ML on Microcontrollers

Gian Marco Iodice

Tech-lead Machine Leaning @arm

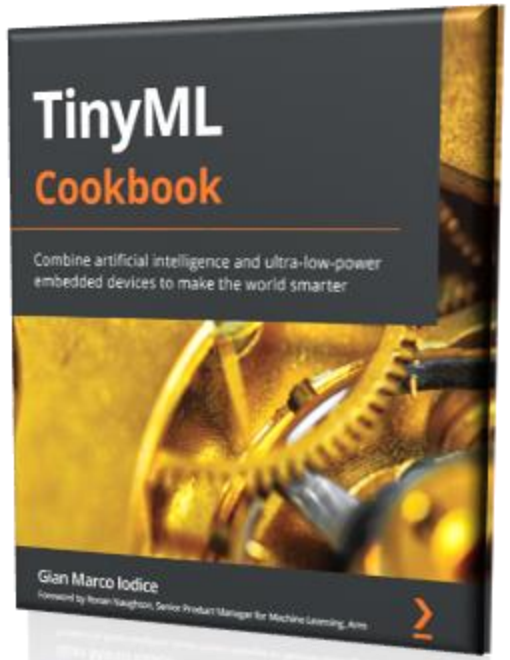
Author of the TinyML Cookbook

04 August 2022

Bio

Gian Marco Iodice is a tech lead in the Machine Learning Group at Arm (Cambridge, UK) and author of the TinyML Cookbook, published in April 2022.

At Arm, Gian Marco looks after the ML performance optimisations for the Arm Compute Library, which he co-created in 2017 to get the best performance on Arm Cortex-A CPUs and Mali GPUs. Arm Compute Library is currently the most performant library for ML on Arm, and it's deployed on billions of devices worldwide – from servers to smartphones



arm COMPUTE LIBRARY

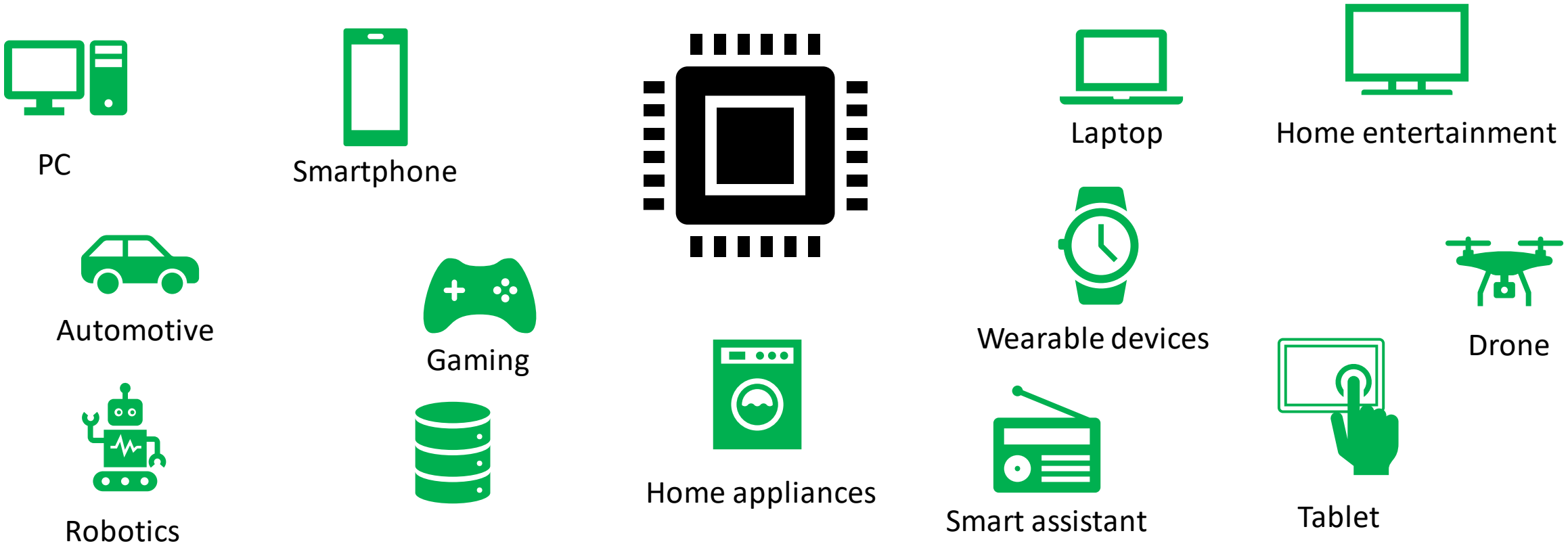
arm

Arm

Technology at the heart of where computing happens

215 Billion Devices — from Sensors to Smartphones to Servers

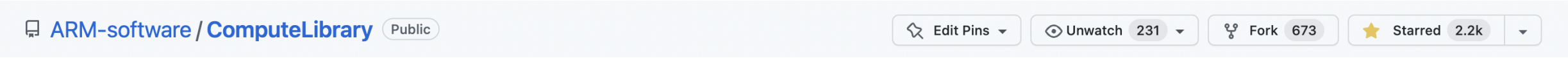
www.arm.com



Arm Compute Library

arm COMPUTE LIBRARY

https://github.com/ARM-software/ComputeLibrary



Collection of low-level machine learning functions optimized for Arm® Cortex®-A and Arm® Mali™ GPUs architectures

- Provides superior performance to other open source alternatives and immediate support for new Arm® technologies
- Open source software available under a permissive MIT license
- Over 100 machine learning functions for CPU and GPU
- Multiple convolution algorithms (GeMM, Winograd, FFT, Direct and indirect-GeMM)
- Support for multiple data types: FP32, FP16, INT8, UINT8, BFLOAT16

arm

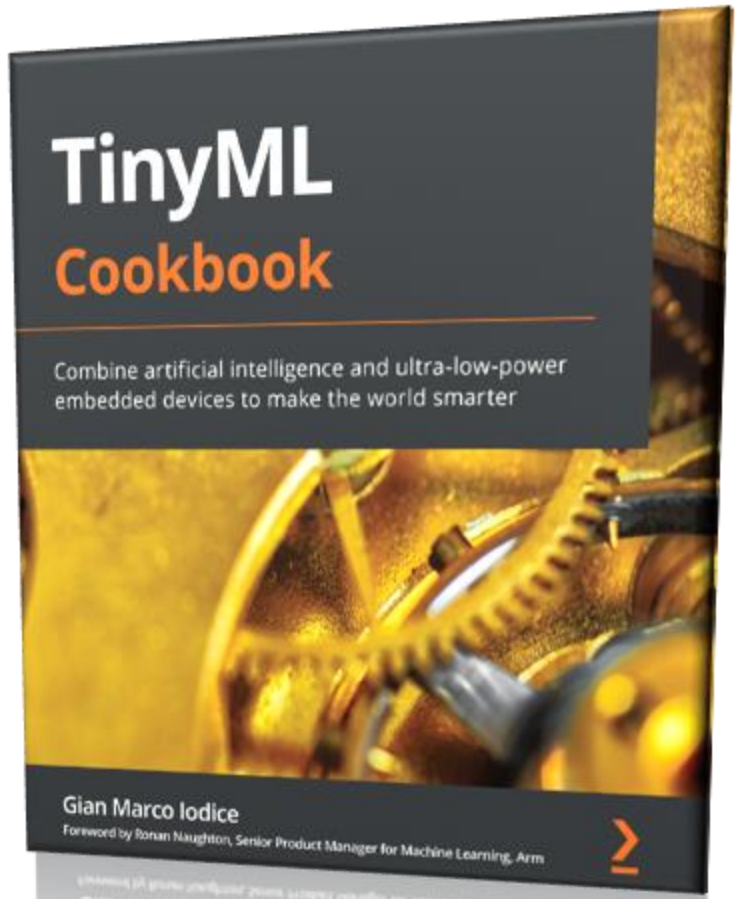
TinyML Cookbook

Motivation

Demonstrate how easy TinyML is for everyone, even for those with no or little familiarity with embedded programming

Target audience

ML developers/engineers interested in developing ML applications on microcontrollers through practical examples.
However, embedded developers who have some basic understanding of ML can also benefit from this book



https://amz.run/5QmC

Free book giveaway!!! Only one day left!

