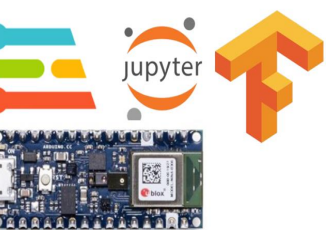


BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

- Hardware
 - Microcontroller vs Microprocessor

	Microprocessor	>	Microcontroller	
Platform				Nano
Compute	1GHz–4GHz	~10X	1MHz–400MHz	64MHz
Memory	512MB–64GB	~10000X	2KB–512KB	256KB
Storage	64GB–4TB	~100000X	32KB–2MB	1MB
Power	30W–100W	~1000X	150μW–23.5mW	




BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

- Hardware
 - Microcontroller

Implications

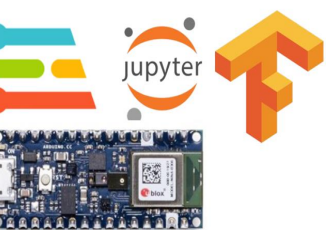
- How complicated is the running task?
- How much memory does it need to have?
- How long does the job have to perform?

Microcontroller

1MHz-400MHz
2KB - 512KB
32KB - 2MB
150μW-23.5mW

BUILDING BLOCKS OF COMPUTING HARDWARE

- ❖ Building Blocks of Computing Hardware
 - Computing hardware



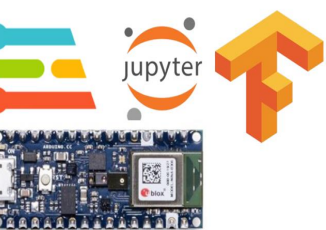


BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

□ Computing hardware

	Board	MCU / ASIC	Clock	Memory	Sensors	Radio
	Himax WE-I Plus EVB	HX6537-A 32-bit EM9D DSP	400 MHz	2MB flash 2MB RAM	Accelerometer, Mic, Camera	None
	Arduino Nano 33 BLE Sense	32-bit nRF52840	64 MHz	1MB flash 256kB RAM	Mic, IMU, Temp, Humidity, Gesture, Pressure, Proximity, Brightness, Color	BLE
	SparkFun Edge 2	32-bit ArtemisV1	48 MHz	1MB flash 384kB RAM	Accelerometer, Mic, Camera	BLE
	Espressif EYE	32-bit ESP32-D0WD	240 MHz	4MB flash 520kB RAM	Mic, Camera	WiFi, BLE



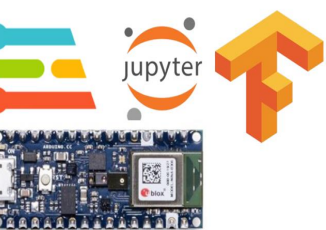
BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

□ Computing hardware



Board	MCU / ASIC	Clock	Memory	Sensors	Radio
Himax WE-I Plus-EVB	HX6537-A 32-bit EM9D DSP	400 MHz	2MB flash 2MB RAM	Accelerometer, Mic, Camera	None
Arduino Nano 33 BLE Sense	32-bit nRF52840	64 MHz	1MB flash 256kB RAM	Mic, IMU, Temp. Humidity, Gesture, Pressure, Proximity, Brightness, Color	BLE
SparkFun Edge 2	32-bit ArtemisV1	48 MHz	1MB flash 384kB RAM	Accelerometer, Mic, Camera	BLE
Espressif EYE	32-bit ESP32-D0WD	240 MHz	4MB flash 520kB RAM	Mic, Camera	WiFi, BLE



BUILDING BLOCKS OF COMPUTING HARDWARE

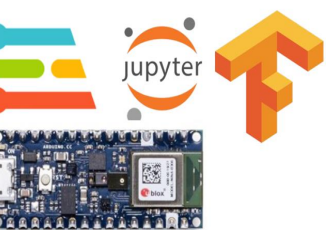
❖ Building Blocks of Computing Hardware

- Software

Hardware



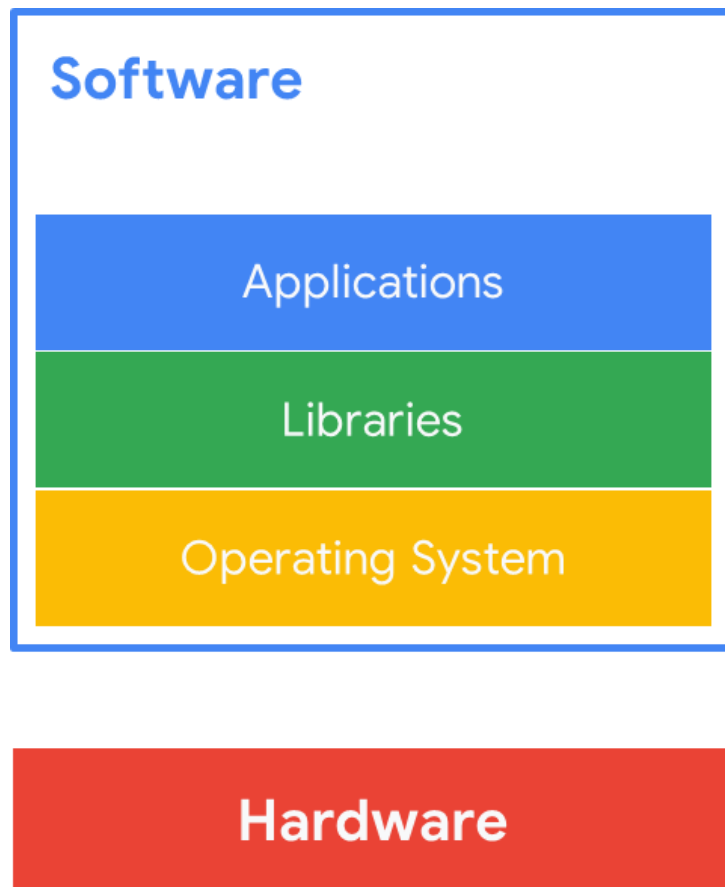
Software

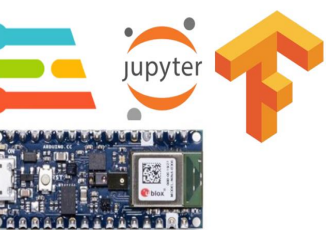


BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

□ Software



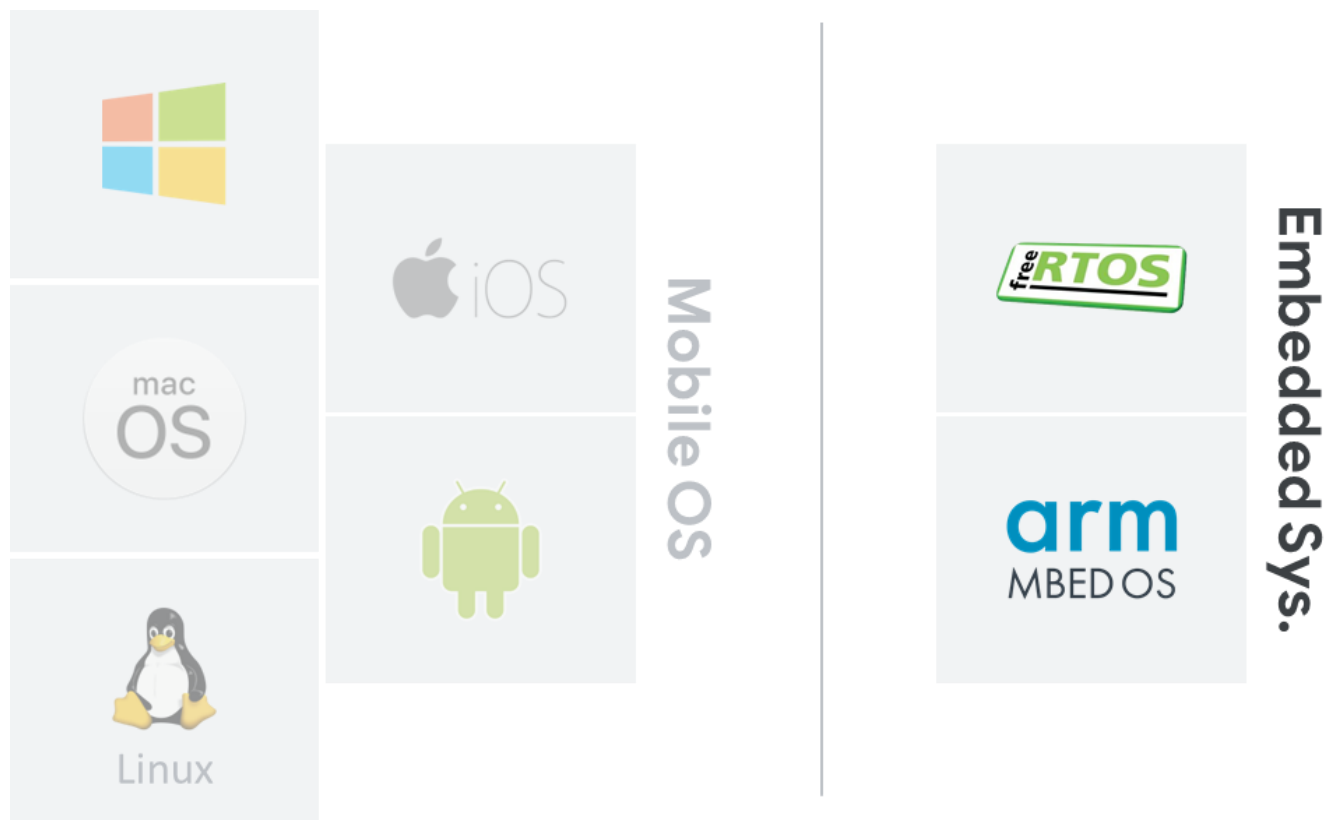


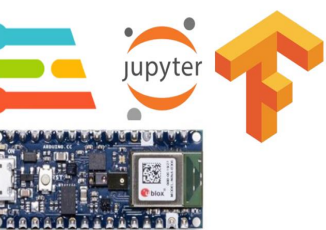
BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

□ Software

- Widely used operating systems





BUILDING BLOCKS OF COMPUTING HARDWARE

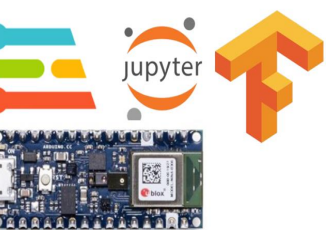
❖ Building Blocks of Computing Hardware

- Software
 - Libraries



```
import numpy as np

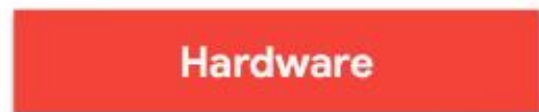
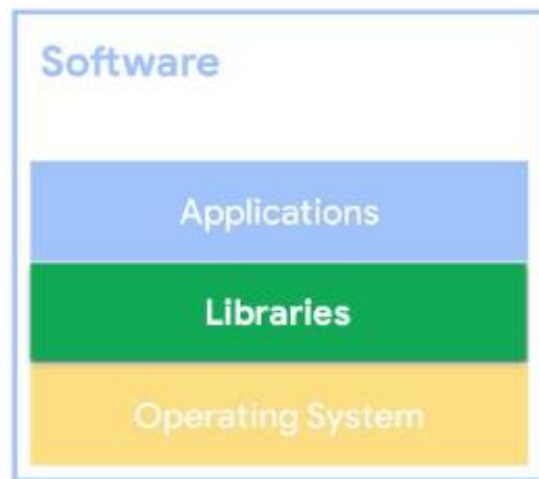
for x in range(10):
    np.SaveTheWorld()
```



BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

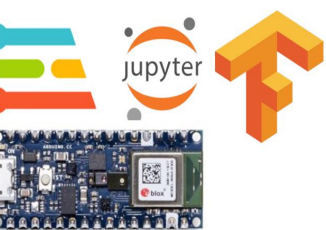
□ Software



Portability Opportunity

Able to execute the same code on different microprocessor hardware and architectures.



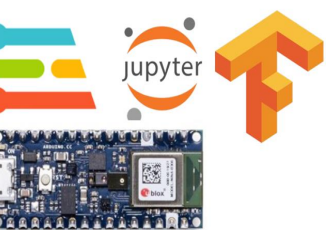


BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

□ Portability trade-offs

Option 1	Universal Code Portability/Compatibility		✓
	Cost (\$)		✗
	Power Consumption (W)		✗
	Engineering Effort		✗
Option 2	Lower Code Portability		✗
	Cost (\$)	✓	
	Power (W)	✓	
	Eng. Effort	✓	

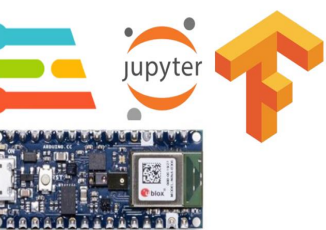


BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

□ Portability trade-offs

Option 1	Universal Code Portability/Compatibility		✓
	Cost (\$)		✗
	Power Consumption (W)		✗
	Engineering Effort		✗
Option 2	Lower Code Portability		✗
	Cost (\$)	✓	
	Power (W)	✓	
	Eng. Effort	✓	

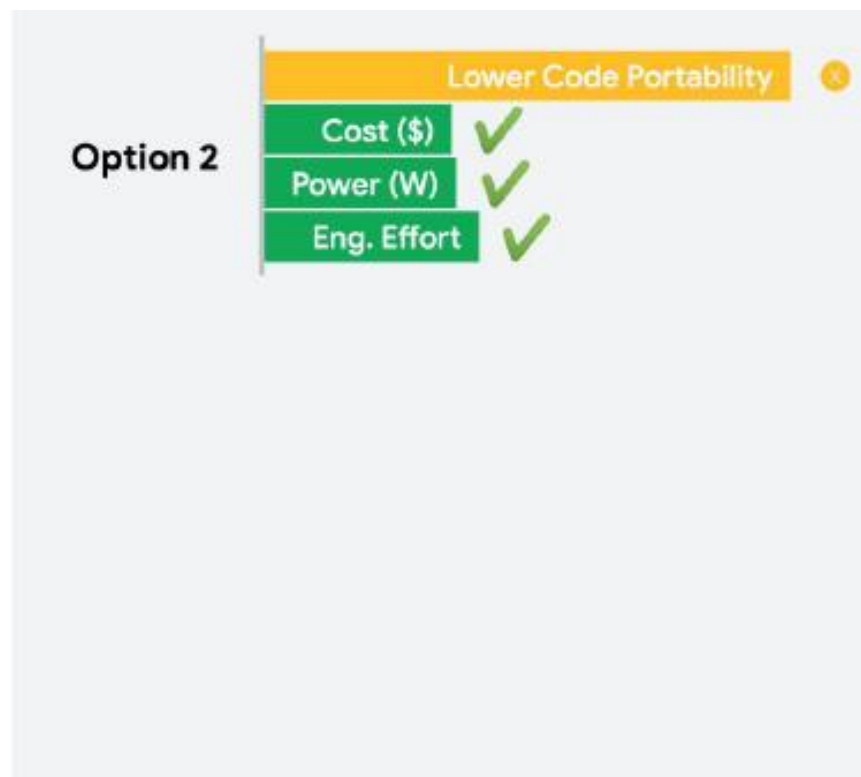


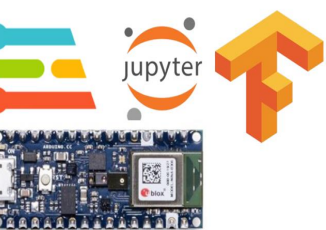
BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

