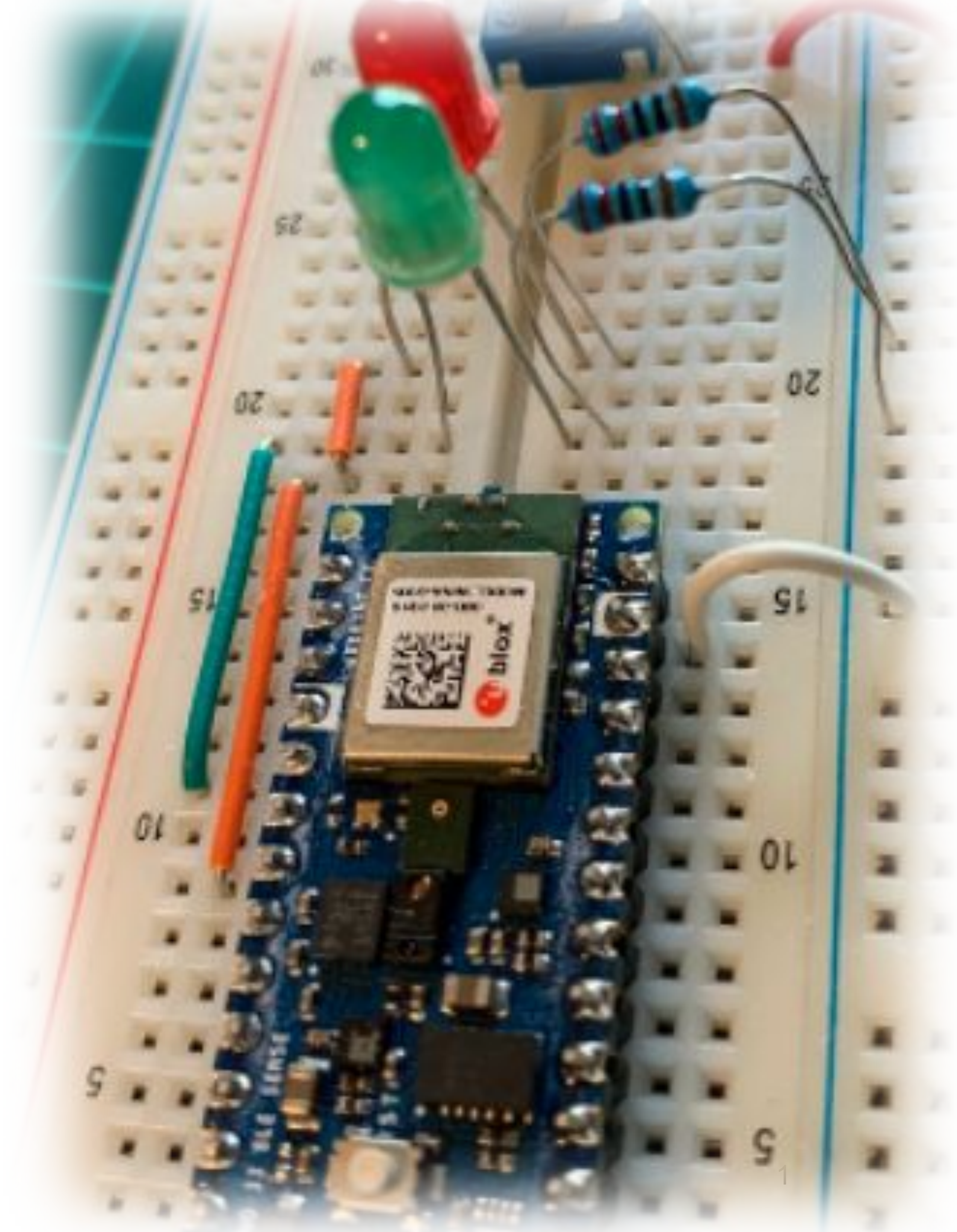
IESTI01 – TinyML

Embedded Machine Learning

14. Fundamentals wrap-up and Application's preview

Prof. Marcelo Rovai

UNIFEI



Tiny Machine Learning (TinyML)