How to join the book giveaway

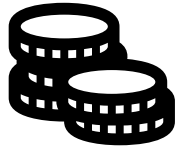
- Answer the following question in my post on LinkedIn

- *What do you hope to learn from this book?*

Why TinyML

Bring intelligence to objects around us with a focus on power consumption, data privacy, and cost





Make AI ubiquitous for good

What Good Means

Make positive contributions to the United Nations Sustainable Development Goals

https://www.tinyml.org/event/tinyml-for-good/

What is TinyML

TinyML is the set of technologies in ML and embedded systems to make use of smart applications on extremely low-power devices. Generally, these devices have limited memory and computational capabilities, but they can sense the physical environment through sensors and act based on the decisions taken by ML algorithms.*

Ingredients

Characteristics of low-power devices

System input

*TinyML Cookbook, Gian Marco Iodice – April 2022

arm

What is TinyML

Level of computing on top of sensors that allows smartness in a minimal intrusive way

Alasdair Allan, Head of documentation, Raspberry Pi

TinyML Cookbook Release party

20th of April 2020

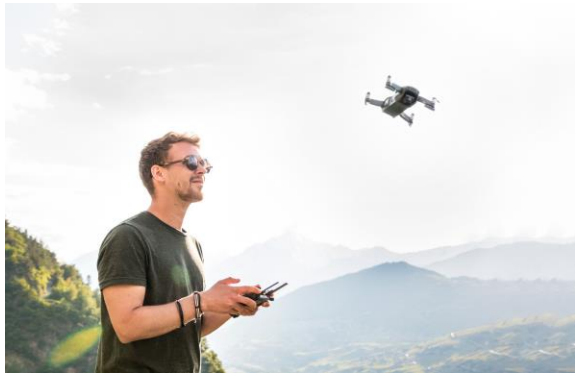
https://youtu.be/g6UcuCmAgwg

- *Alessandro Grande, Edge Impulse*
- *Gian Marco Iodice, Arm*
- *Allan Alasdair, Raspberry Pi*
- *Massimo Banzi, Arduino*

TinyML Applications 1of2

TinyML finds its natural home wherever a power supply from the mains is impossible or complex to have, and the application must operate with a battery for as long as possible

Bring intelligence to battery-powered devices



TinyML Applications 2of2

Battery-powered solutions are not limited to consumer electronics only…

There are scenarios where we might need devices to monitor environments. For example, we may consider deploying battery-powered devices running ML in a forest to detect fires and prevent fires from spreading over a large area

Centralized Vs Distributed Smart Systems

Centralized TinyML applications

No communication with other devices

Smart watches

- *Handwriting recognition*
- *Wake-up words (“Hey, Google!”, “Siri”,)*
- *Activity recognition*

Smart assistants / Phones

- *Wake-up words (“Hey, Google!”, “Siri”,)*
- *Activity recognition*

Distributed TinyML applications

We may need to communicate with other devices (WSN*)

- *Fire detection*
- *Endangered species conservation*

Agriculture

- *Autonomous irrigation system*
- *Precision/sustainable farming*

*Wireless Sensor Network

arm

TinyML Foundation

tinyML Foundation (www.tinyml.org) is a non-profit professional organization supporting and connecting the TinyML world.

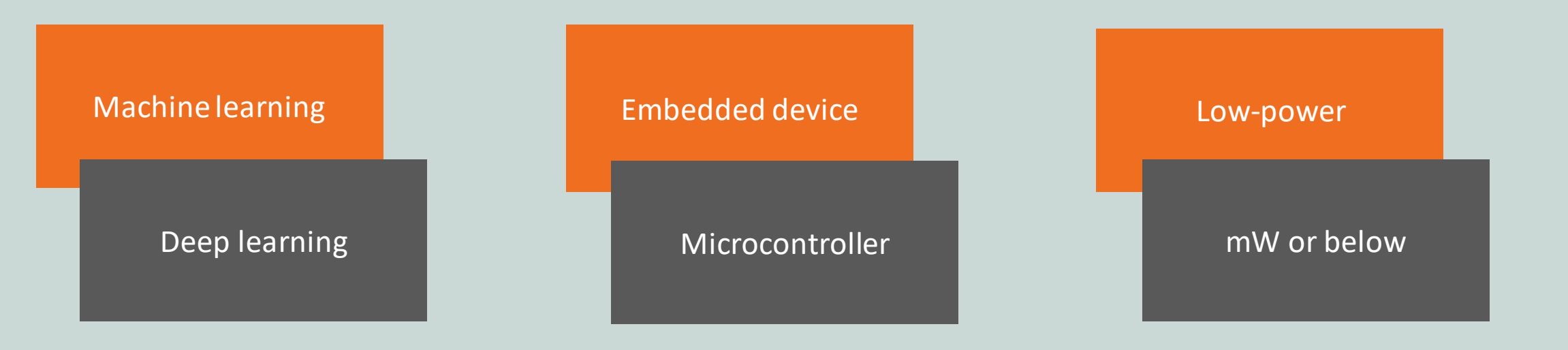
To do this, tinyML Foundation is growing a diverse community worldwide between hardware, software, system engineers, scientists, designers, product managers, and businesspeople.

With several Meetup (https://www.meetup.com) groups in different countries, you can join a TinyML one near you for free


arm
Let's explore the ingredients of TinyML on microcontrollers

TinyML on Microcontrollers

Fast growing field at the intersection of machine learning (ML) and embedded systems to enable smart applications on extremely low-powered devices



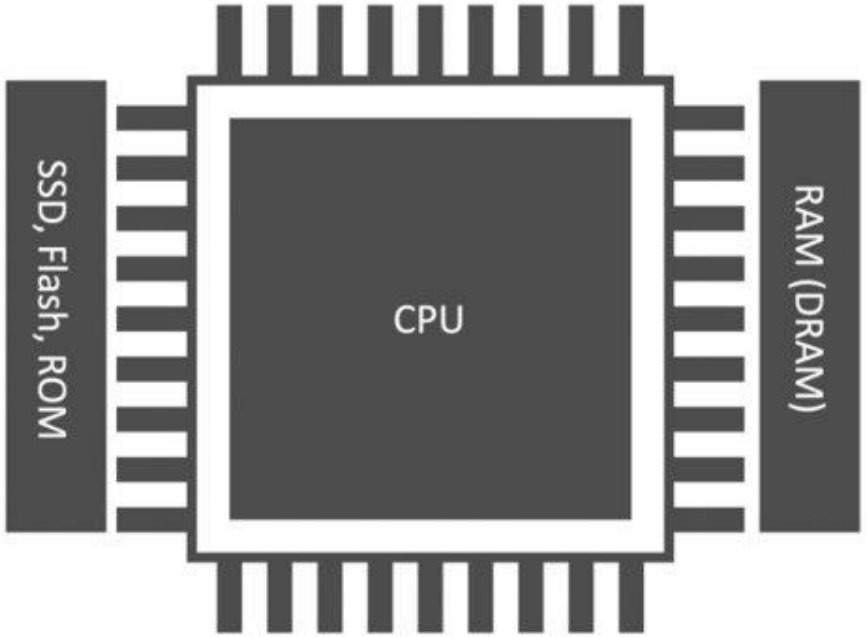
Microcontrollers

A microcontroller is a full-fledged computer because it has a processor (CPU), a memory system (for example, RAM or ROM), and some peripherals