□ Portability trade-offs

Sacrifice portability across systems for efficiency in system performance and power efficiency





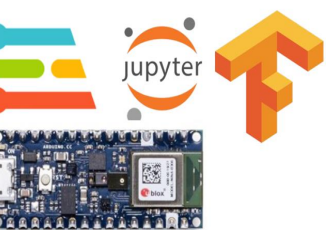
BUILDING BLOCKS OF COMPUTING HARDWARE

❖ Building Blocks of Computing Hardware

□ Summary

- **Embedded hardware** is extremely limited in performance, power consumption and storage

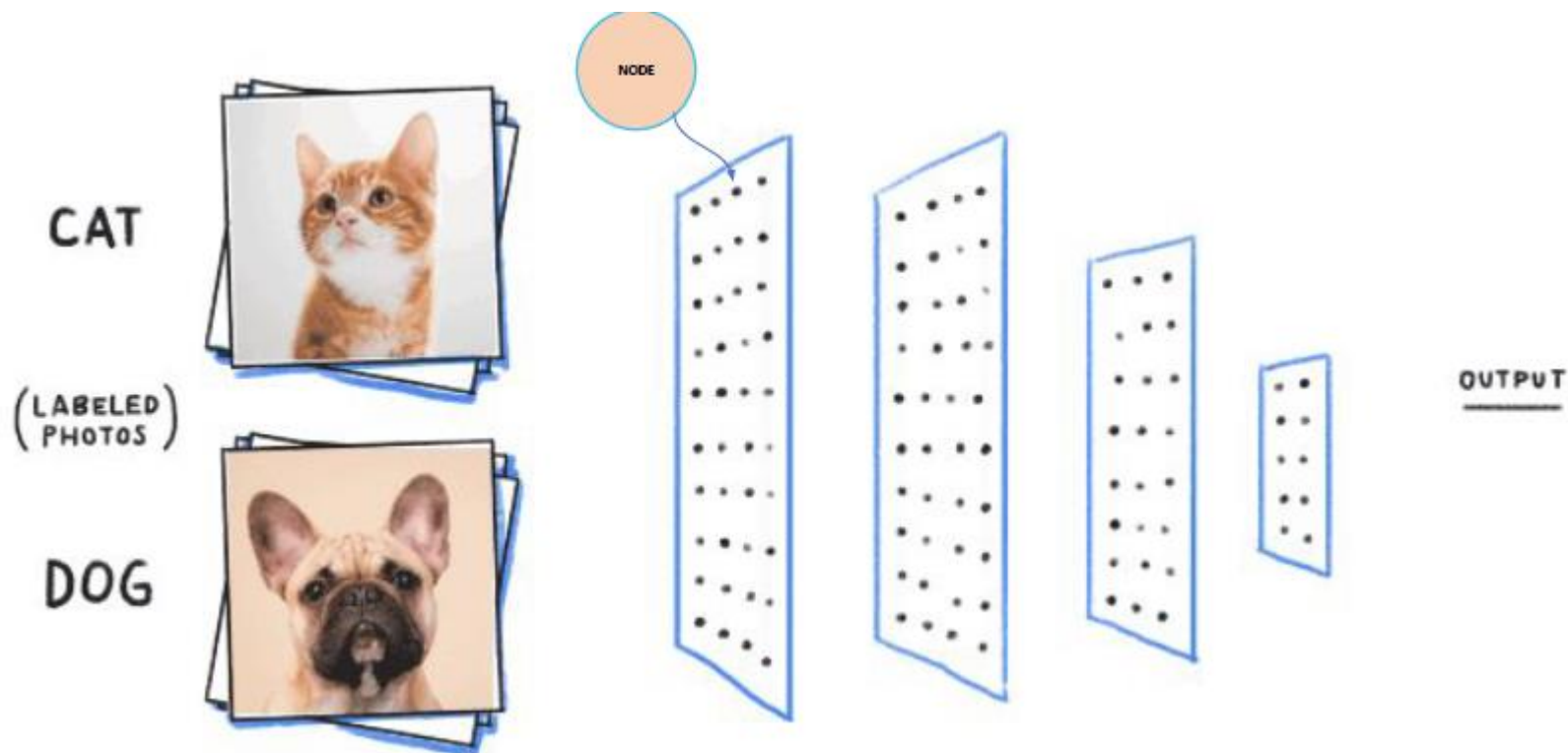
- **Embedded software** is not as portable and flexible as mainstream computing

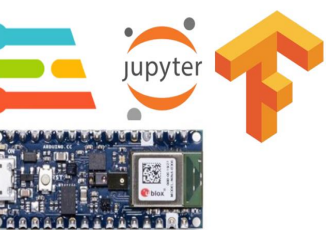


MACHINE LEARNING

❖ Machine Learning

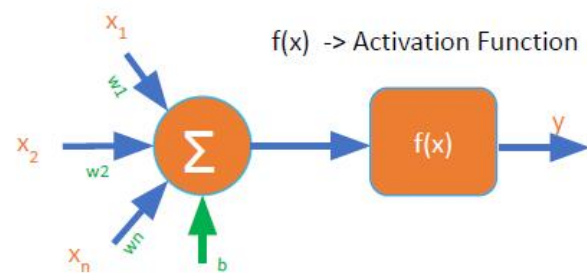
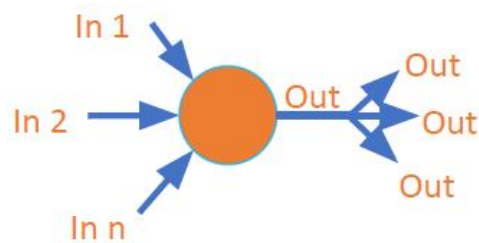
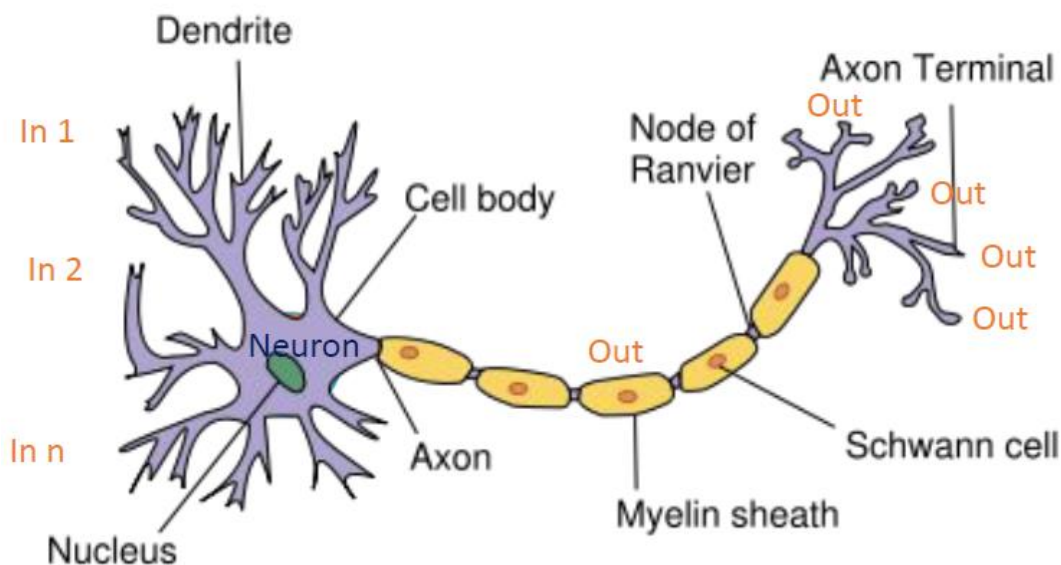
- Deep Learning: Subset of Machine Learning in which multilayered neural networks learn from vast amounts of data





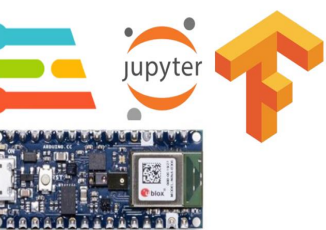
MACHINE LEARNING

- ❖ Machine Learning
 - Neuron (Perceptron)



Parameters

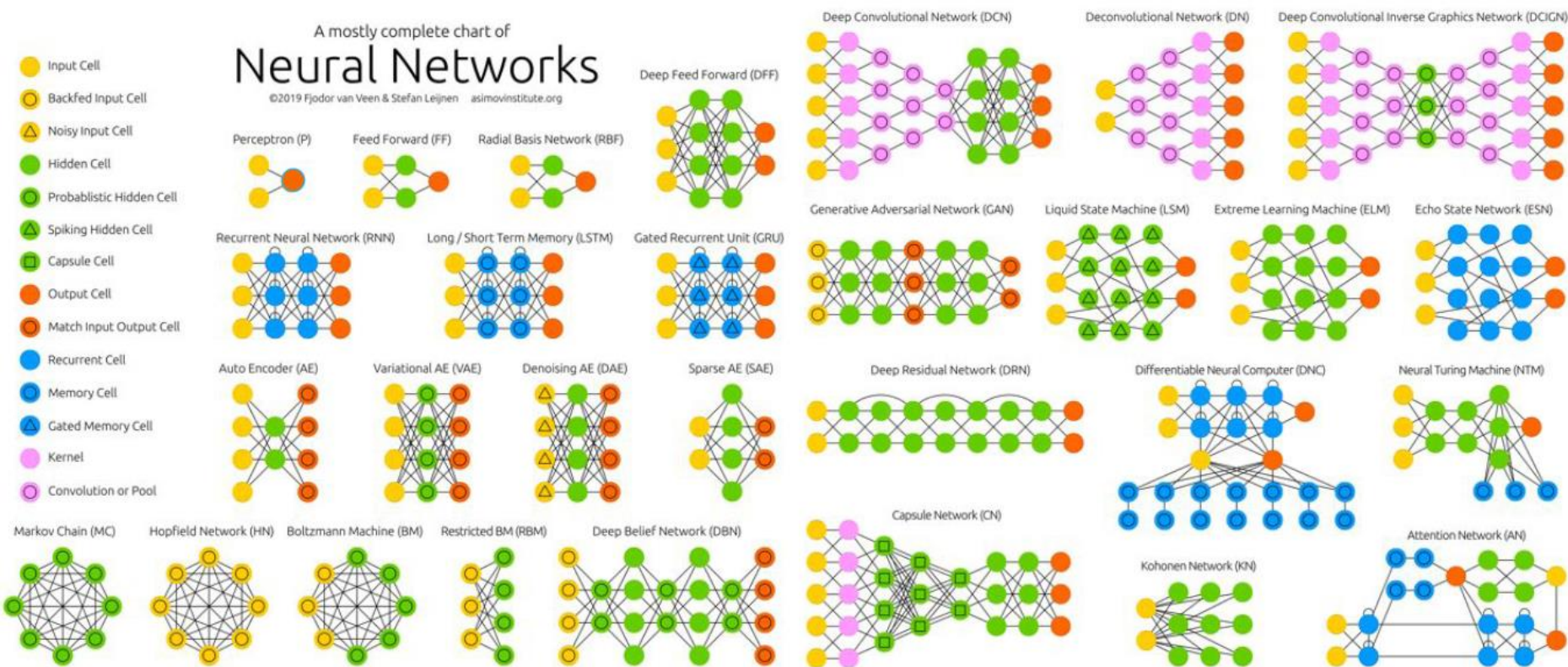
$$y = f\left(\sum_{i=1}^n x_i w_i + b\right)$$

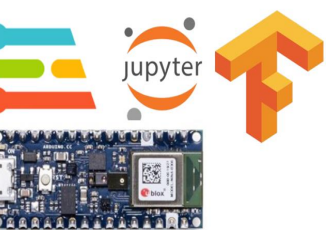


MACHINE LEARNING

❖ Machine Learning

□ The Neural Network model architecture

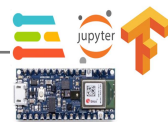
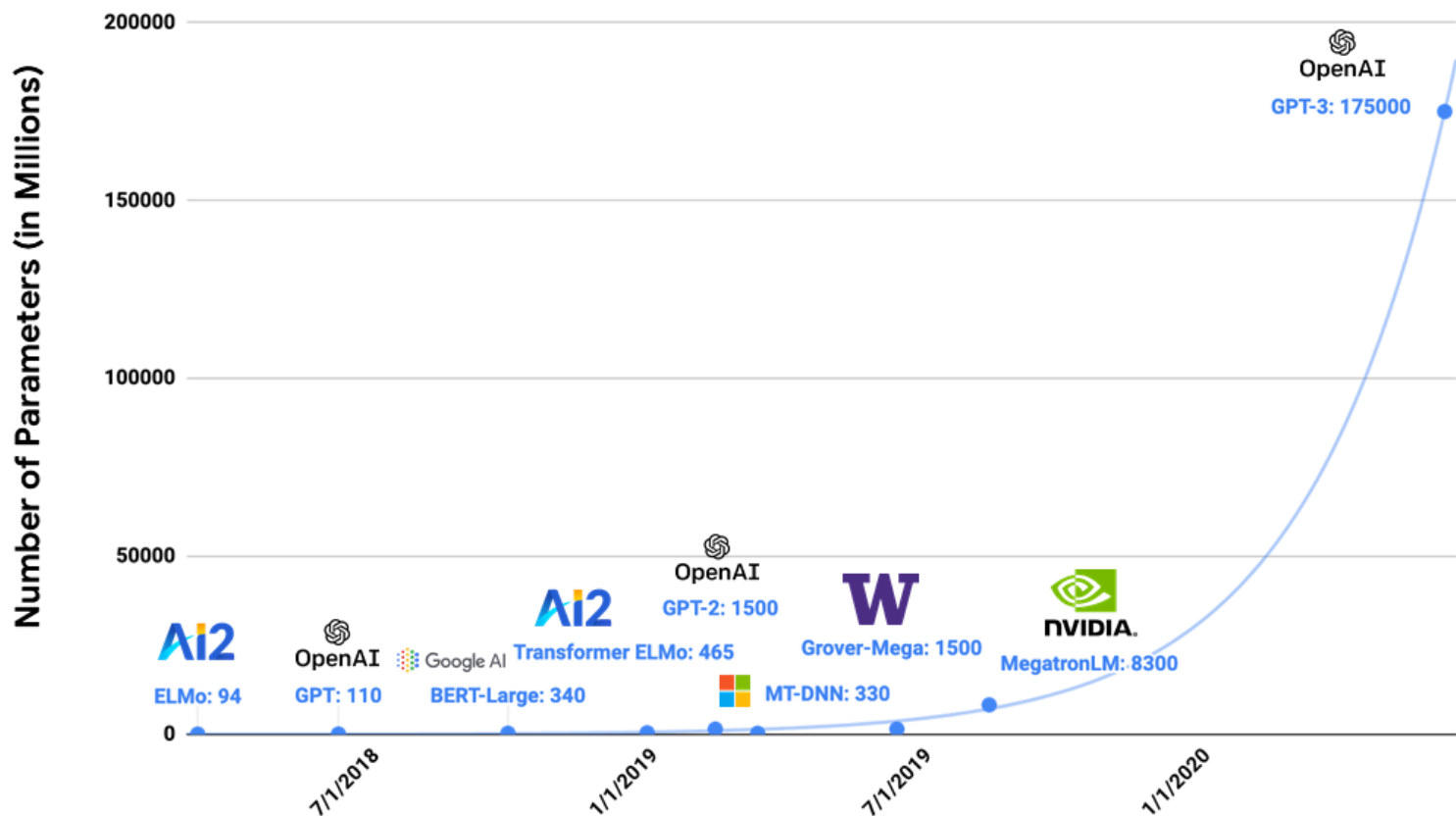