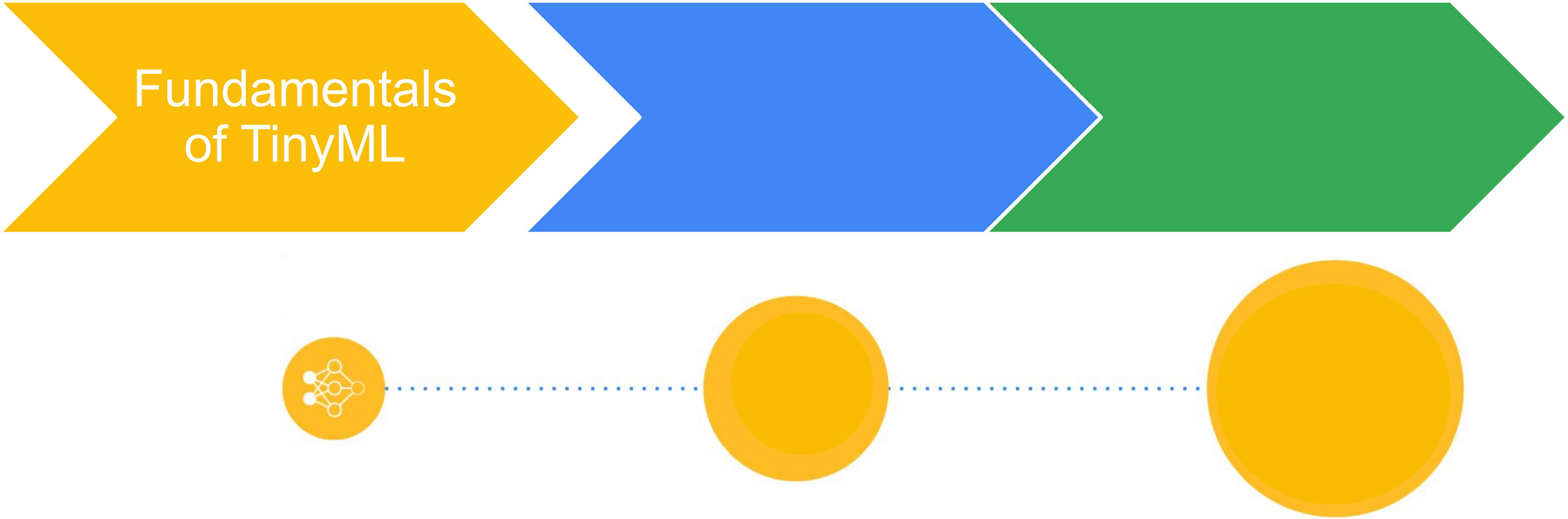
What we learned so far

What is Tiny Machine Learning (**TinyML**)?

- Fast-growing field of **machine learning**
- Algorithms, **hardware, and software**
- **On-device** sensor data analytics
- Extreme **low power** consumption
- **Always-on ML** use-cases
- **Battery**-operated devices

What we already learned?

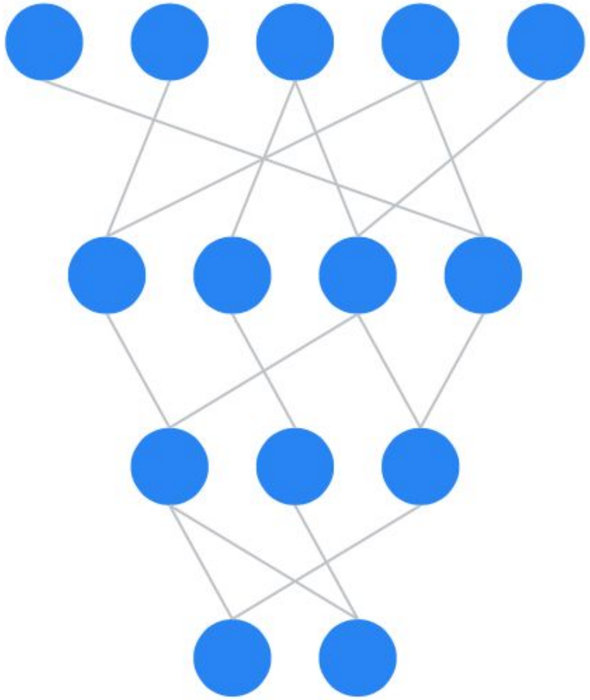
Part 1



Fundamentals of TinyML

So far in the Part 1, we introduced ML with TensorFlow. Was all about talking about what is the language of machine learning.