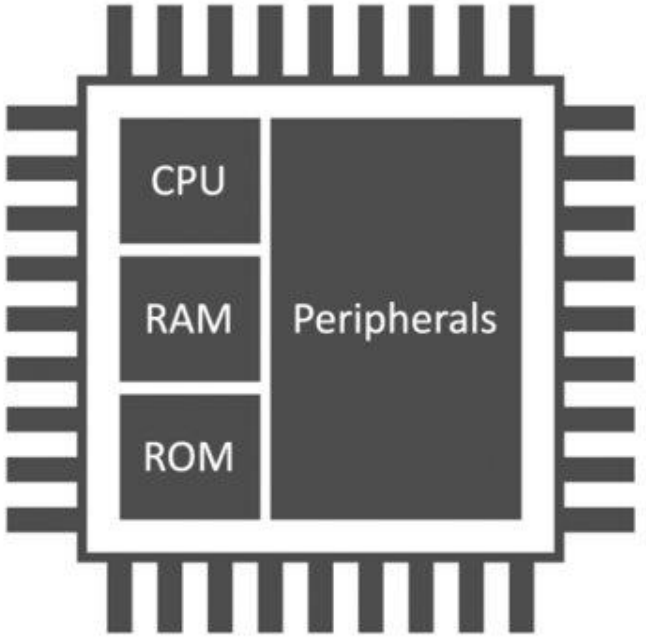
Question

However, how does it differ from a microprocessor?

Answer



Microprocessor



Microcontroller

Microprocessor versus microcontroller - 1of2

The target applications influence their architectural design choices

- **Dynamic** (for example, can change with user interaction or time)
- **General-purpose**
- **Compute intensive**

Microcontroller

- **Tasks are single purpose and repetitive**
  - The device does not require strict re-programmability
- **Tasks may have real-time constraints**
  - The device must be latency predictable
- **Tasks may be battery-powered**
  - RAM and ROM must be on-chip
  - Low clock frequency
  - Reduced computational capabilities (e.g., just integer arithmetic)

Microprocessor versus microcontroller - 2of2

Feature	Microprocessor	Microcontroller
Application	General-purpose	Single-purpose
CPU arithmetic	It can perform heavy mathematical calculations in floating-point or double precision	Mainly integer arithmetic
RAM	A few GB	A few hundred KB
ROM (or hard-drive)	GB or TB	KB or MB
Clock frequency	GHz	MHz
Power consumption	W	mW or below
Operating System (OS)	Required	Not strictly required
Cost	From ten to hundreds of dollars	From a few cents (low-end) to a few dollars (high-end)

Memory Architecture

In the microcontroller context, we physically dedicate two separate memories for the instructions (program) and data:

- Program memory (ROM)
  - Non-volatile read-only memory reserved for the program to execute
  - It can store CONSTANT data

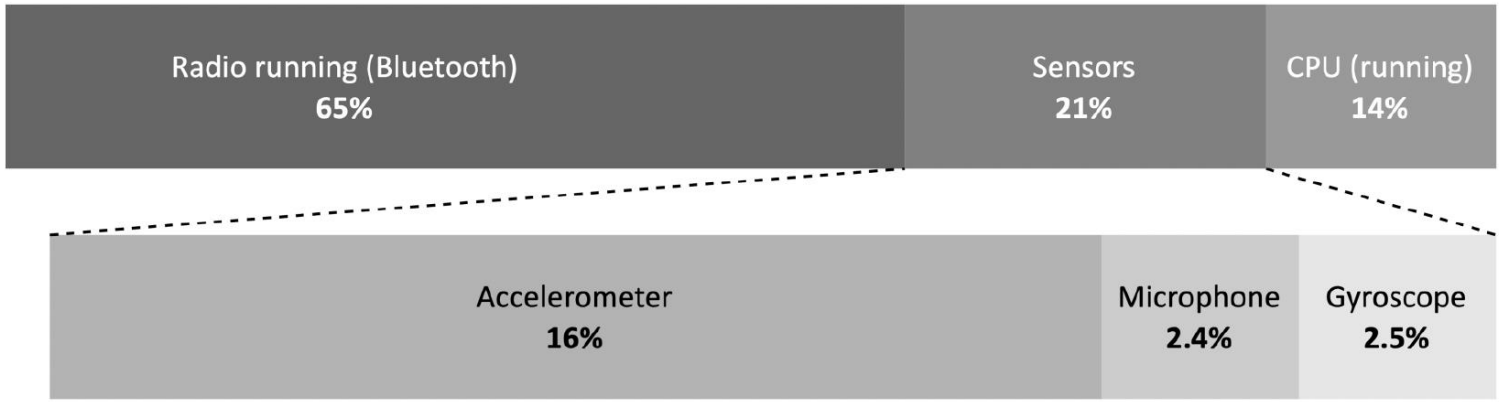
- Data memory (RAM/SRAM)
  - Volatile memory reserved to store/read temporary data

Why ML on Microcontrollers

- Popularity
  - They are everywhere (e.g., automotive, consumer electronics, kitchen appliances, healthcare,…)
  - With the rise of IoT, 28.1 billion microcontrollers sold in 2018! (Note: Smartphone 1.8 billion and 67 million)
- Inexpensive
  - From a few cents to a few dollars
- Easy to program
  - You can write programs in C (or Python nowadays!)
  - IDE can be FREE and web-based
- Powerful enough to run ML

Why Run ML Locally

- Reducing latency
  - Sending data back and forth from the Cloud is not instant!
- Reducing power consumption
  - Sending/receiving data to and from the Cloud is not power efficient



- Privacy

Question

Suppose you have a processing task and you have the option to execute it on two different processors. These processors have the following power consumptions:

Processing unit	Power consumption
PU1	12
PU2	3

What processor would you use to execute the task?

Answer

Although PU1 has higher (4x) power consumption than PU2, this does not imply that PU1 is less energy-efficient. On the contrary, PU1 could be more computationally performant than PU2 (for example, 8x), making it the best choice from an energy perspective, as shown in the following formulas:

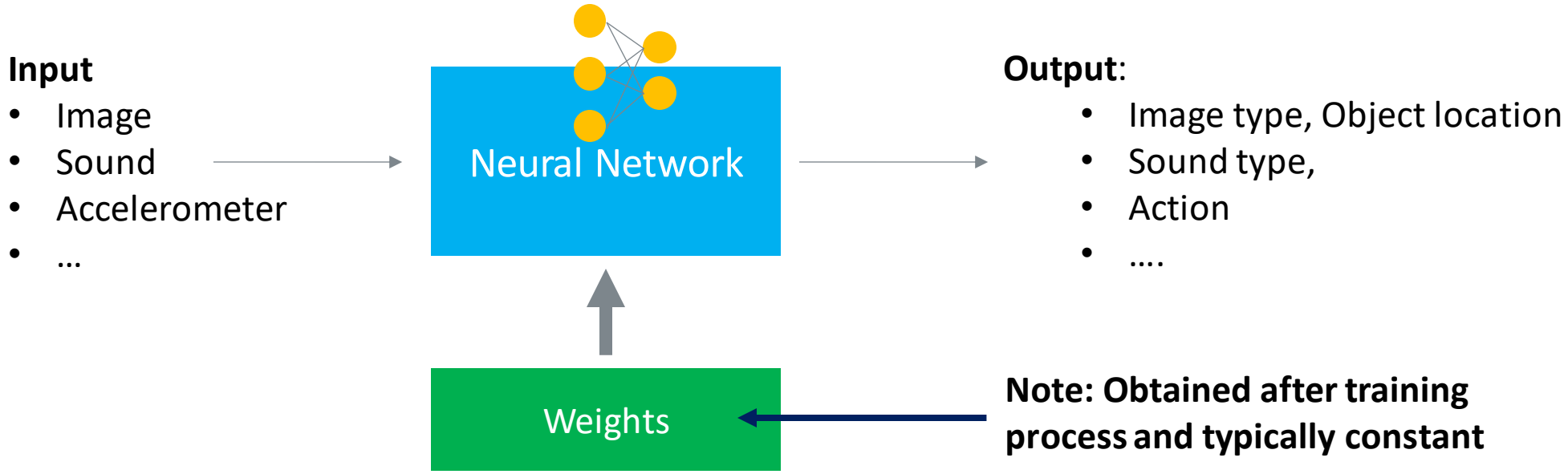
$$E_{PU1} = 12 \cdot T_1$$

$$E_{PU2} = 3 \cdot T_2 = 3 \cdot 8 \cdot T_1 = 24 \cdot T_1$$

Machine Learning (ML)

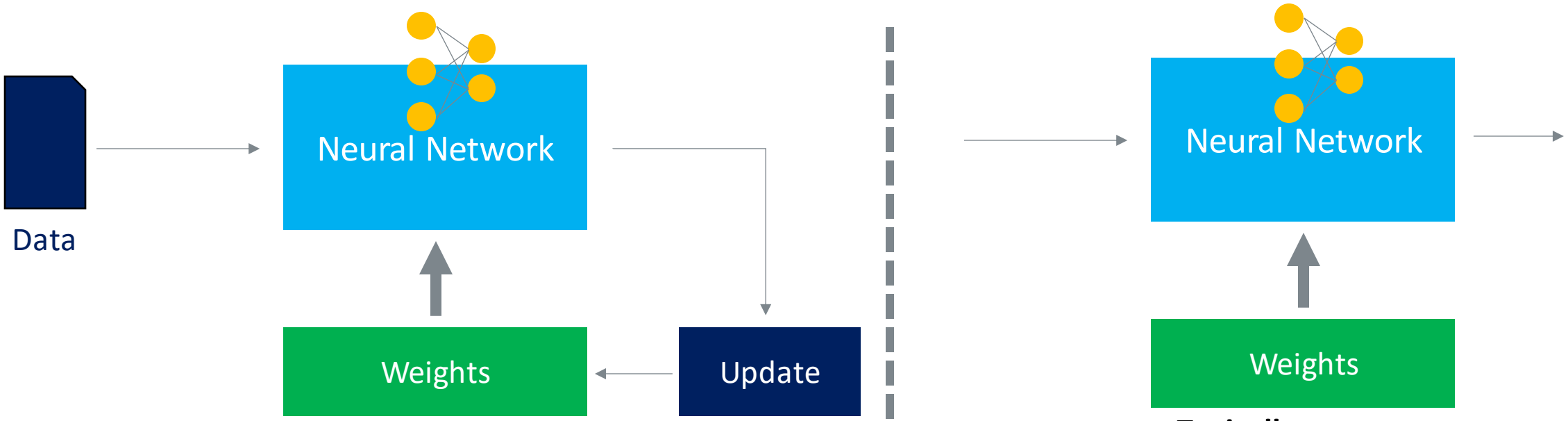
ML: it is the study of algorithms capable to learn automatically from experience and data how to behave

Neural Network is a common ML algorithm:



Training Versus Deployment

The model is only as good as the data used for training



Training step

Deployment step

Question

Where can we store the weights of our ML model in the microcontroller?

Answer

Depends on whether the model has constant weights. If the weights are constant, so do not change during inference, it is more efficient to store them in program memory for the following reasons:

- *Program memory has more capacity than SRAM*
- *It reduces memory pressure on the SRAM since other functions require storing variables or chunks of memory at runtime*

The Deep Learning Era

From feature engineering to convolutional neural networks (CNN)

Feature Extraction

Neural Network

Dog

Old ML fashion

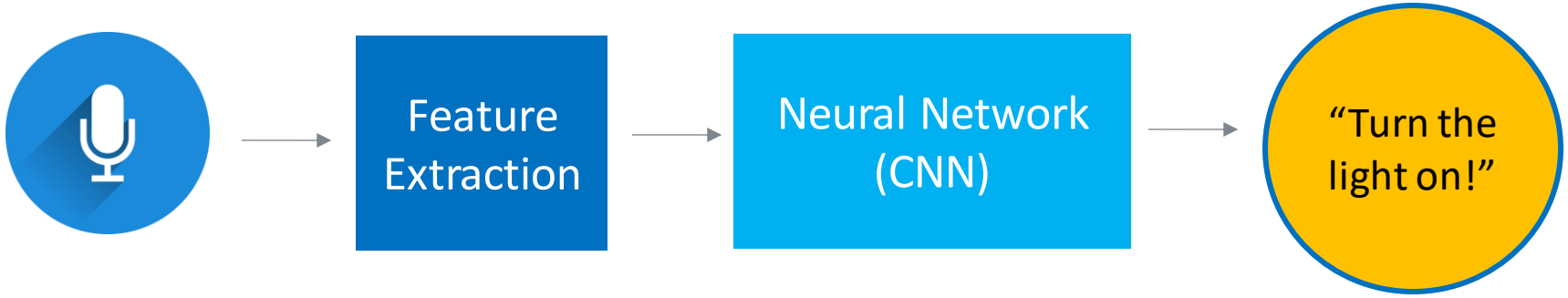
Neural Network (CNN)

Dog

Saint bernard

Deep learning

The Deep Learning Era



Old ML fashion

It could be tailored in specific situations
(e.g., sound recognition, gesture recognition)

How it Looks a TinyML Program

Pseudo C-code

```
load_model ();
initialize_model_memory();
data = acquire_data();
input = prepare_input(data);
output = run_model(input);
// Code to interpret the output result
```