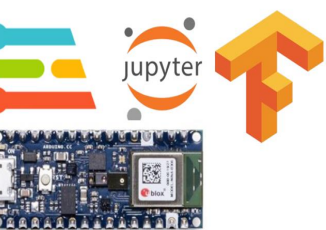


MACHINE LEARNING

❖ Machine Learning

□ ML model size growth



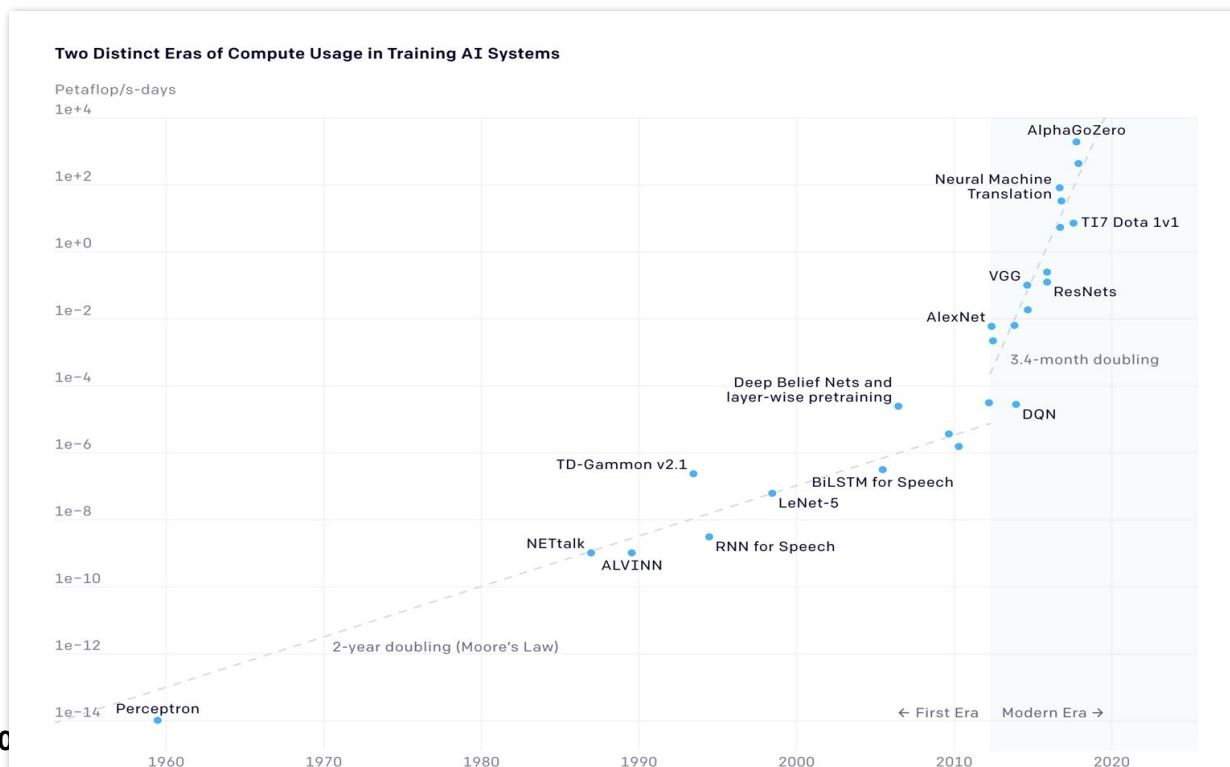


MACHINE LEARNING

❖ Machine Learning

□ ML compute needs

- In recent years, the amount of computing needed has grown remarkably fast
- Compute requirements are doubling nearly every 3 to 4 months



ML COMPUTE NEEDS
(FROM THE 1960S)