Total Recall from **Part 1**



“Language” for **Part 1**

Training Data

Neural Network

Training

Validation Data

Inference

Gradient Descent

Test Data

Features

Classification

Loss Function

Filters

Overfitting

Kernels

Regression

Data augmentation

CNNs

Responsible AI

DNNs

Preprocessing

“Language” for **Part 1**

Training Data

Neural Network

Training

Validation Data

Inference

Gradient Descent

Test Data

Features

Classification

Loss Function

Filters

Overfitting

Kernels

Regression

Data augmentation

CNNs

DNNs

Responsible AI

Preprocessing

“Language” for **Part 1**

Training Data

Neural Network

Training

Validation Data

Inference

Gradient Descent

Test Data

Features

Classification

Loss Function

Filters

Overfitting

Kernels

Regression

Data augmentation

CNNs

DNNs

Responsible AI

Preprocessing

“Language” for **Part 1**

Training Data

Neural Network

Training

Validation Data

Inference

Gradient Descent

Test Data

Features

Classification

Loss Function

Filters

Overfitting

Kernels

Regression

Data augmentation

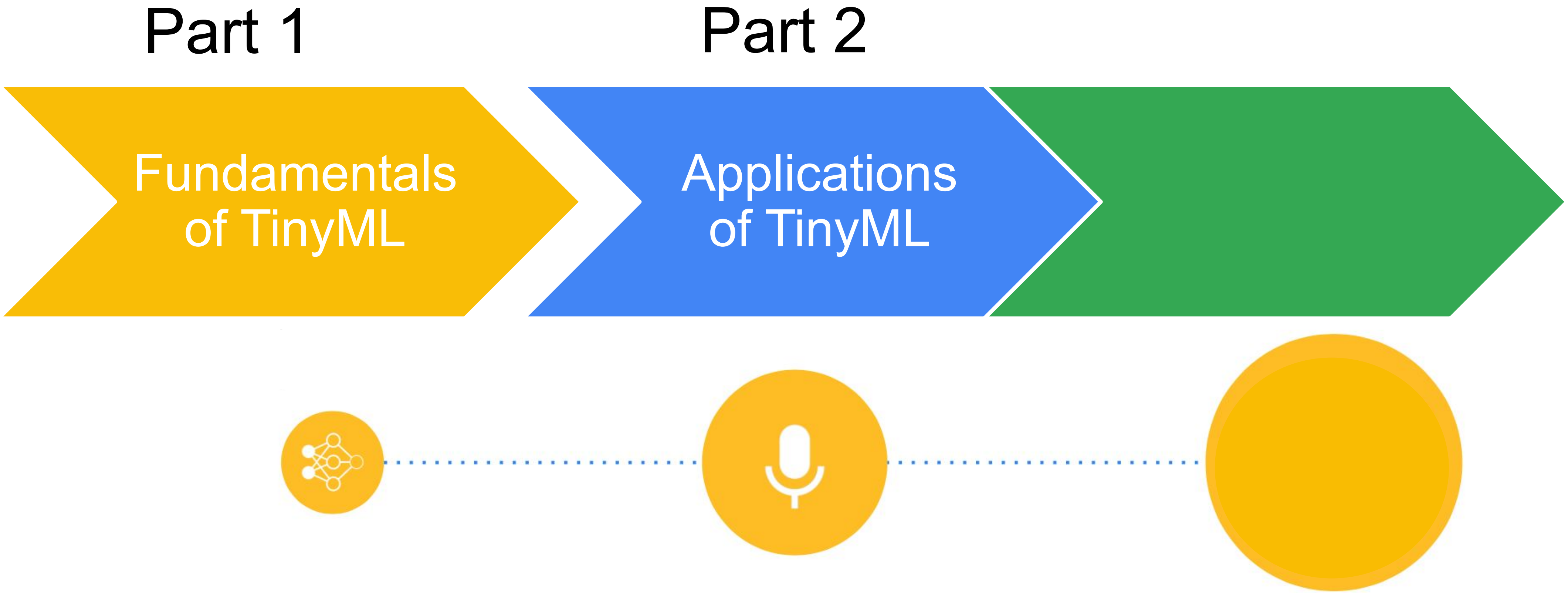