→ We load the ML to execute

→ Allocate the memory required by the model

→ Acquire the data from the sensor/s

→ Prepare the input required by the model

→ Execute the model

arm

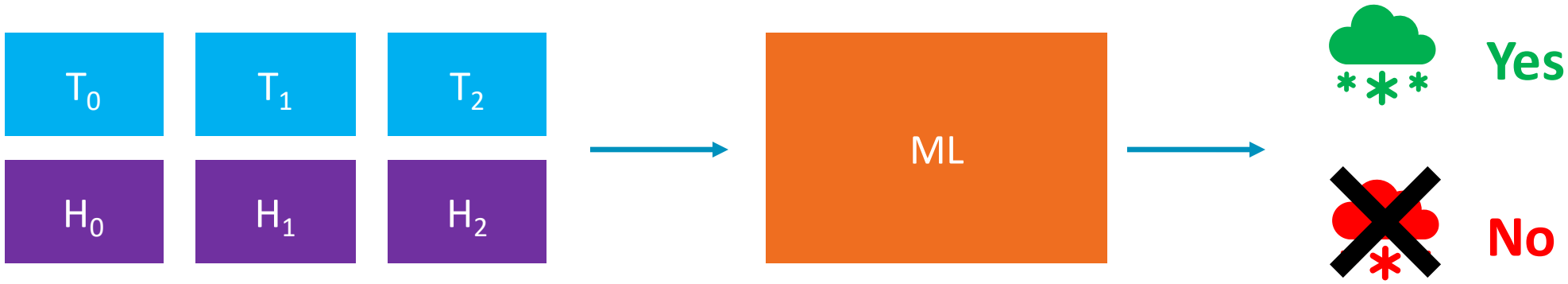
Building a Weather Station with TFLu

Based on the project of Chapter 3 of TinyML Cookbook

https://github.com/PacktPublishing/TinyML Cookbook/tree/main/Chapter03

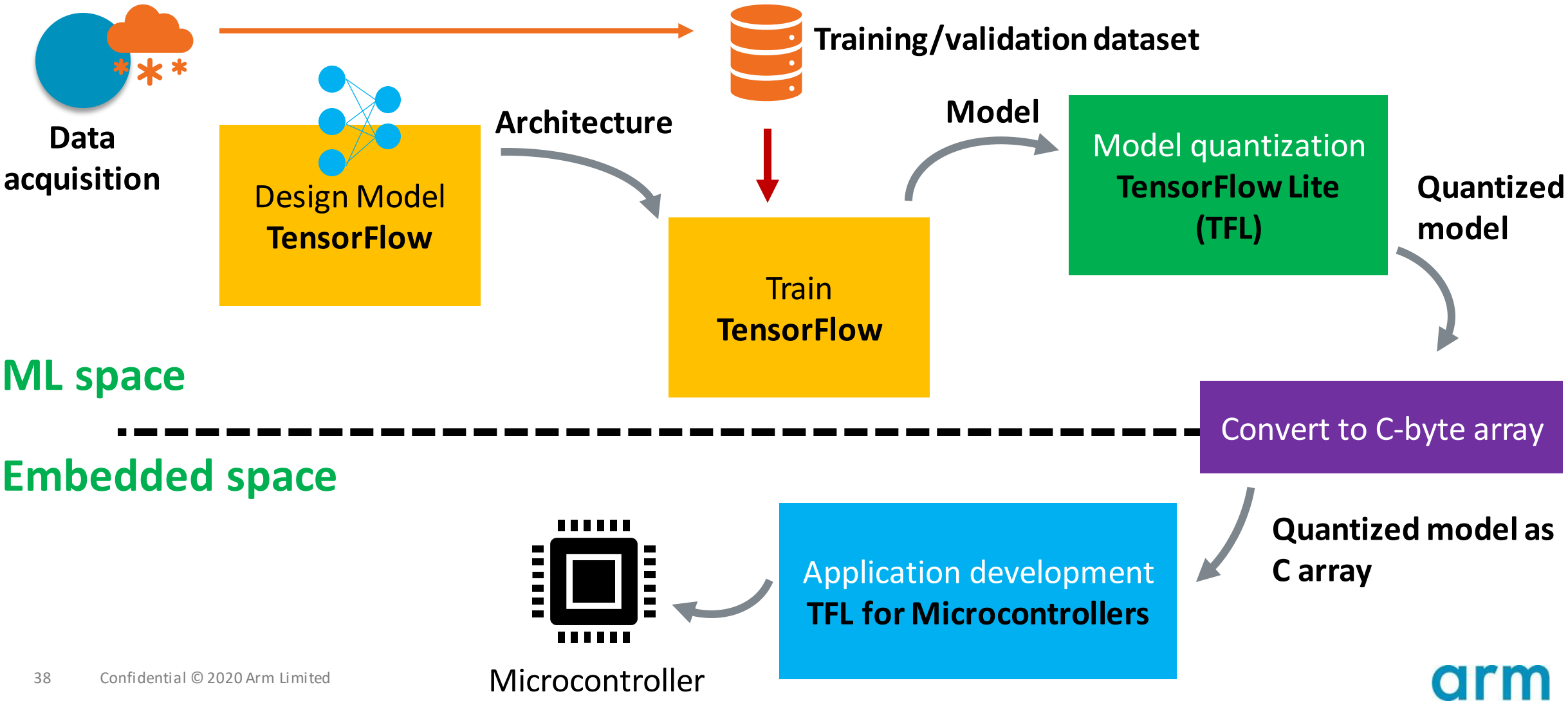
How

Use the temperature and humidity of the last three hours to forecast the snow

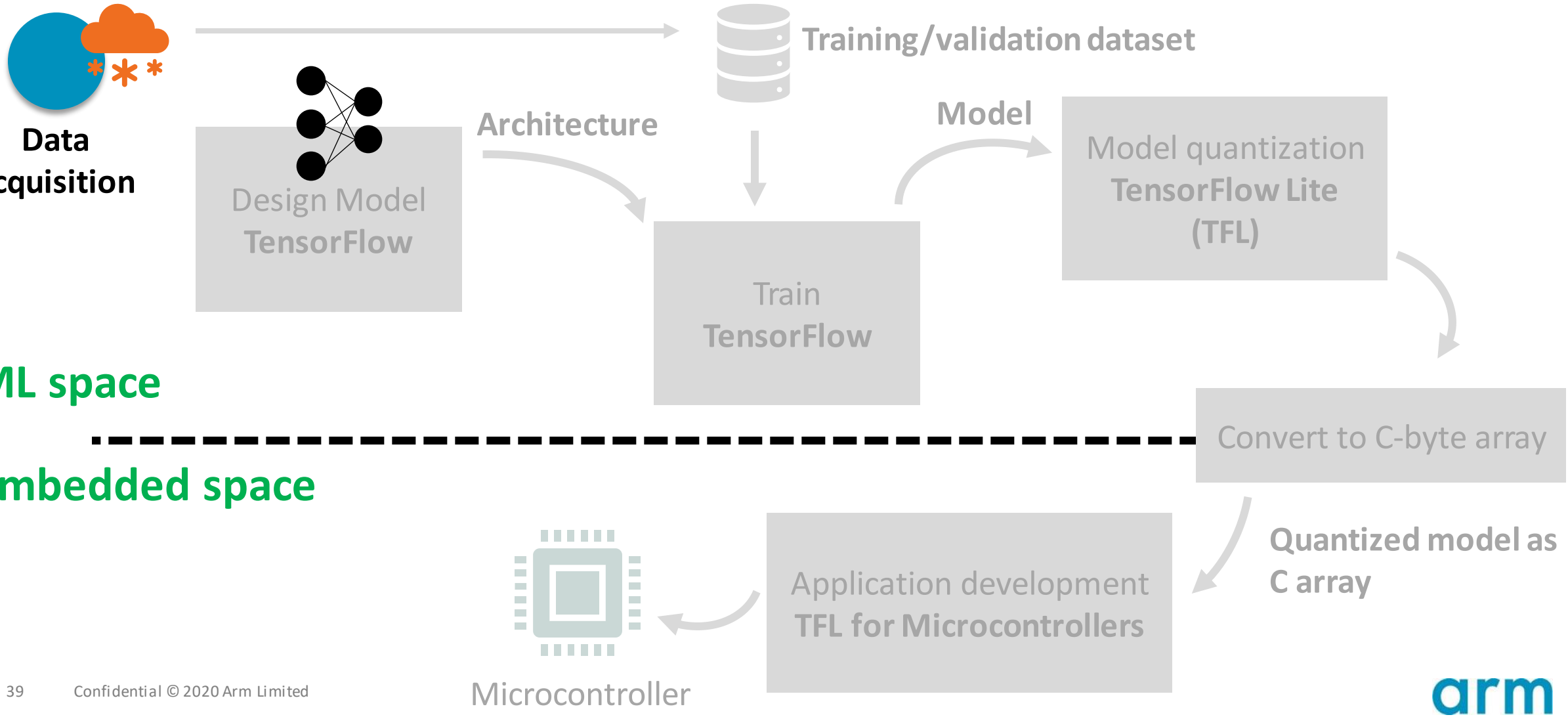


T = Temperature
H = Humidity

Workflow



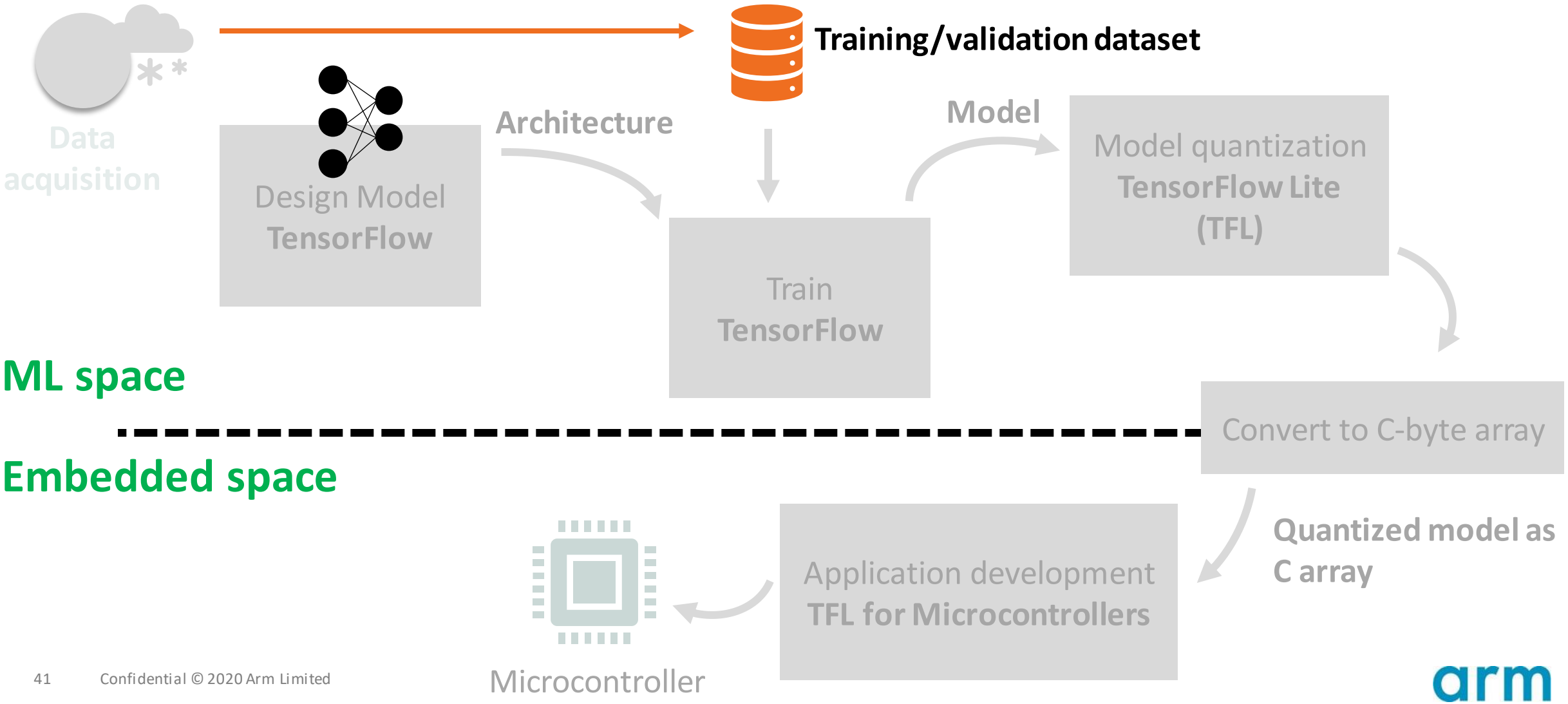
Step 1: Importing Weather Data



Step 1: What to Do

- Collect hourly historical weather data from WorldWeatherOnline from a location where it snows regularly (for example, Canazei - Italy) using Python