MACHINE LEARNING

- ❖ Machine Learning
 - ML compute needs

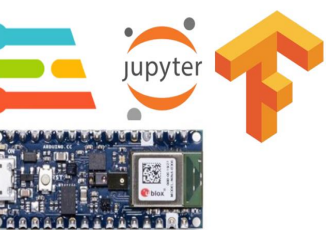


Cloud TPU



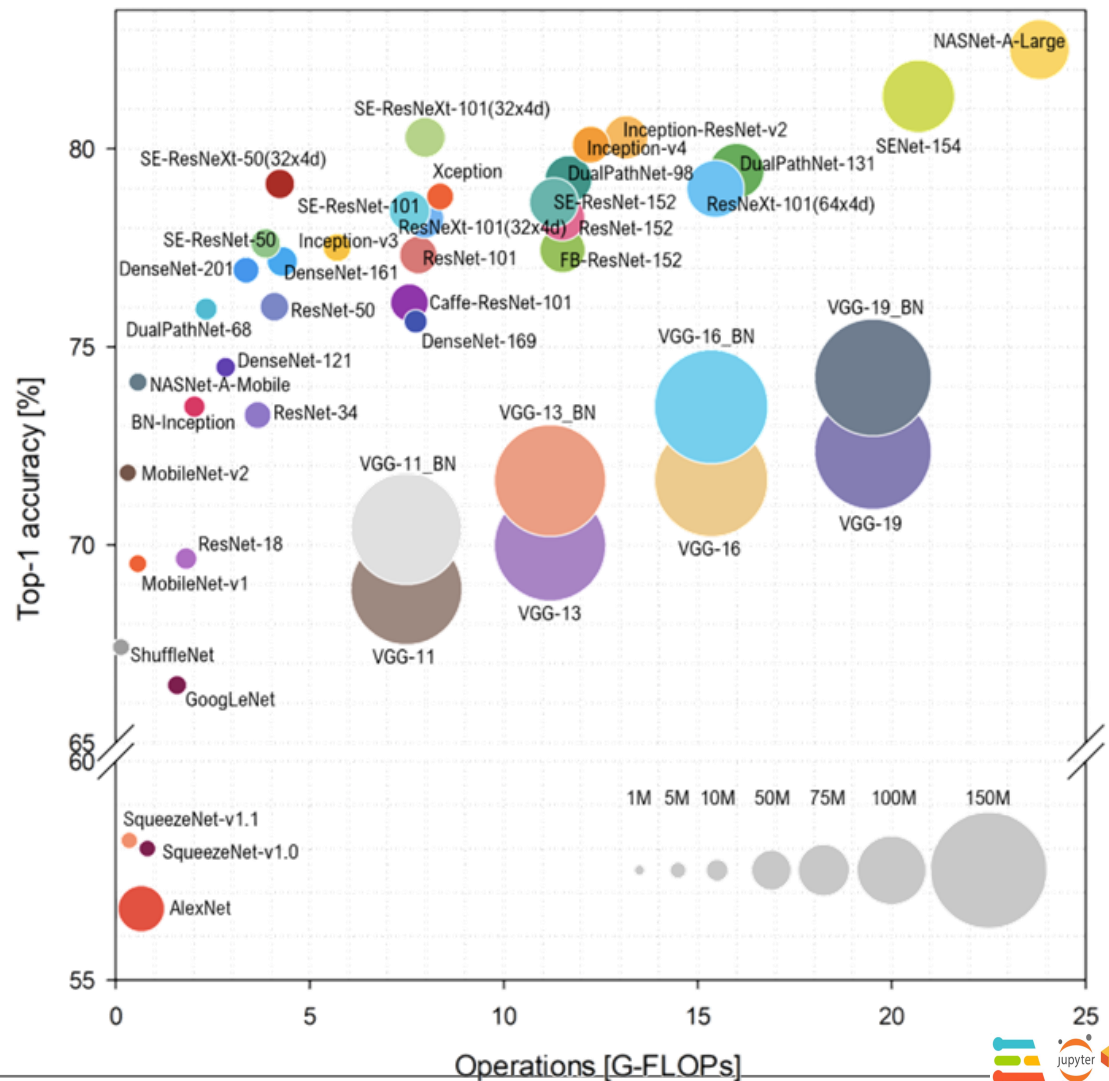
TinyML

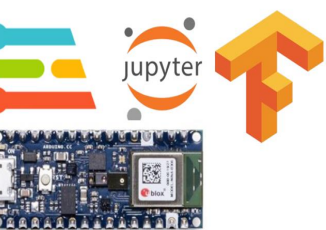




MACHINE LEARNING

- ❖ Machine Learning
- ML model evolution

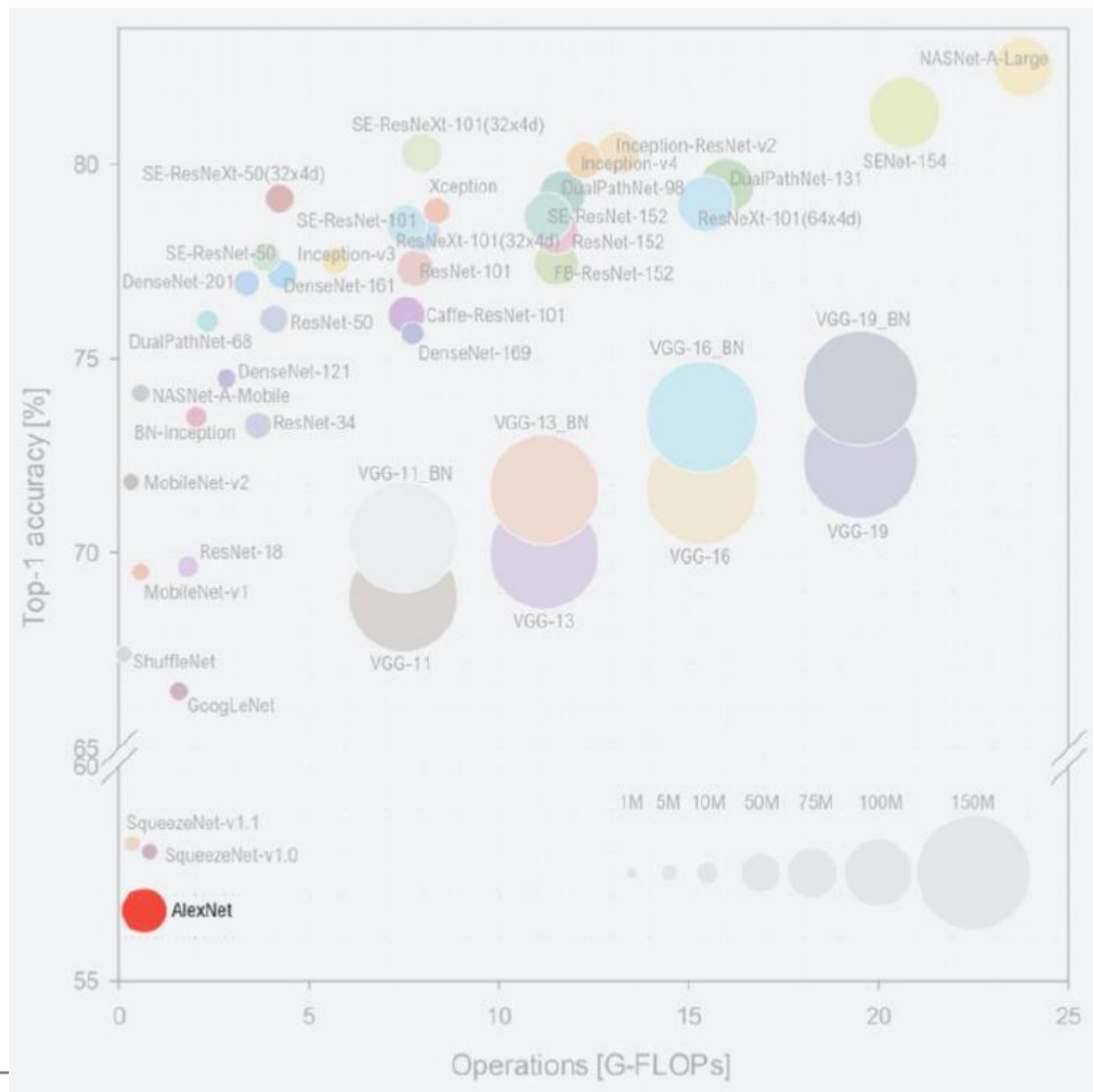


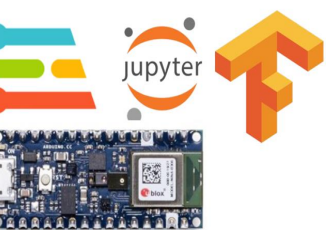


MACHINE LEARNING

- ❖ Machine Learning
 - ML model evolution

- **AlexNet (2012)**
 - 57.1% accuracy
 - 61MB in size

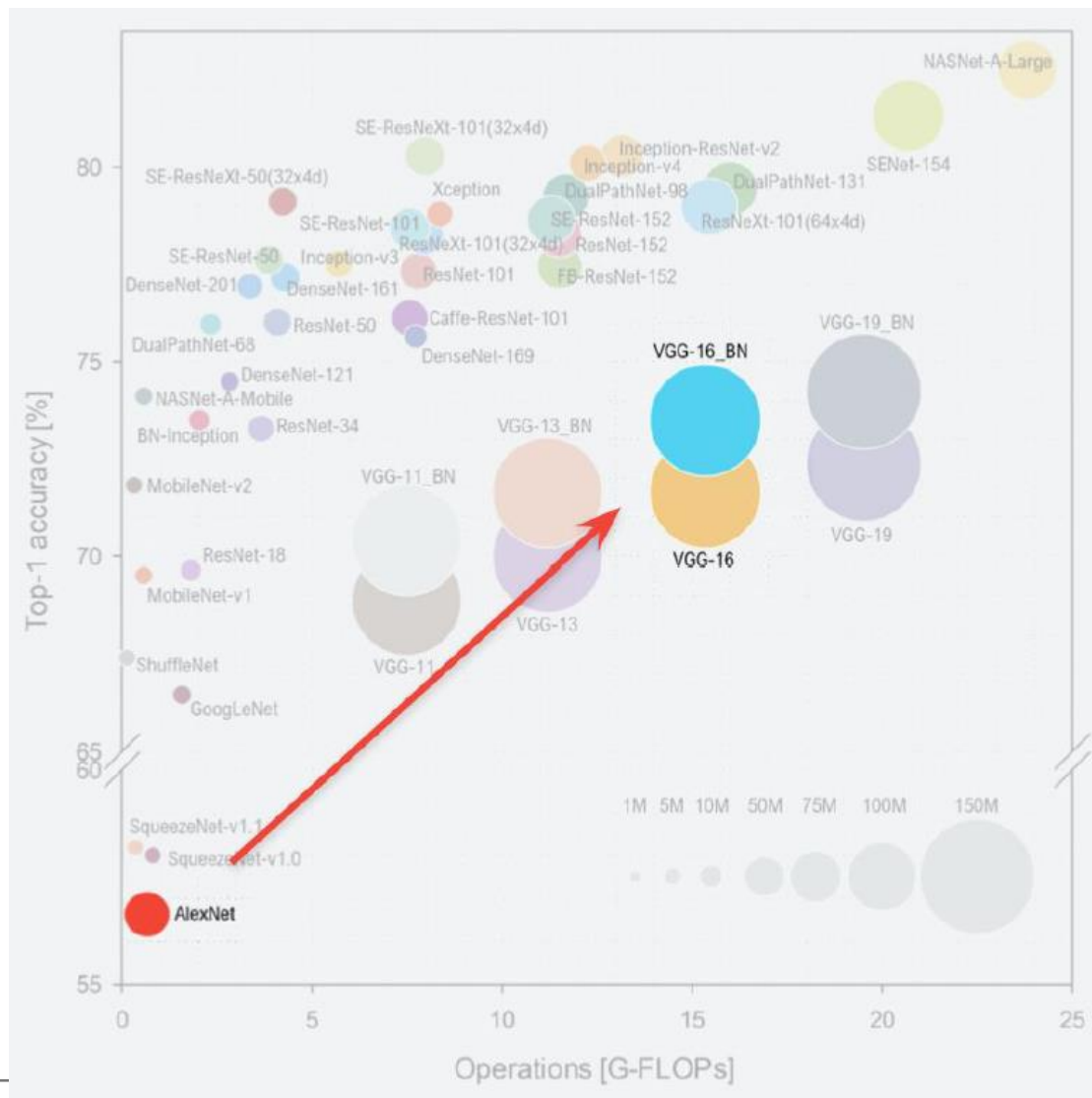


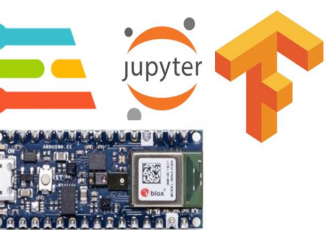


MACHINE LEARNING

- ❖ Machine Learning
 - ML model evolution

- **VGGNet (2014)** [VGG-16]
 - **71.5%** accuracy
 - **528MB** in size

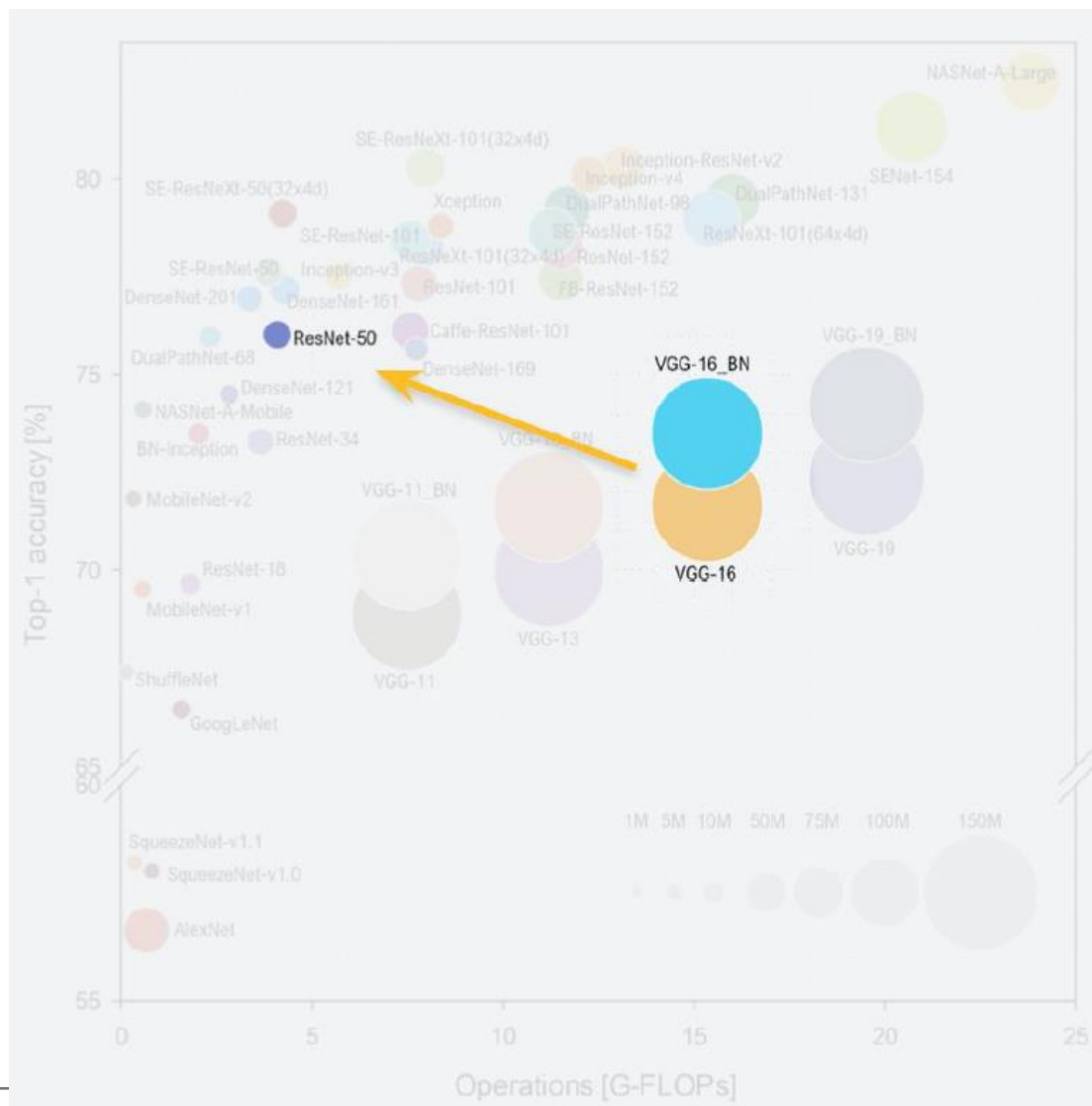


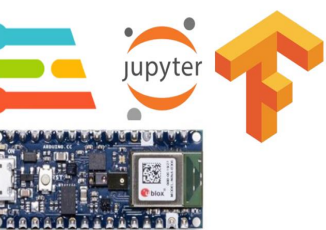


MACHINE LEARNING

- ❖ Machine Learning
 - ML model evolution

- **ResNet (2015)**