CNNs

DNNs

Responsible AI

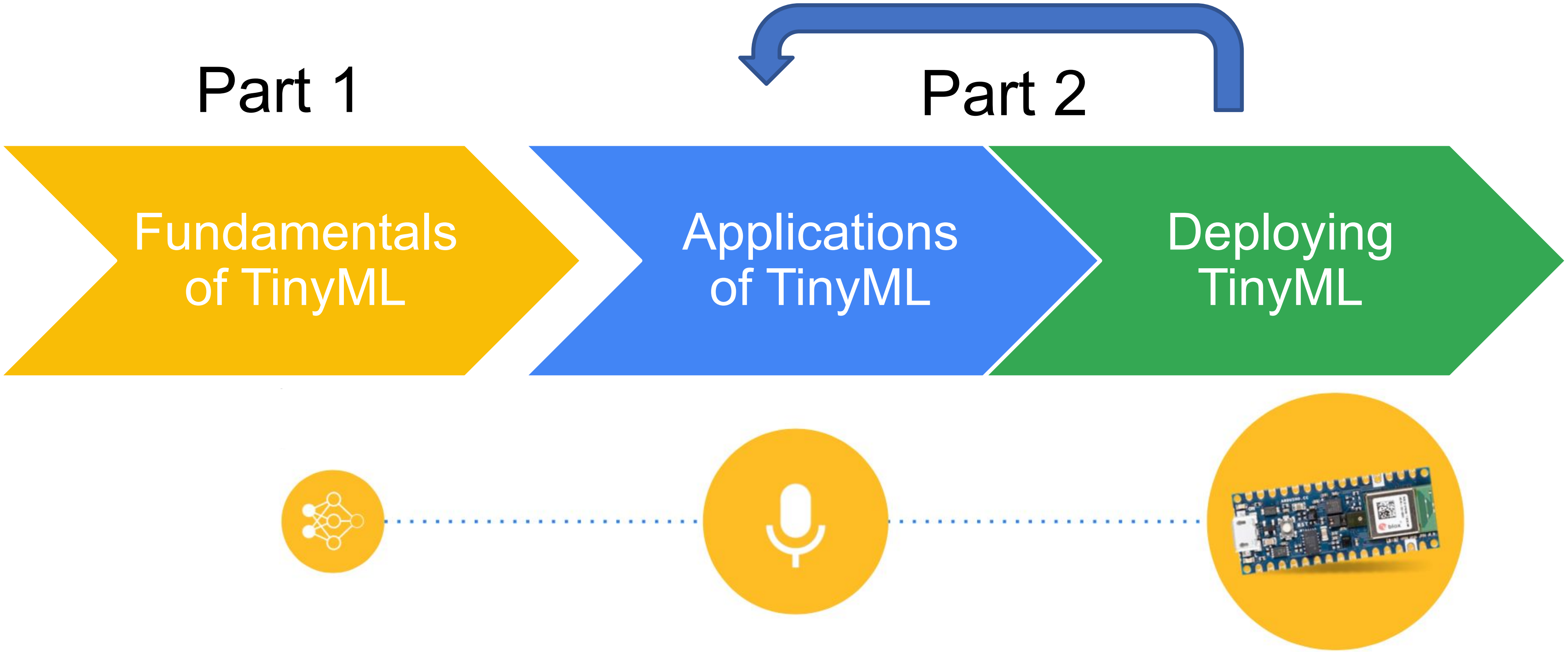
Preprocessing

What we will learn?



In Part 2, we will get a sneak peek into the variety of different TinyML applications, as keyword spotting (“Alexa”), gesture recognition, understand how to leverage the sensors, and so forth.

What we will learn?



Part 1

Part 2

Fundamentals of TinyML

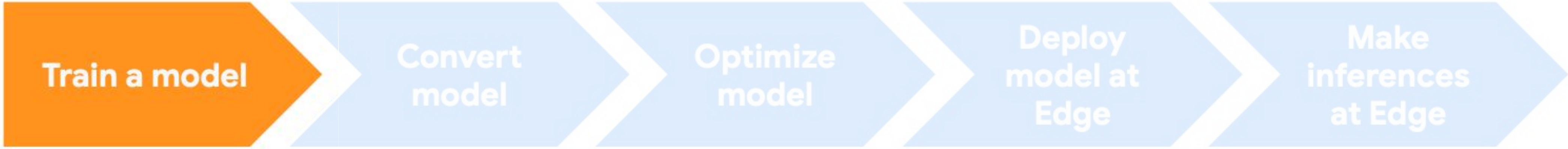
Applications of TinyML

Deploying TinyML

In Part 2, we will **also** learn how to deploy models on a real microcontroller. Along the way we will explore the challenges unique to and amplified by TinyML (e.g., preprocessing, post-processing, dealing with resource constraints).

TensorFlow

TensorFlow Lite


Train a model

Convert model

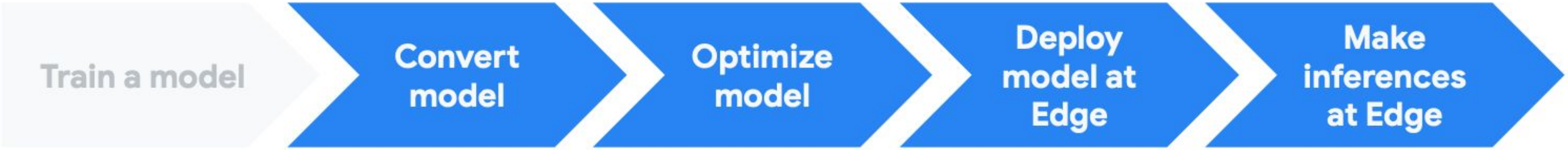
Optimize model

Deploy model at Edge

Make inferences at Edge

TensorFlow

TensorFlow Lite



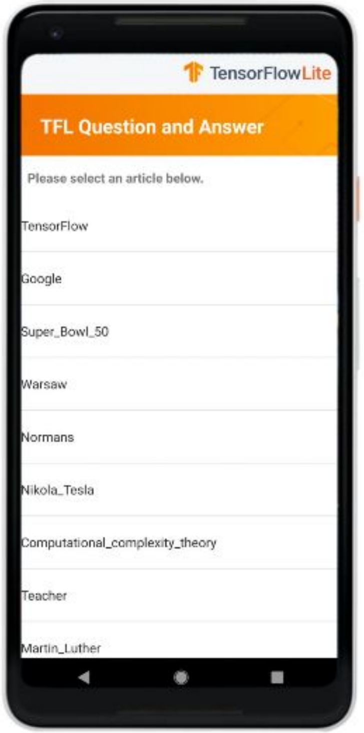
Train a model

Convert model

Optimize model

Deploy model at Edge

Make inferences at Edge



TensorFlowLite

TFL Question and Answer

Please select an article below.