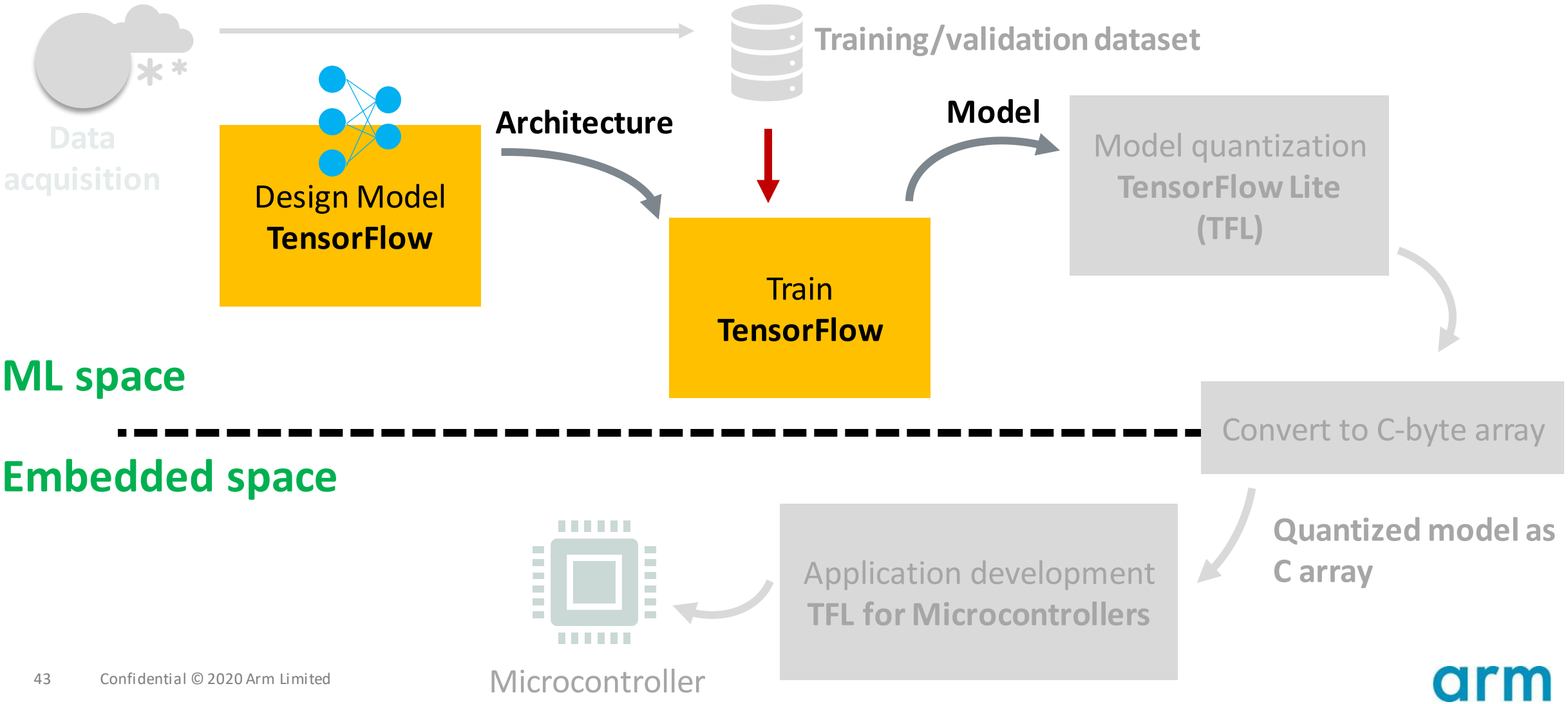
Step 2: Preparing the Dataset



Step 2: What to Do

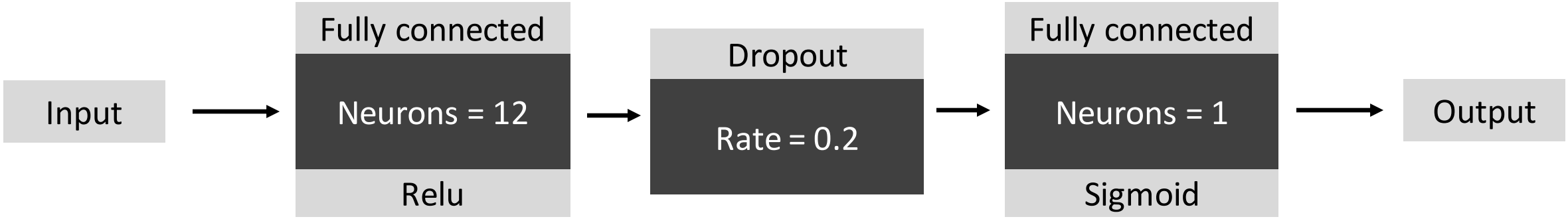
- Extract the temperature, humidity, and snowfall data
- Balance the dataset
- Scale the input features
- Prepare the dataset