 - **75.8%** accuracy
 - **22.7MB** in size

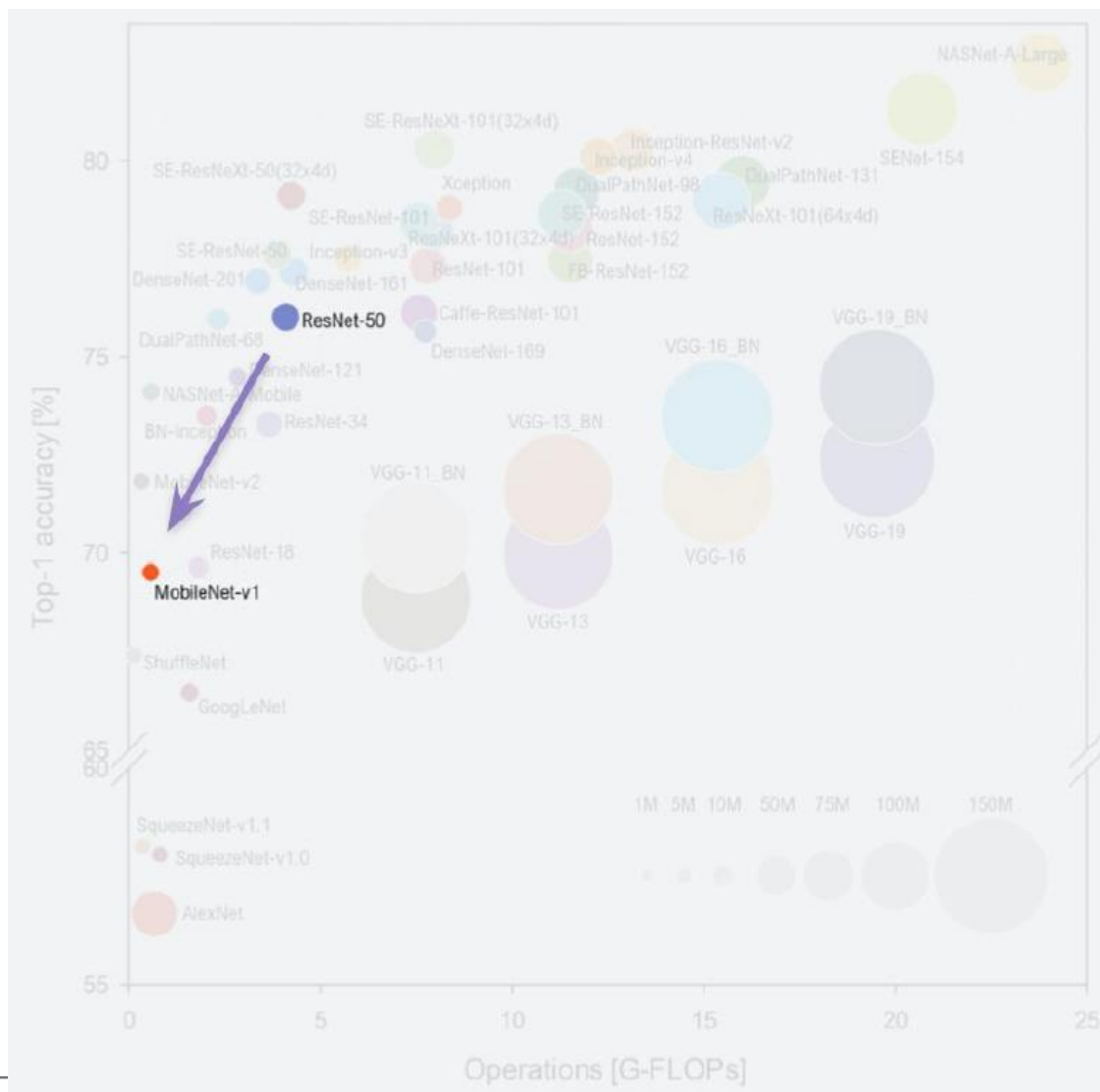


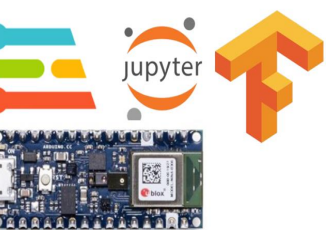


MACHINE LEARNING

- ❖ Machine Learning
 - ML model evolution

- **MobileNet (2015)**
 - **MobileNetv1**
 - **70.6%** accuracy
 - **16.9MB** in size





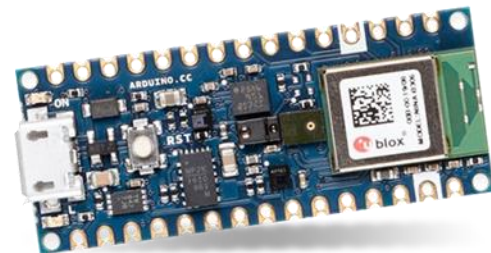
MACHINE LEARNING

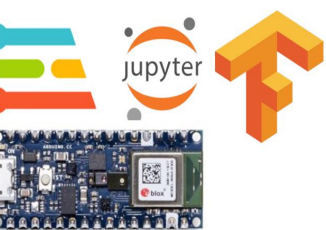
- ❖ Machine Learning
 - ML model evolution

- **MobileNet (2015)**
 - **MobileNetv1**
 - 70.6% accuracy
 - 16.9MB in size

Problem:

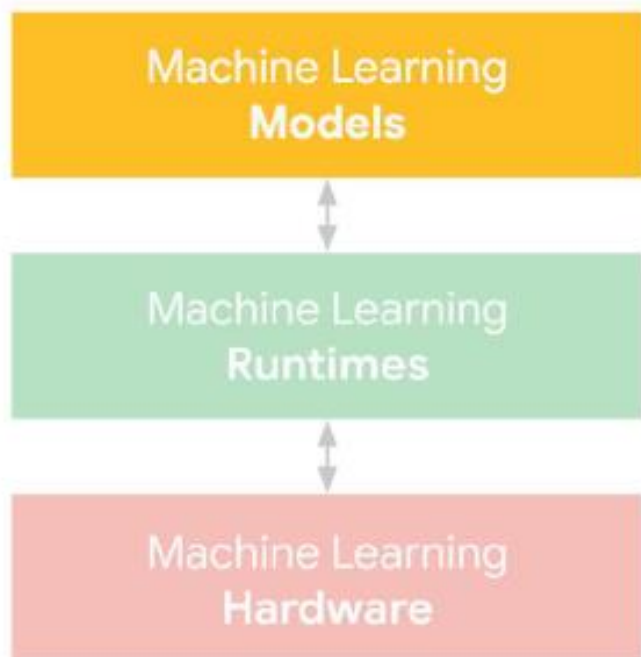
Our board (in your kit for Course 3) only has **256KB** of RAM (memory) yet **MobileNetv1** needs **16.9MB!**





MACHINE LEARNING

- ❖ Machine Learning
 - Model compression techniques



Model Compression Techniques

Pruning

Quantization

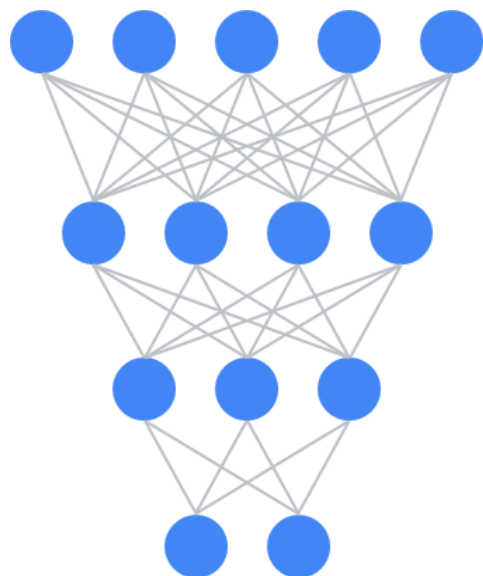
Knowledge Distillation

...