TensorFlow

Google

Super_Bowl_50

Warsaw

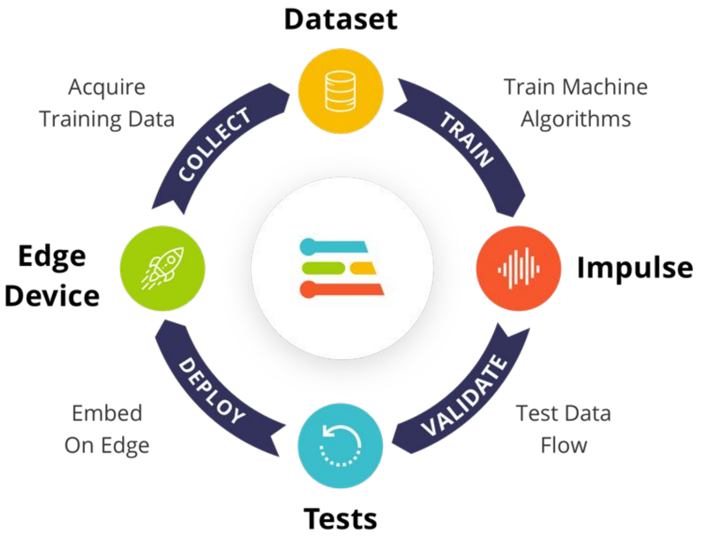
Normans

Nikola_Tesla

Computational_complexity_theory

Teacher

Martin_Luther



Tiny Machine Learning (TinyML)

Applications

TinyML Application Areas

Home

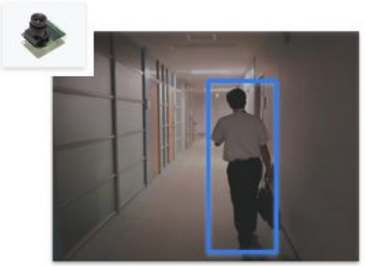
Office

Industry

TinyML Application Areas

Home

Office



Industry

Questions

- How do we **capture** the data to feed into the neural network?
- How do you **design** the neural network to take in the speech signal?
- What **dataset** does the neural network need to be trained?
- How do we **pre-process** the data for neural network inference?
- How do you **post-process** the neural network output?
- How do you make sure there is no **bias** in the dataset?
- How do you **deploy** this on the microcontroller?

Endpoints Have **Sensors**, Tons of Sensors

Motion Sensors
Gyroscope, Radar, Accelerometer

Acoustic Sensors
Ultrasonic, Microphones, Geophones, Vibrometers

Environmental Sensors
Temperature, Humidity, Pressure, IR, etc.

Touchscreen Sensors
Capacitive, IR

Image Sensors
Thermal, Image

Biometric Sensors
Fingerprint, Heart rate, etc.