Step 3: Design and Train the ML Model



Step 3: What to Do

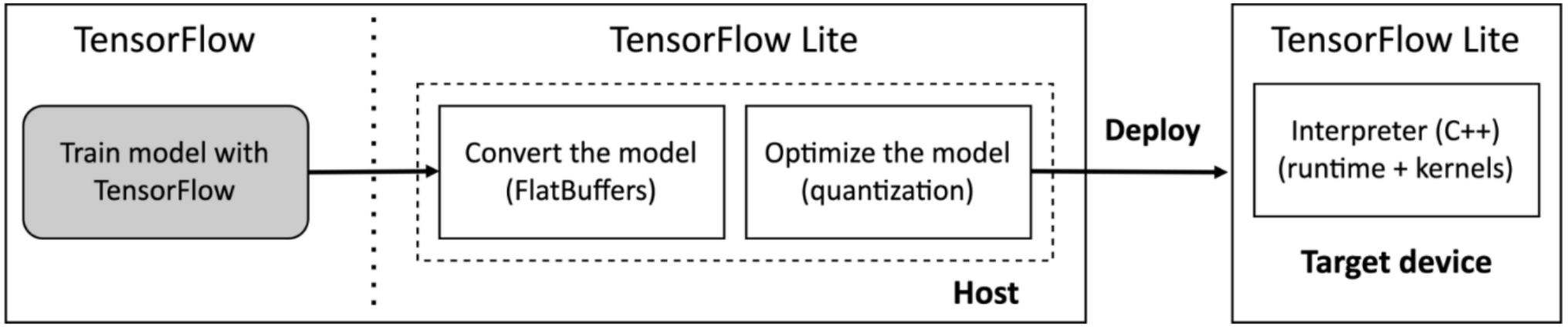
- Design the model with TensorFlow



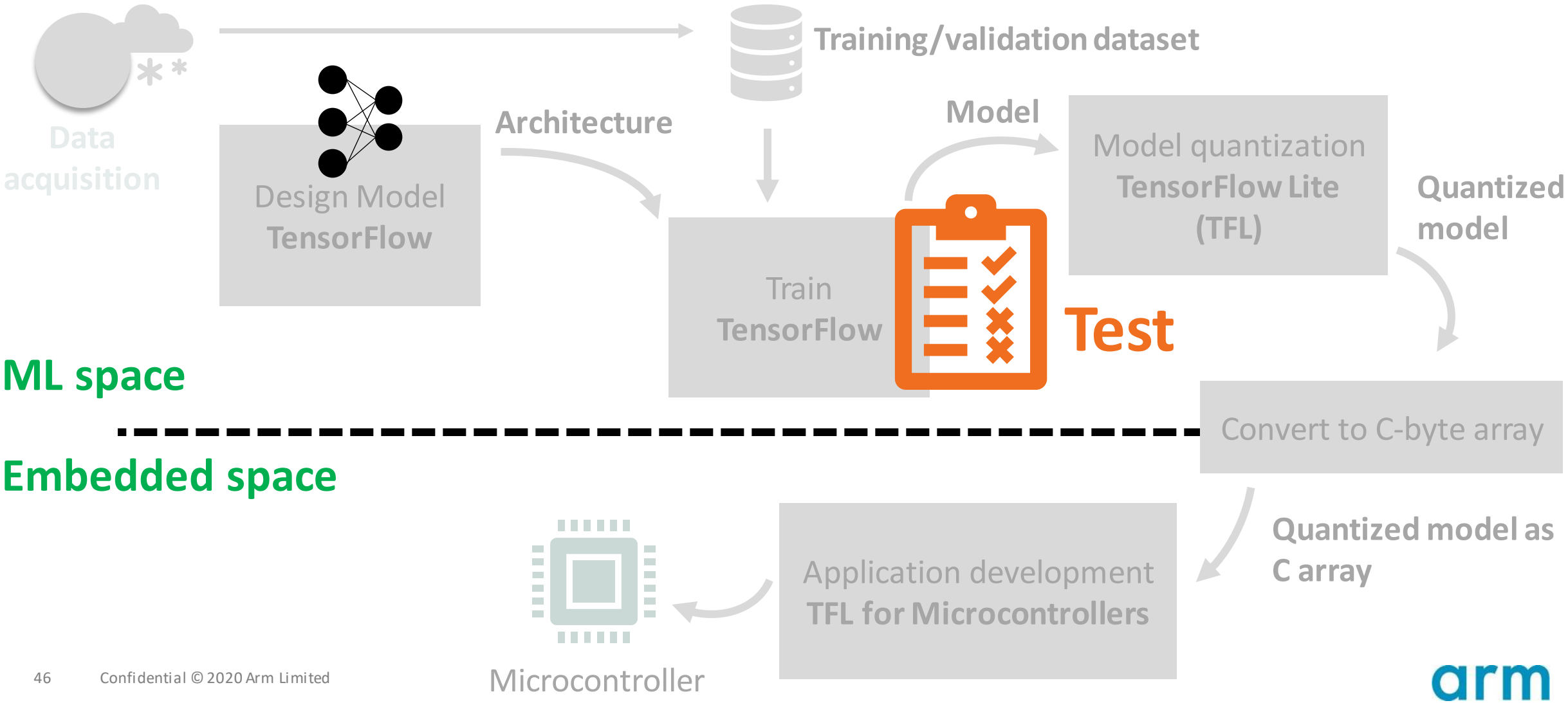
- Train the model with TensorFlow

What is TensorFlow

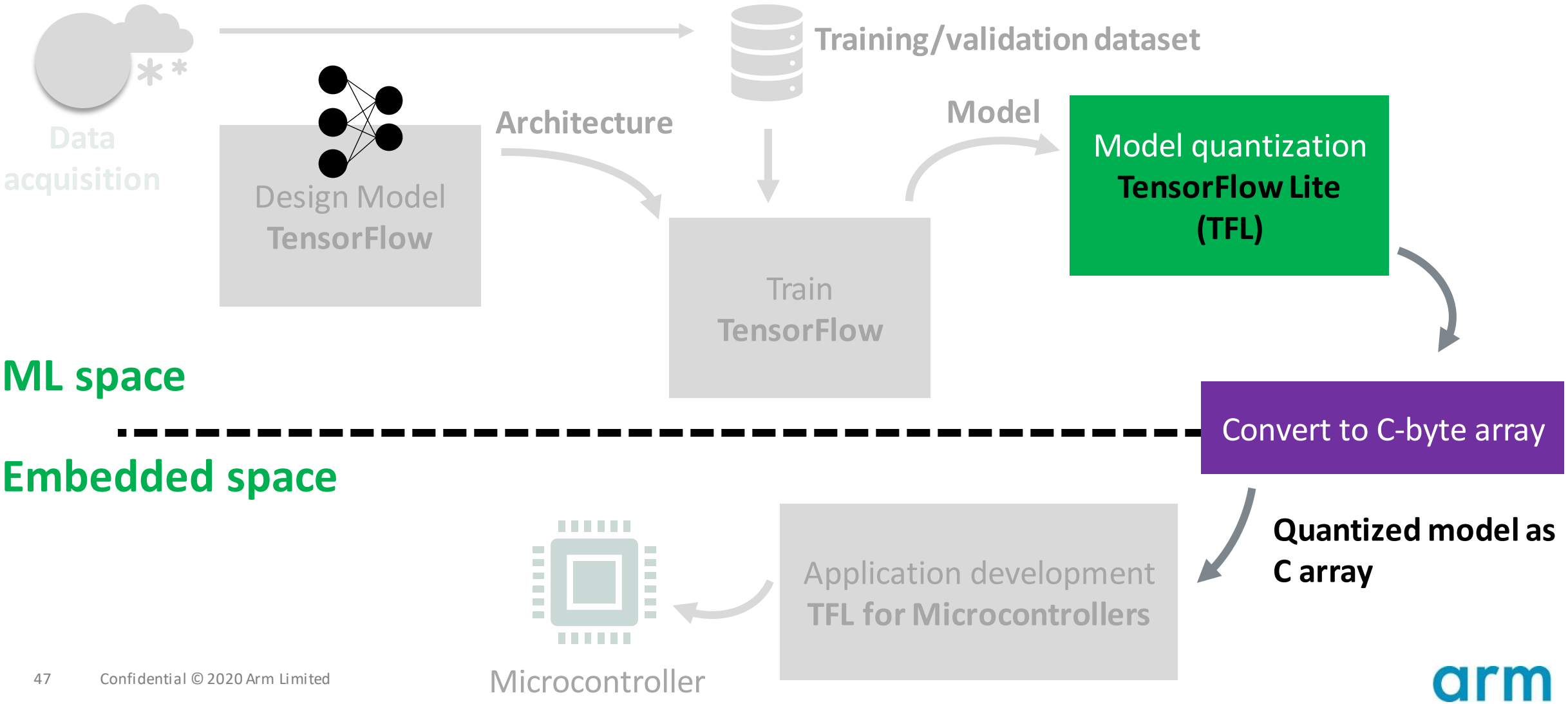
TensorFlow (https://www.tensorflow.org) is an end-to-end free and open source software platform developed by Google for ML.



Step 4: Evaluating the Model's effectiveness