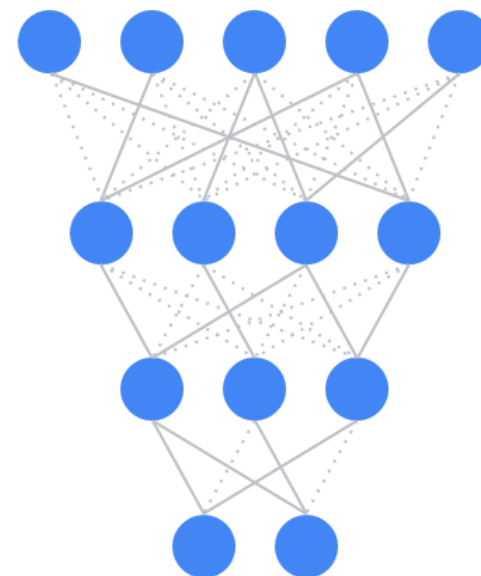
MACHINE LEARNING

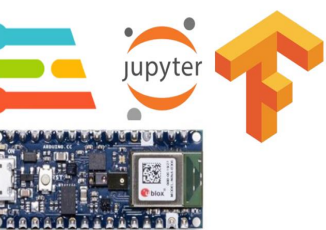
❖ Machine Learning

□ Pruning



PRUNING
SYNAPSES

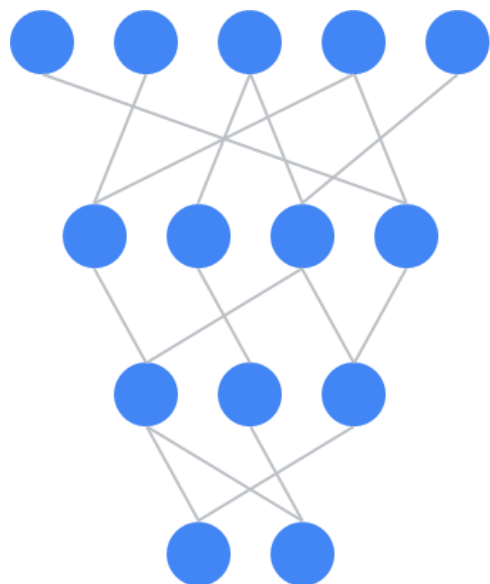




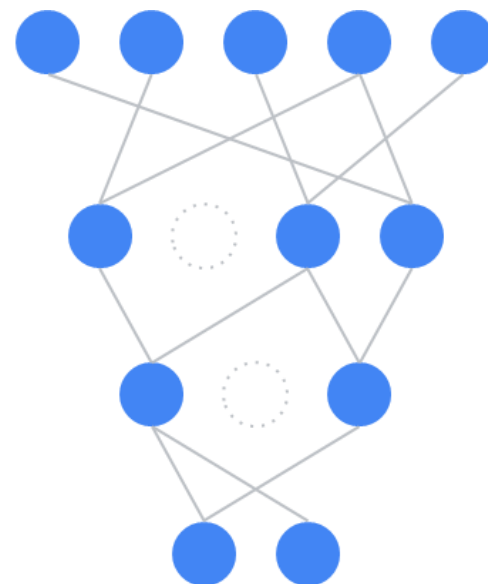
MACHINE LEARNING

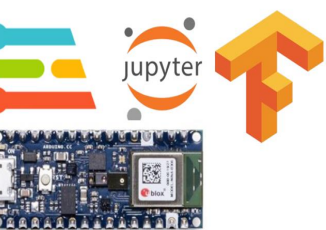
❖ Machine Learning

□ Pruning



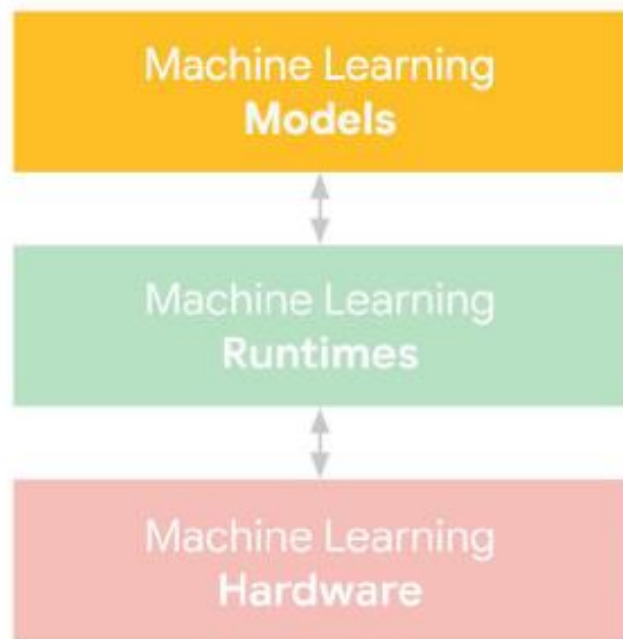
PRUNING
NEURONS





MACHINE LEARNING

- ❖ Machine Learning
 - Model compression techniques



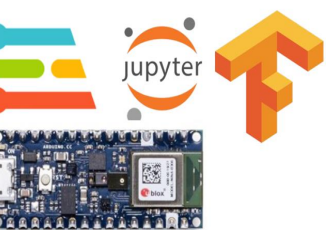
Model Compression Techniques

Pruning

Quantization

Knowledge Distillation

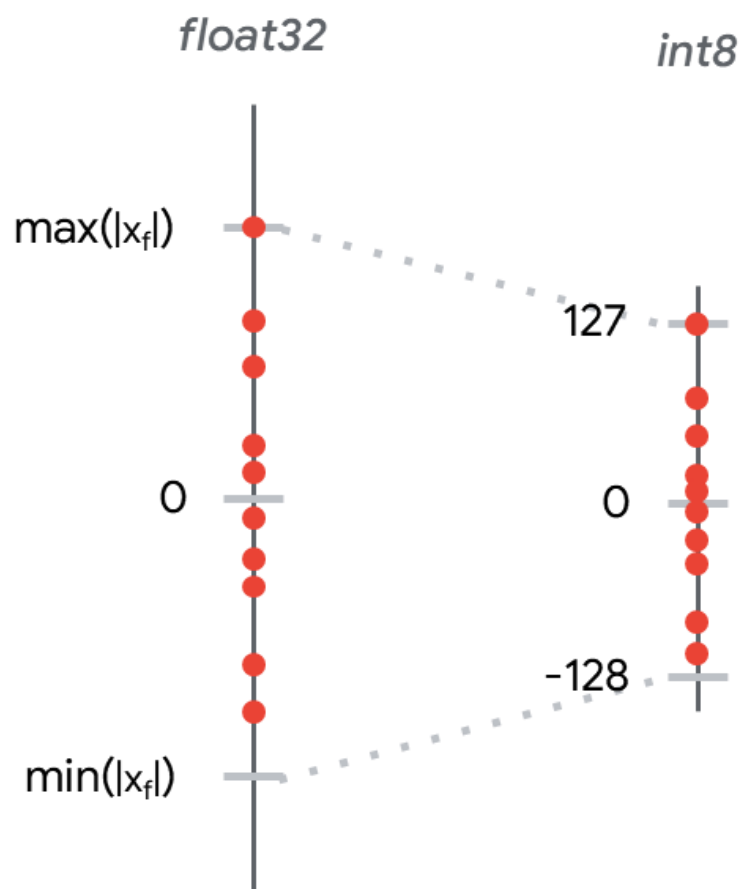
...

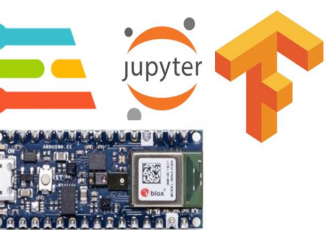


MACHINE LEARNING

❖ Machine Learning

□ Quantization

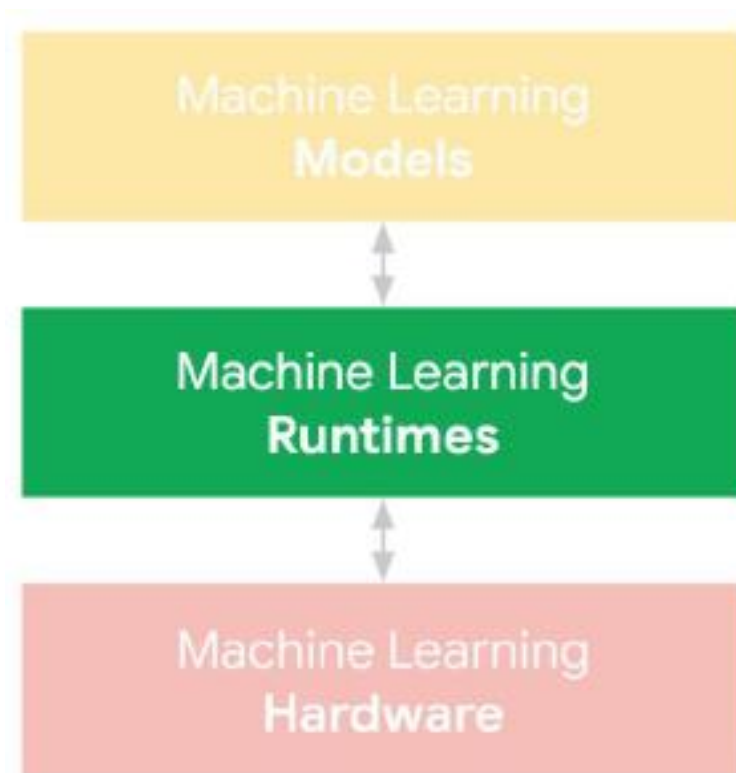


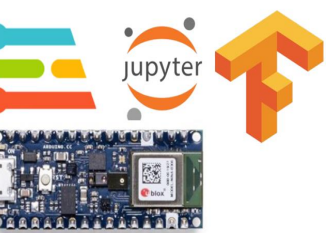


MACHINE LEARNING

❖ Machine Learning

□ Runtimes





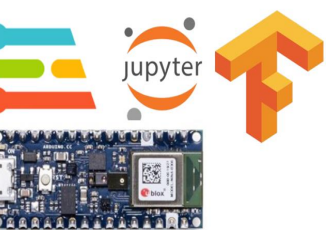
MACHINE LEARNING

❖ Machine Learning

□ Runtimes



TensorFlow



MACHINE LEARNING

❖ Machine Learning

□ Runtimes



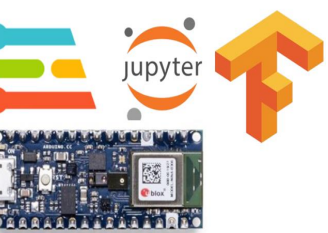
Less memory

Less compute power

Only focused on *inference*



TensorFlow Lite



MACHINE LEARNING

❖ Machine Learning

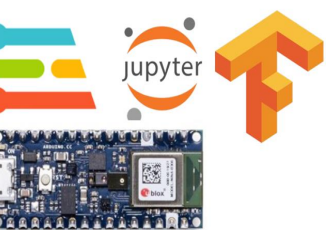
□ Runtimes



TensorFlow



TensorFlow Lite



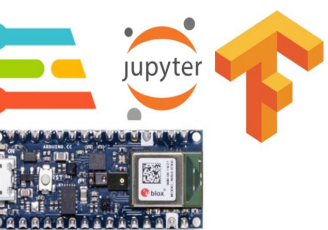
MACHINE LEARNING

❖ Machine Learning

□ Runtimes

- Key differences

	 TensorFlow	 TensorFlow Lite
Topology	Variable	Fixed
Weights	Variable	Fixed
Binary Size	Unimportant	High Priority
Distributed Compute	Needed	Not Needed
Developer Background	ML Researcher	Application Developer



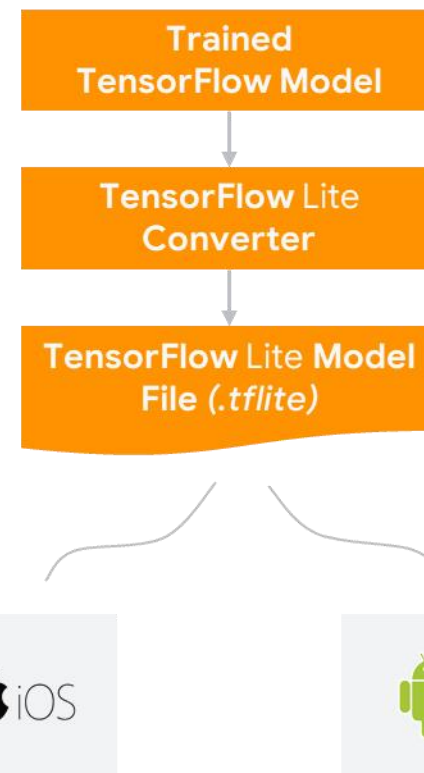
MACHINE LEARNING

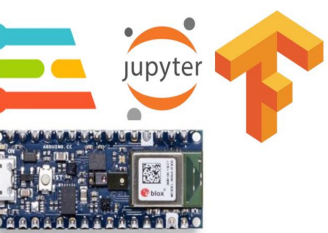
❖ Machine Learning

□ Runtimes



Architecture





MACHINE LEARNING

❖ Machine Learning

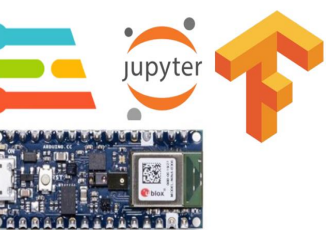
□ Runtimes



Even less memory

Even less compute power

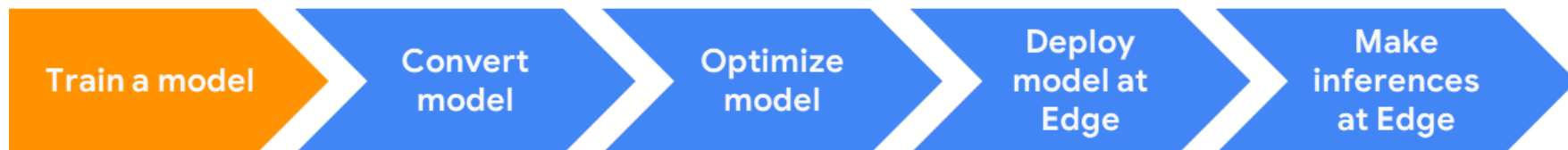
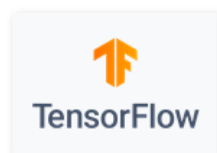
Also, only focused on *inference*