Force Sensors
Pressure, Strain

Rotation Sensors
Encoders

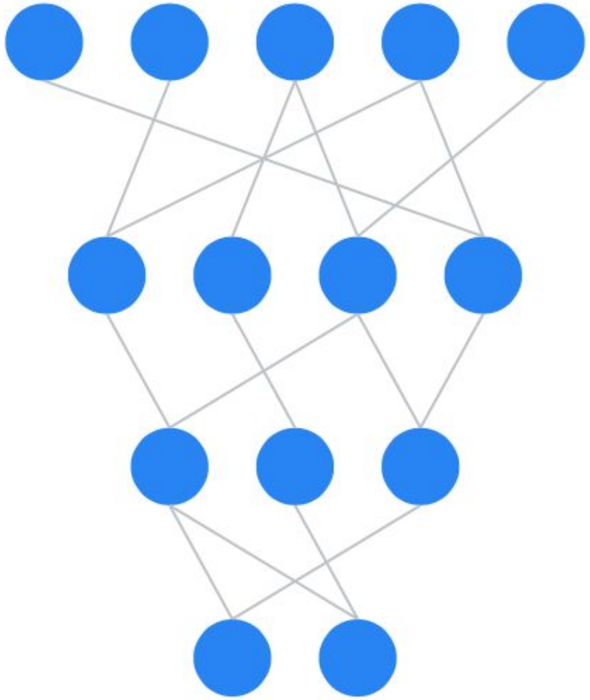
Sensors
Metrics

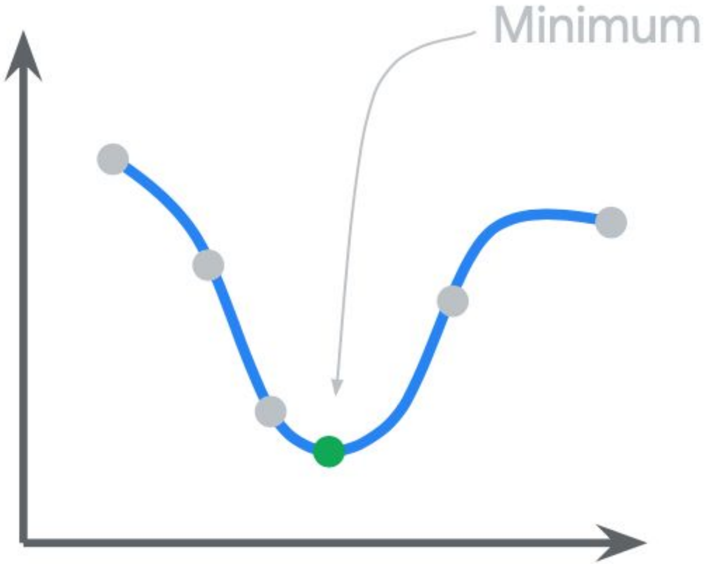
Models

Acoustic Sensors
Ultrasonic, **Microphones**,
Geophones, Vibrometers

Image Sensors
Thermal, **Image**

Motion Sensors
Gyroscope, Radar,
Accelerometer