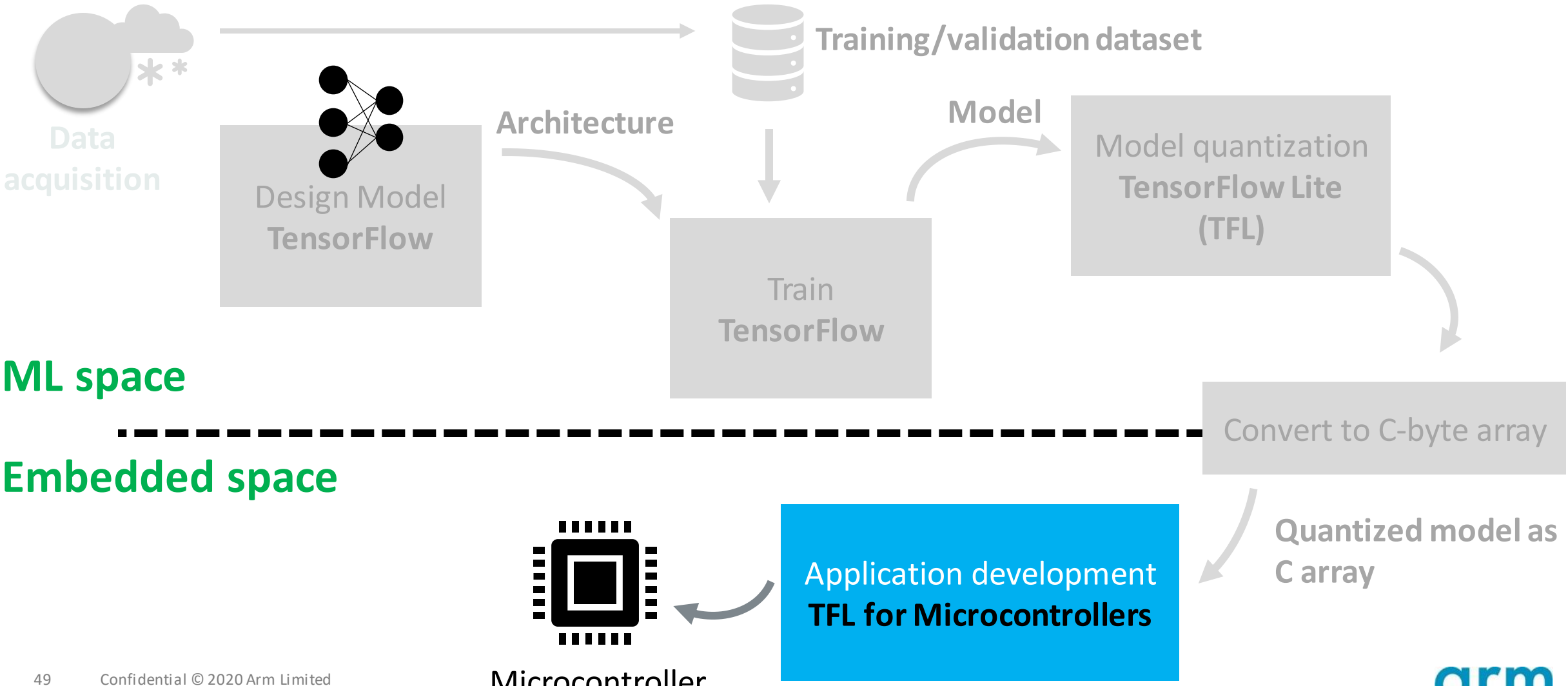
Step 5: Quantizing the Model



Step 5: What to Do

- Quantize to 8-bit the model with TensorFlow Lite Converter
- Convert the model to C-byte array

Step 6: Deploying the Model



Resources

- https://tinyml.seas.harvard.edu/teach/
- [Book] TinyML, Pete Warden & Daniel Situnayake
- [Book] TinyML Cookbook, Gian Marco Iodice

Thank You
Danke
Merci
谢谢
ありがとう
Gracias
Kiitos
감사합니다
धन्यवाद
شكرًا
ধন্যবাদ
תודה