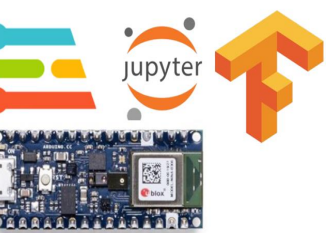


MACHINE LEARNING

❖ Machine Learning

□ Runtimes

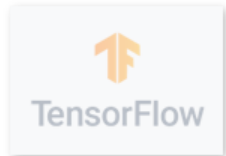




MACHINE LEARNING

❖ Machine Learning

□ Runtimes



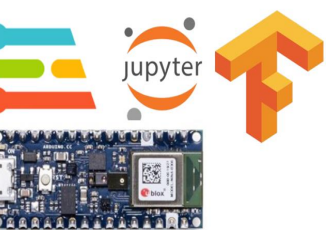
Train a model

Convert
model

Optimize
model

Deploy
model at
Edge

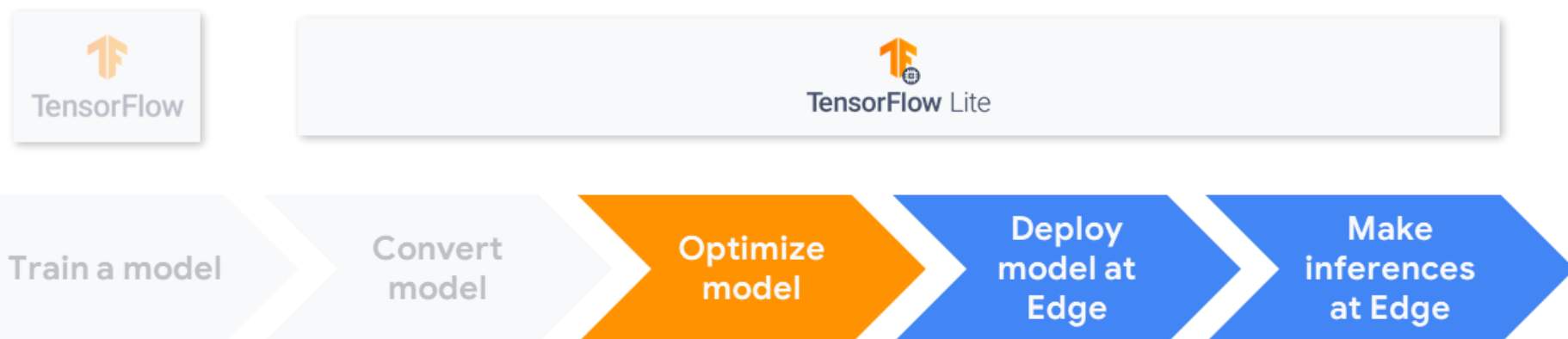
Make
inferences
at Edge

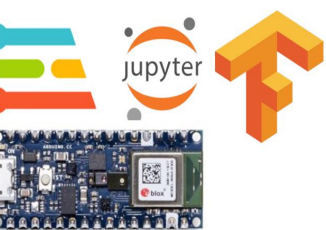


MACHINE LEARNING

❖ Machine Learning

□ Runtimes

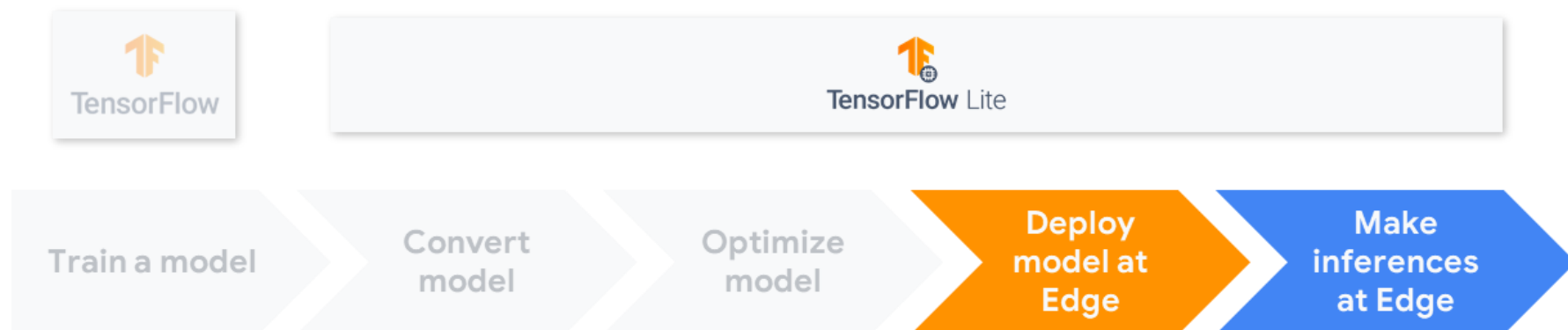


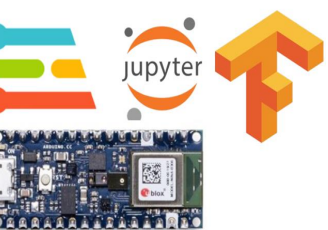


MACHINE LEARNING

❖ Machine Learning

□ Runtimes

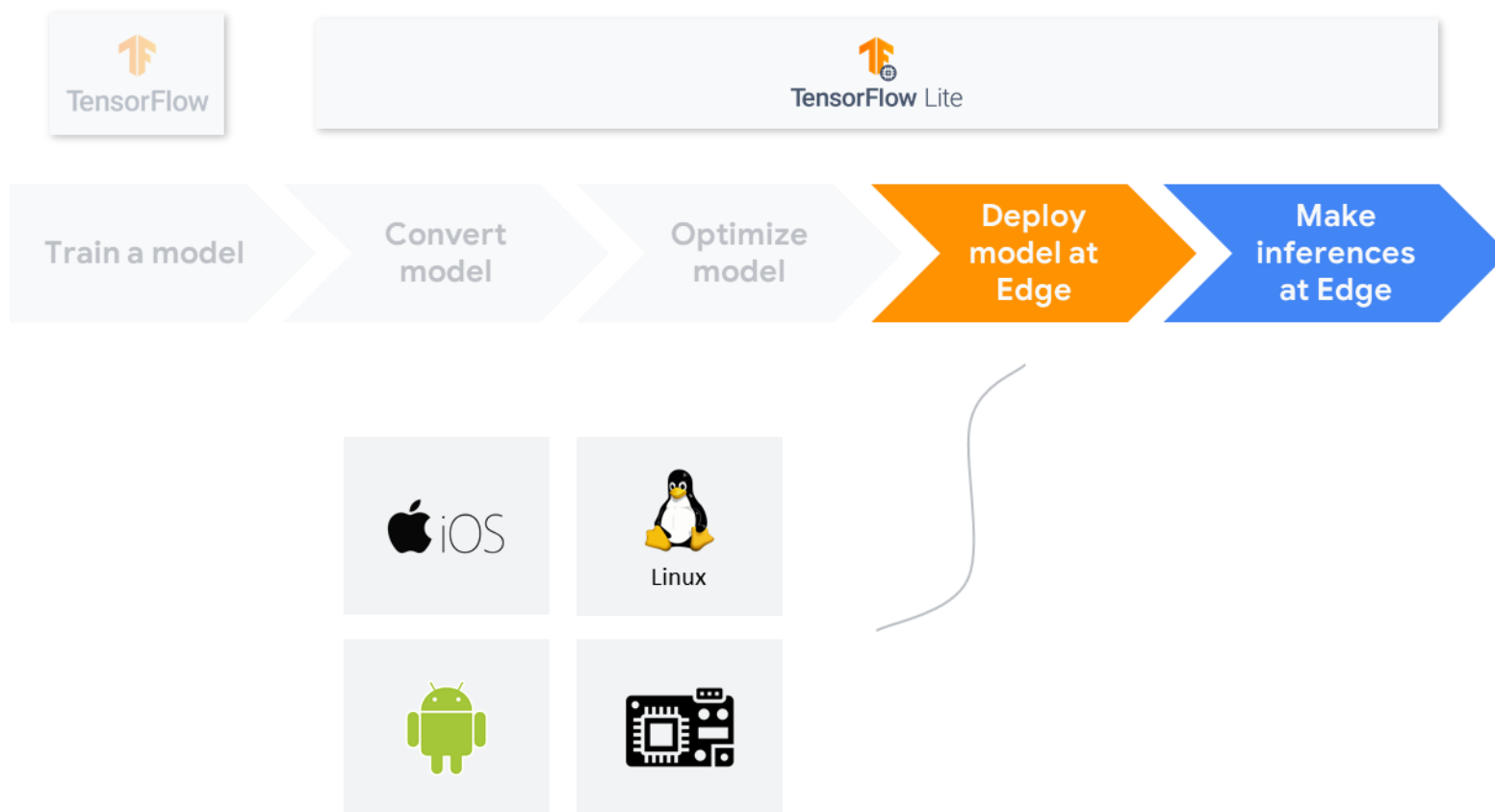


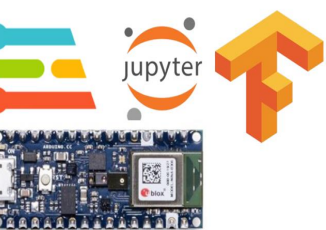


MACHINE LEARNING

❖ Machine Learning

□ Runtimes

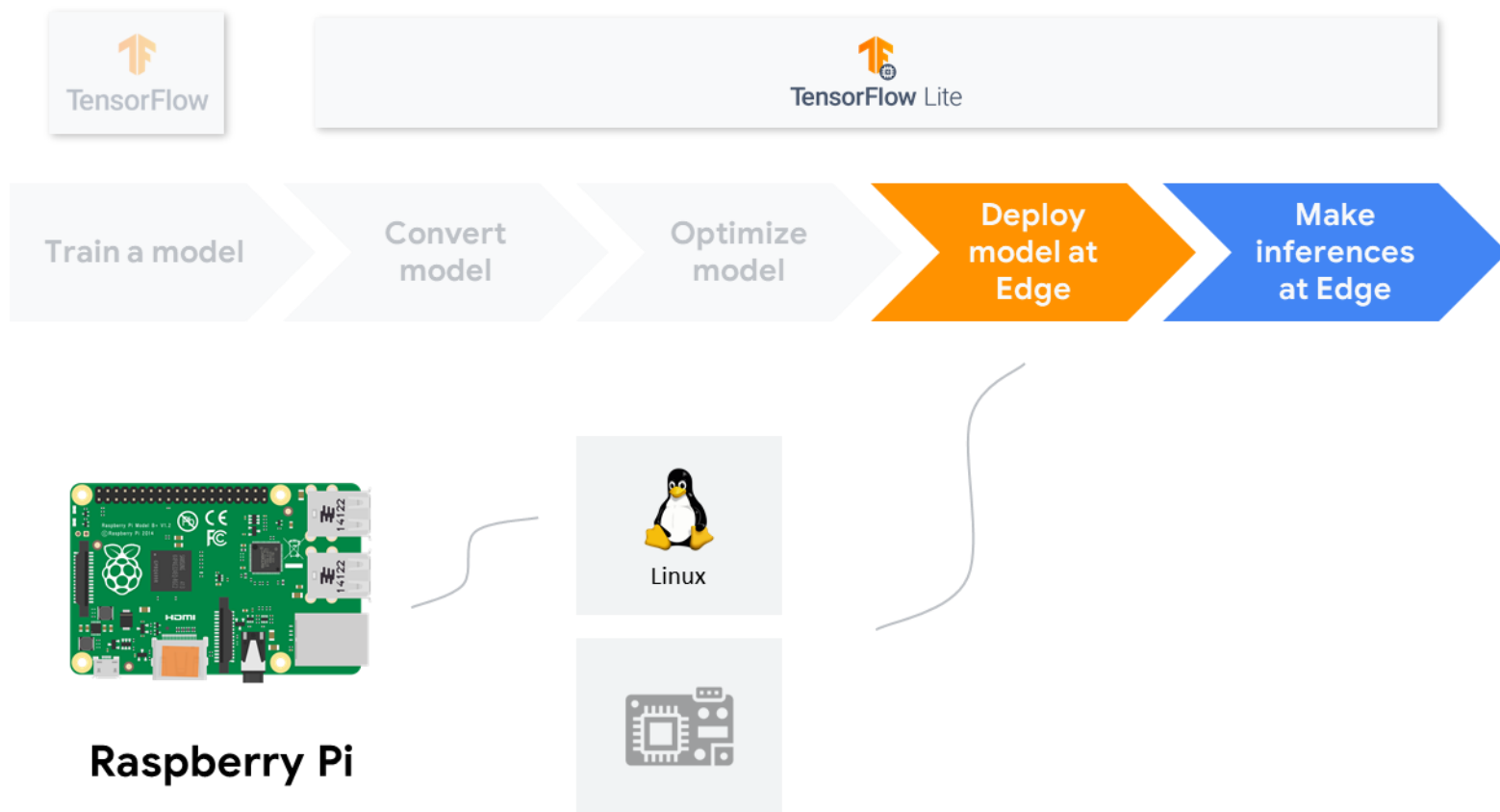


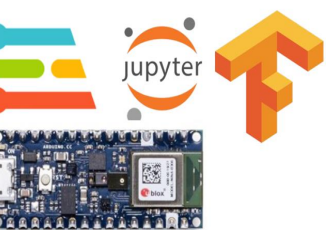


MACHINE LEARNING

❖ Machine Learning

□ Runtimes

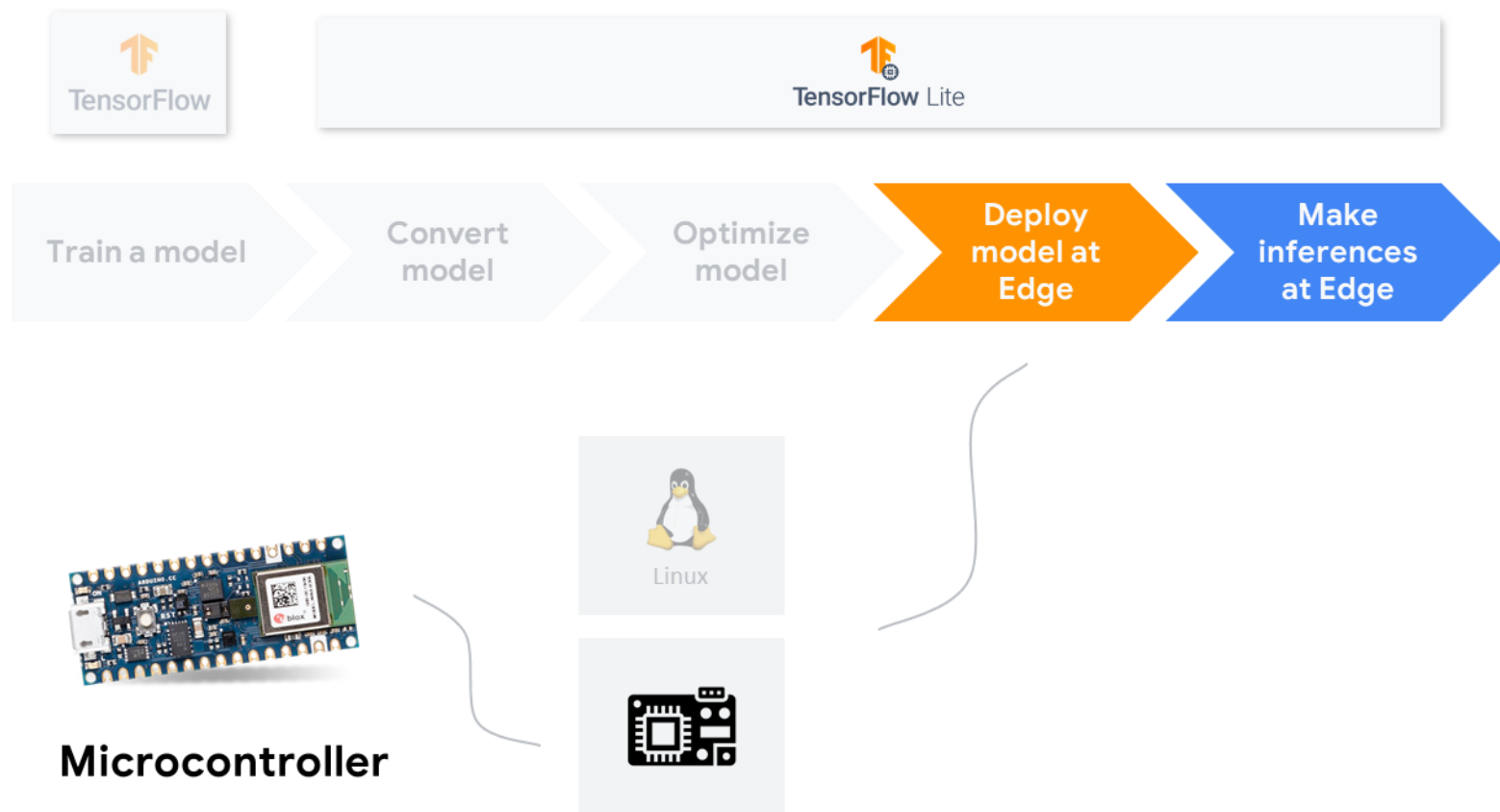


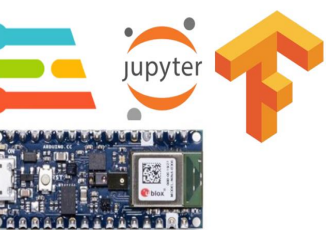


MACHINE LEARNING

❖ Machine Learning

□ Runtimes

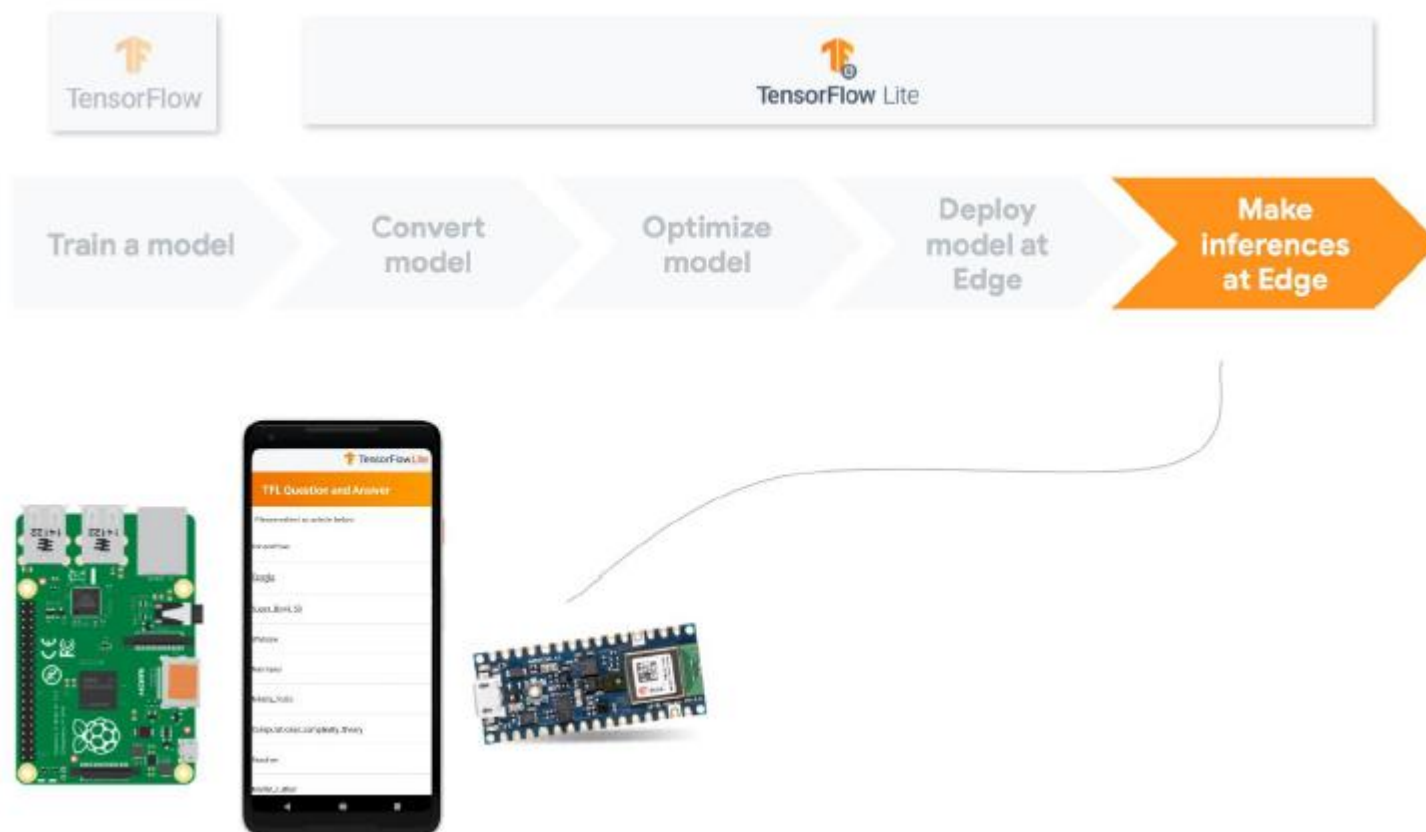


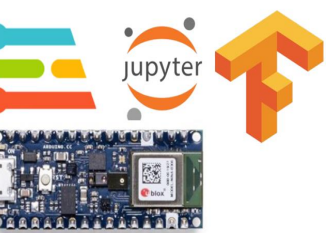


MACHINE LEARNING

❖ Machine Learning

□ Runtimes

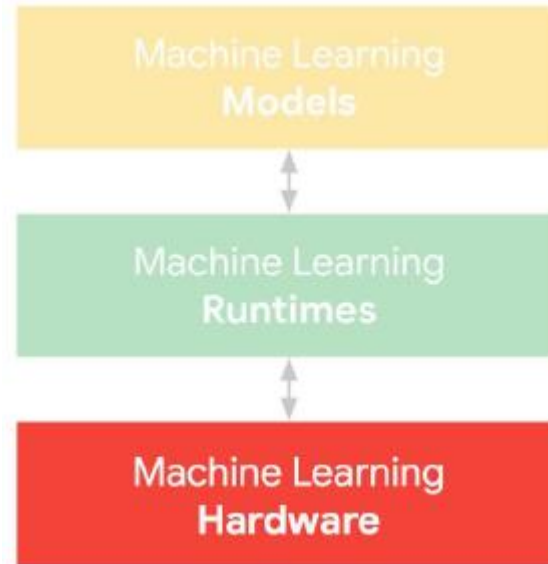


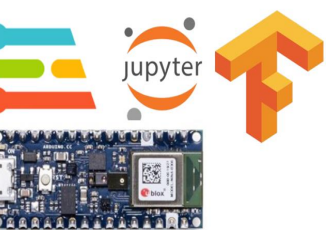


MACHINE LEARNING

❖ Machine Learning

□ Hardware





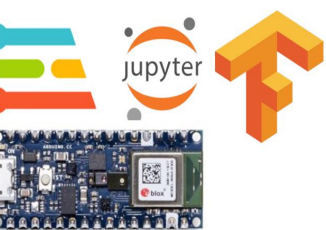
MACHINE LEARNING

❖ Machine Learning

□ Hardware

Broadest Range of ML-optimized Processing Solutions





MACHINE LEARNING

❖ Machine Learning

□ Hardware

Summary



Anomaly Detection
Sensor Classification
20 KB

KeyWord Spotting
Audio Classification
50 KB

Image
Classification
250 KB+

TinyML

EdgeML

Video
Classification
2 MB+

Object
Detection
Complex Voice
Processing
1 MB+



Rpi-Pico
(Cortex-M0+)



Arduino Nano
(Cortex-M4)



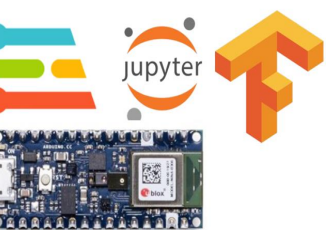
Arduino Pro
(Cortex-M7)



Raspberry Pi
(Cortex-A)



Jetson Nano
(Cortex-A + GPU)