End-to-end **TinyML** application design

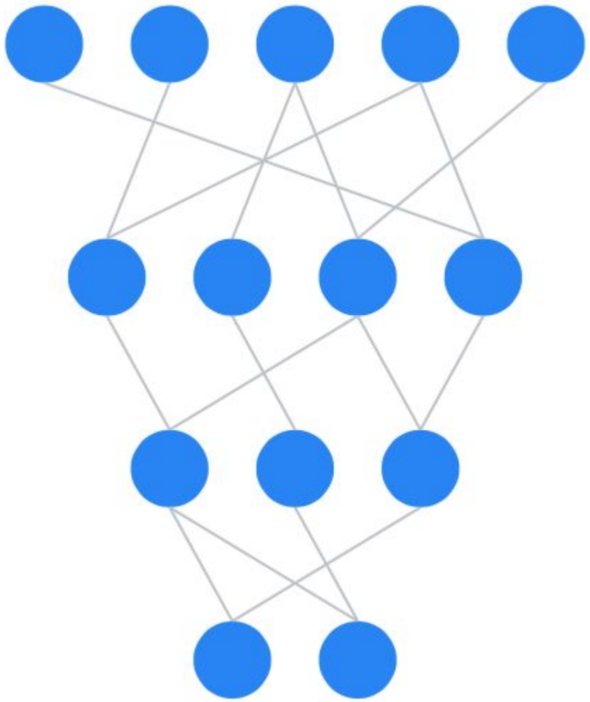
Datasets
Preprocessing

Sound

Vision

Vibration

Quantization
Pruning



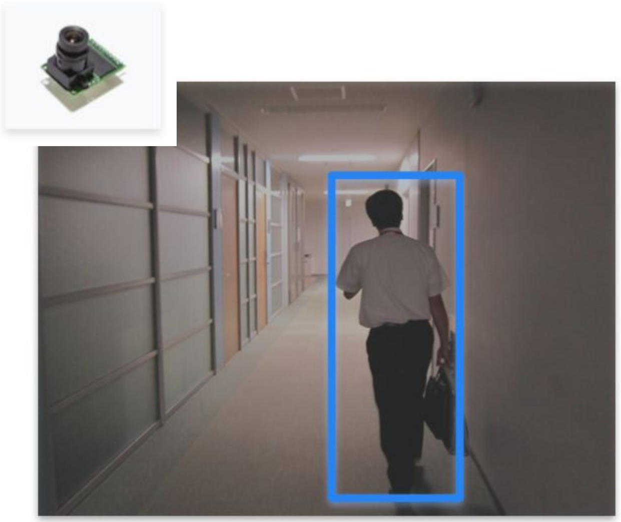
Resource
constraints

End-to-end **TinyML** application design

Sound

Vibration

Vision



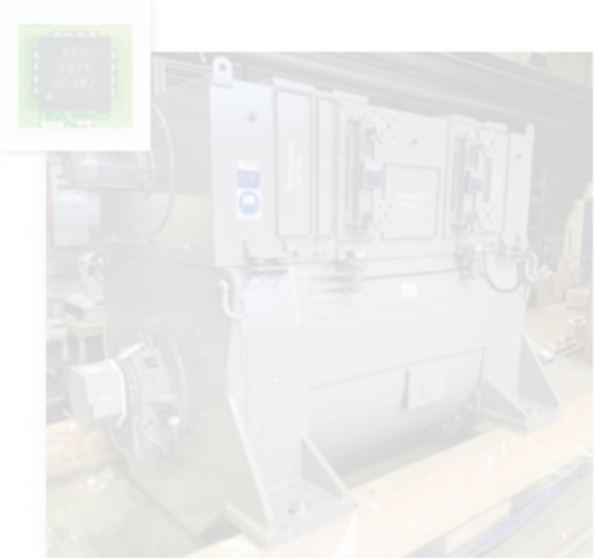