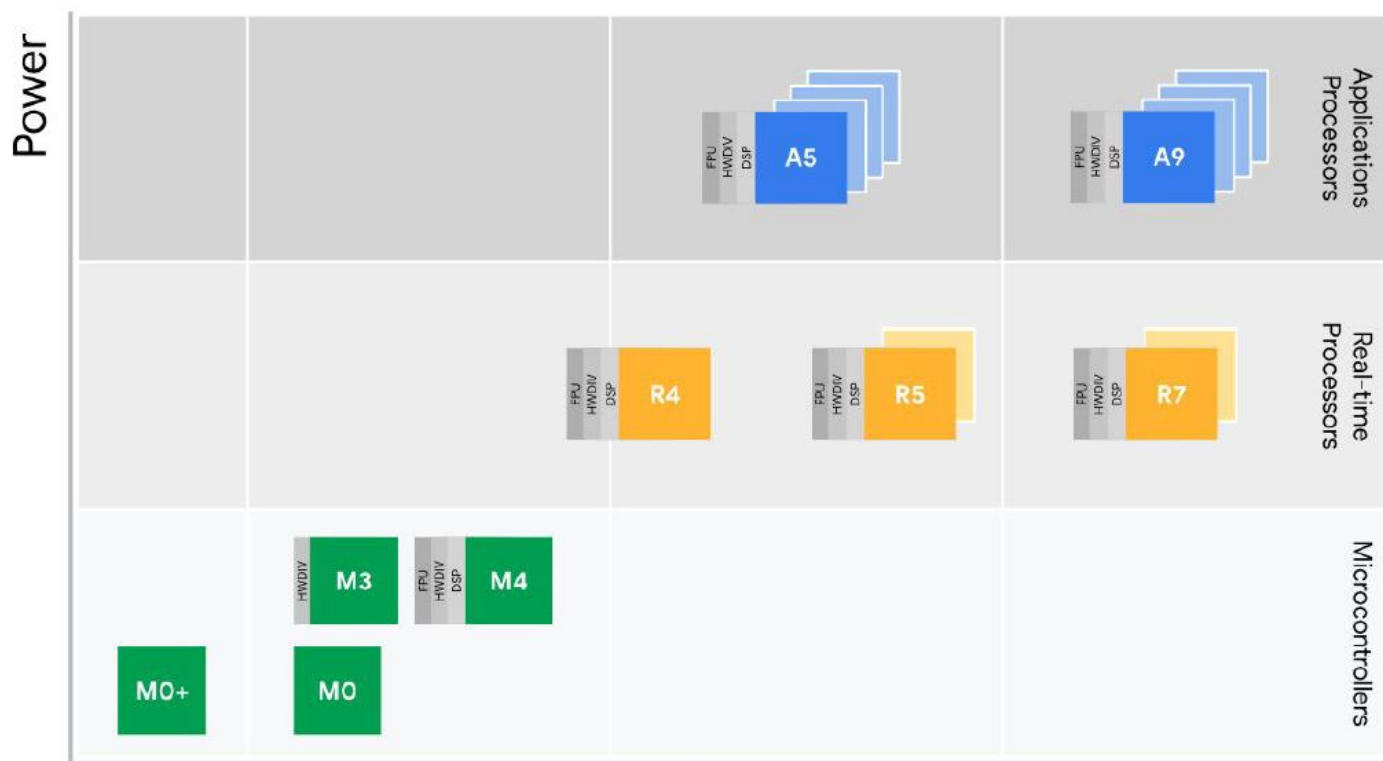


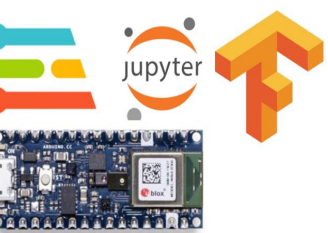
MACHINE LEARNING

❖ Machine Learning

□ Hardware

- ARM Cortex processor profiles



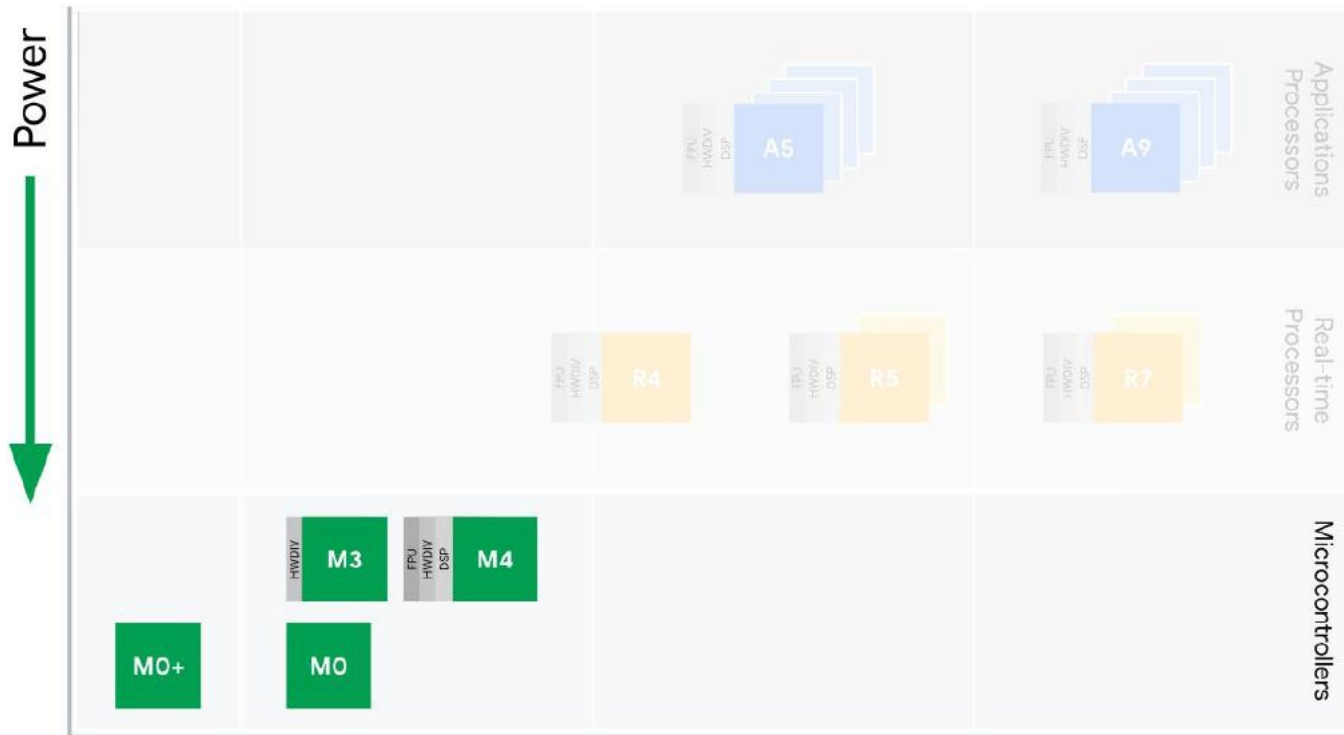


MACHINE LEARNING

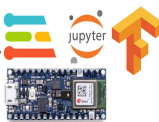
❖ Machine Learning

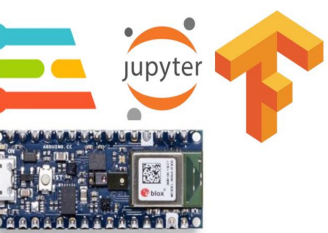
□ Hardware

- ARM Cortex processor profiles



Pipeline
Complexity



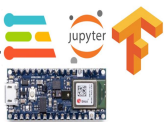
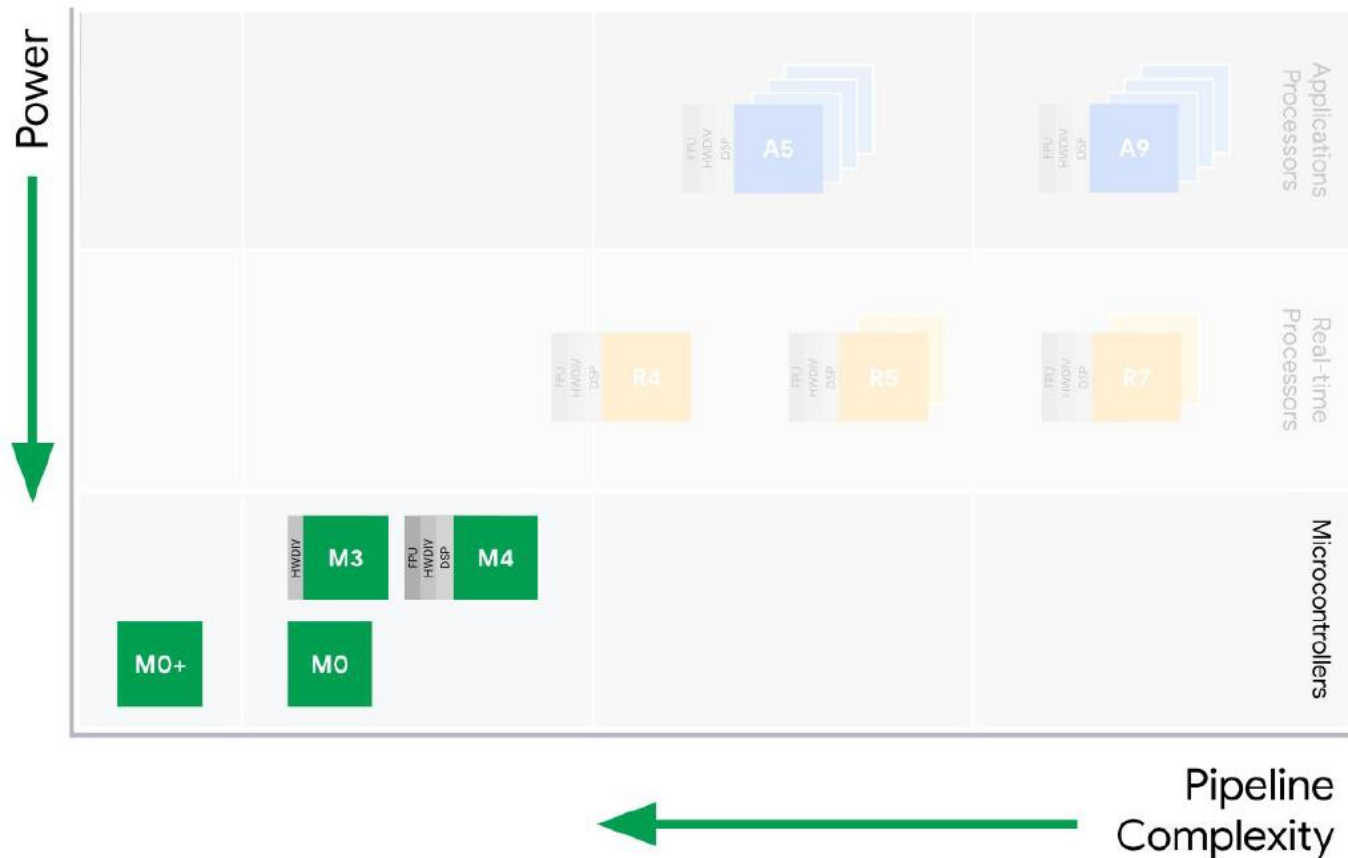


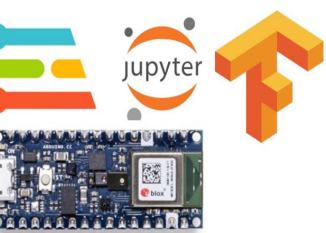
MACHINE LEARNING

❖ Machine Learning

□ Hardware

- ARM Cortex processor profiles



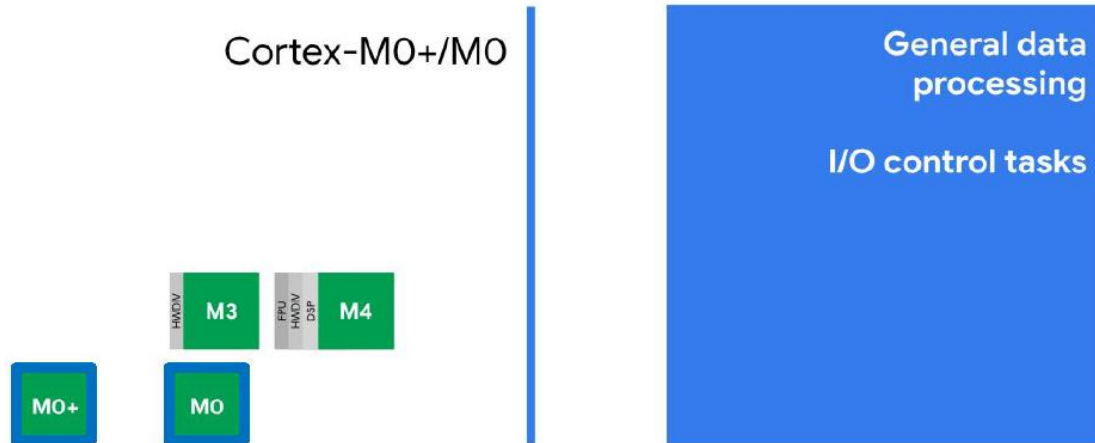


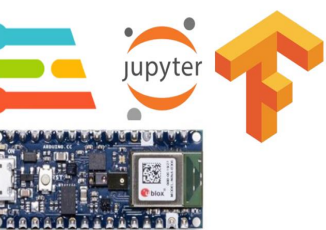
MACHINE LEARNING

❖ Machine Learning

□ Hardware

- ARM Cortex-M ISA





MACHINE LEARNING

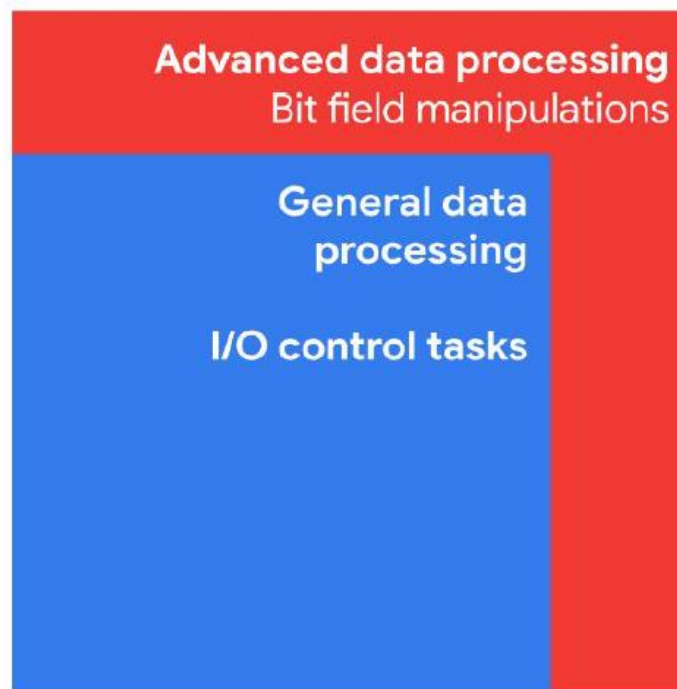
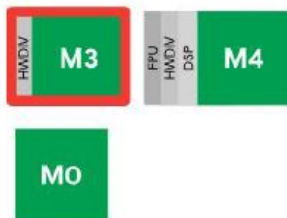
❖ Machine Learning

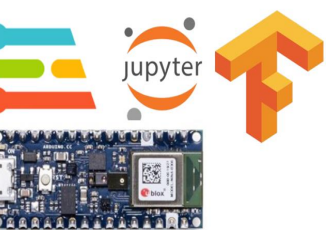
□ Hardware

- ARM Cortex-M ISA

Cortex-M3

Cortex-M0+/M0



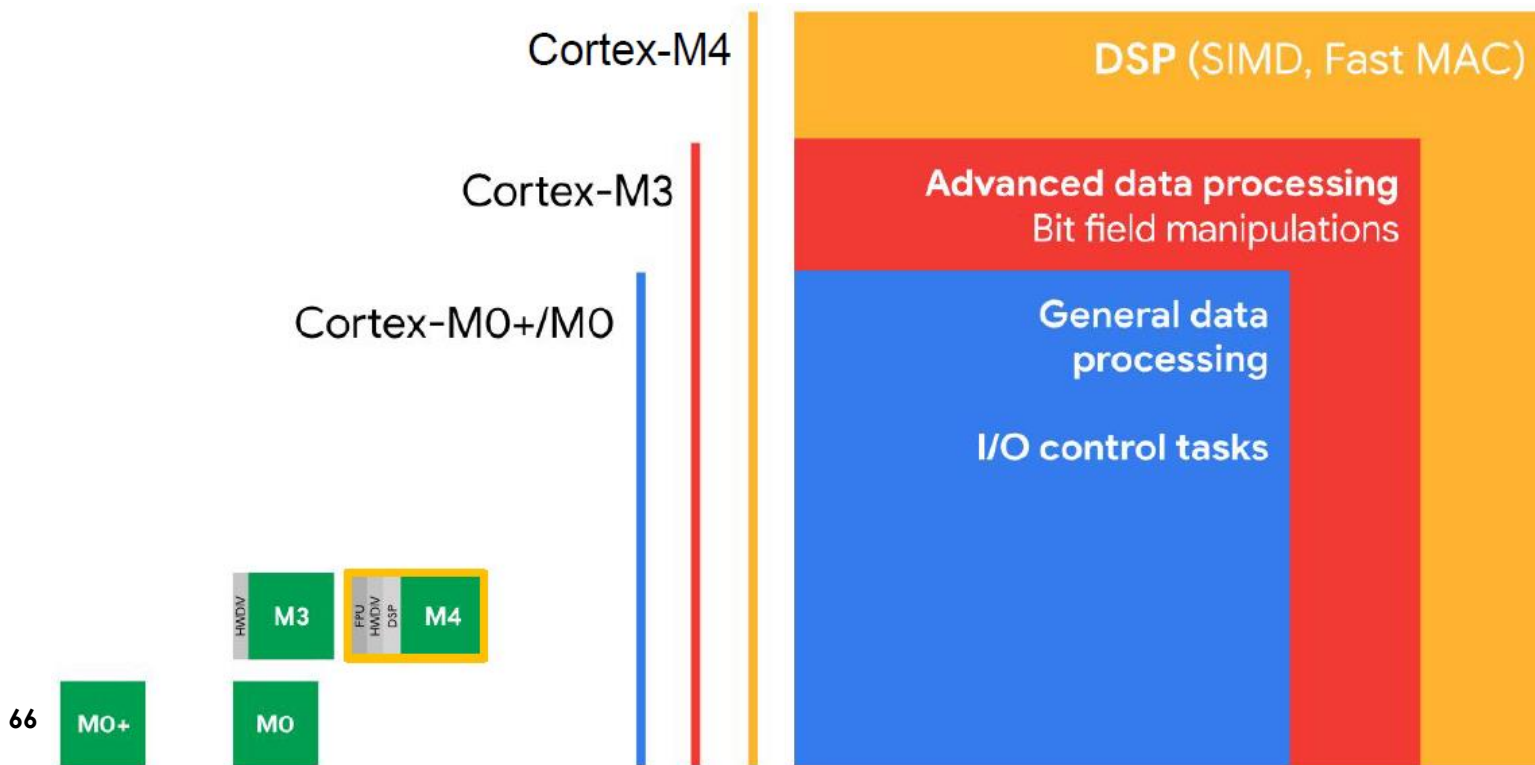


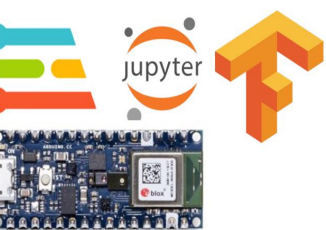
MACHINE LEARNING

❖ Machine Learning

□ Hardware

- ARM Cortex-M ISA





MACHINE LEARNING

❖ Machine Learning

□ Hardware

- ARM Cortex-M ISA