TinyML Application

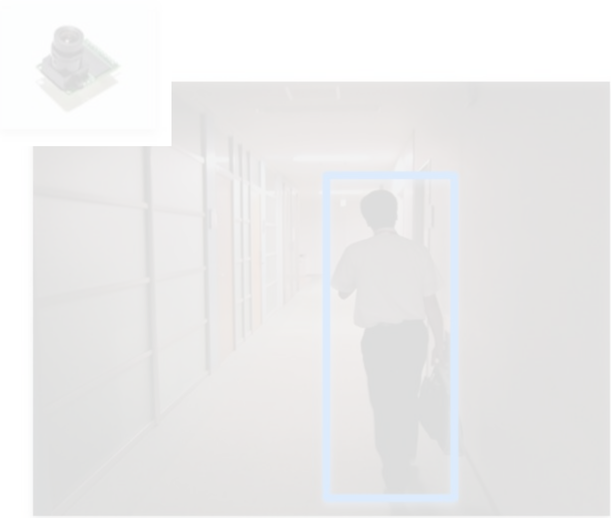
Example

Sound

Vibration

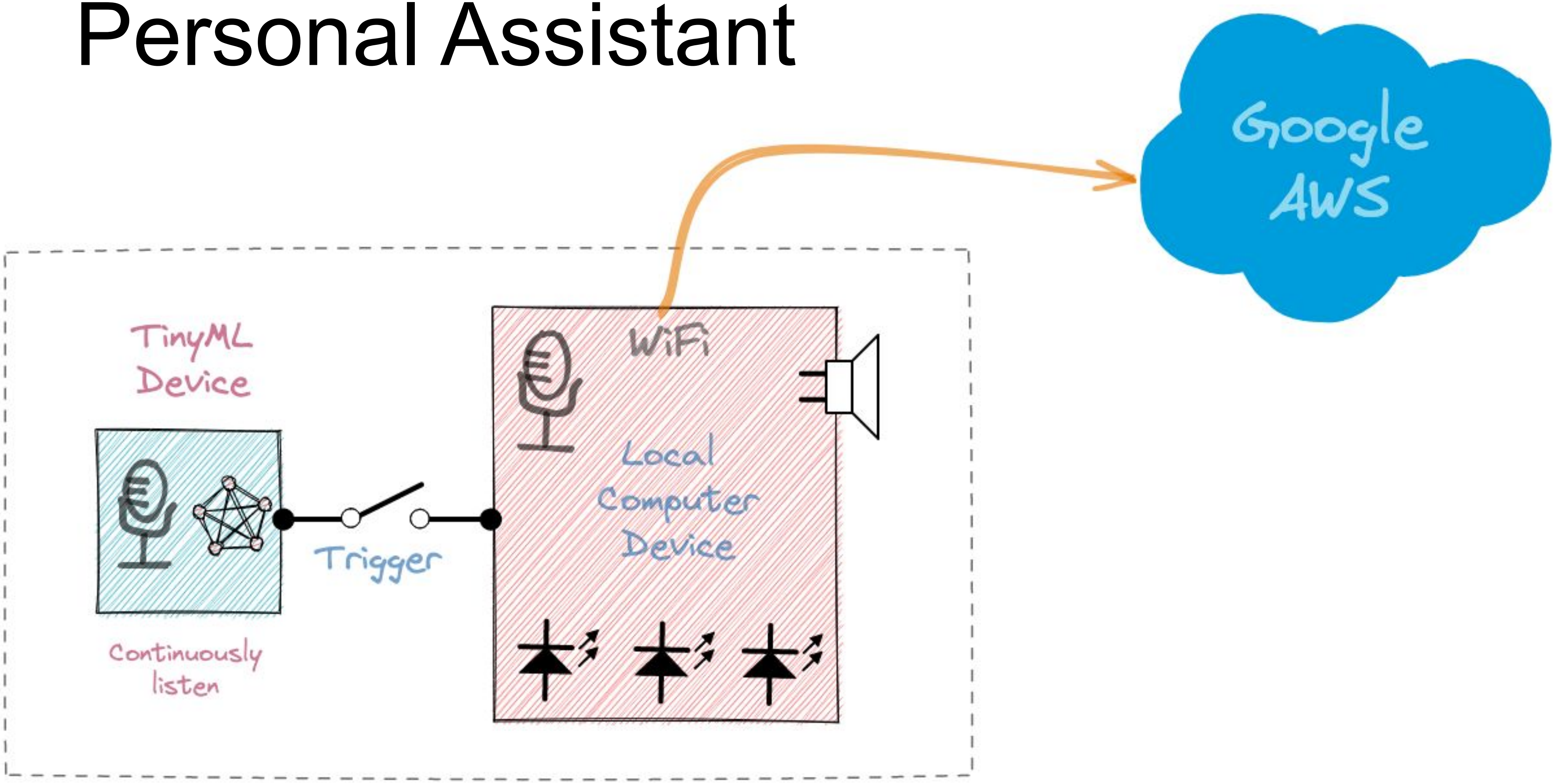
Vision



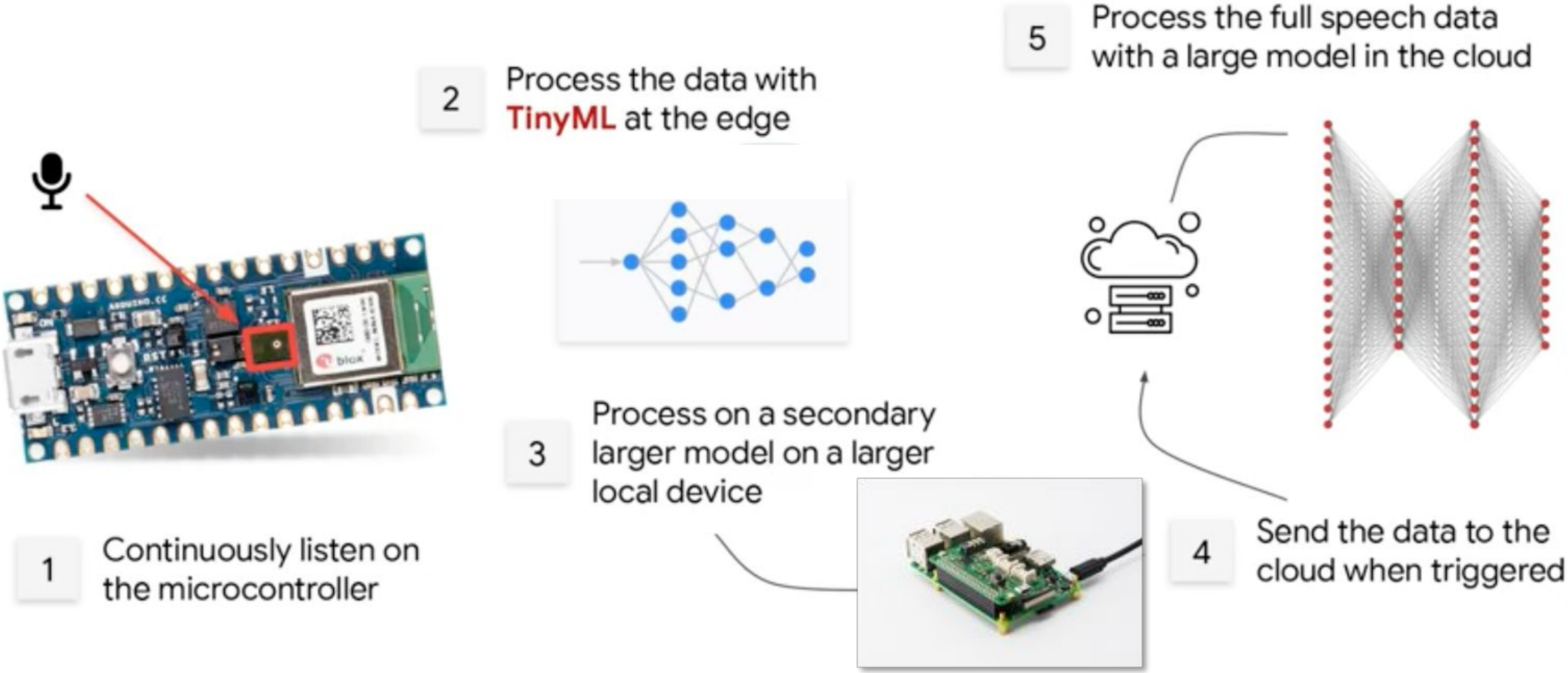


Personal Assistant

Personal Assistant



“Cascade” Detection: multi-stage model



1 Continuously listen on the microcontroller

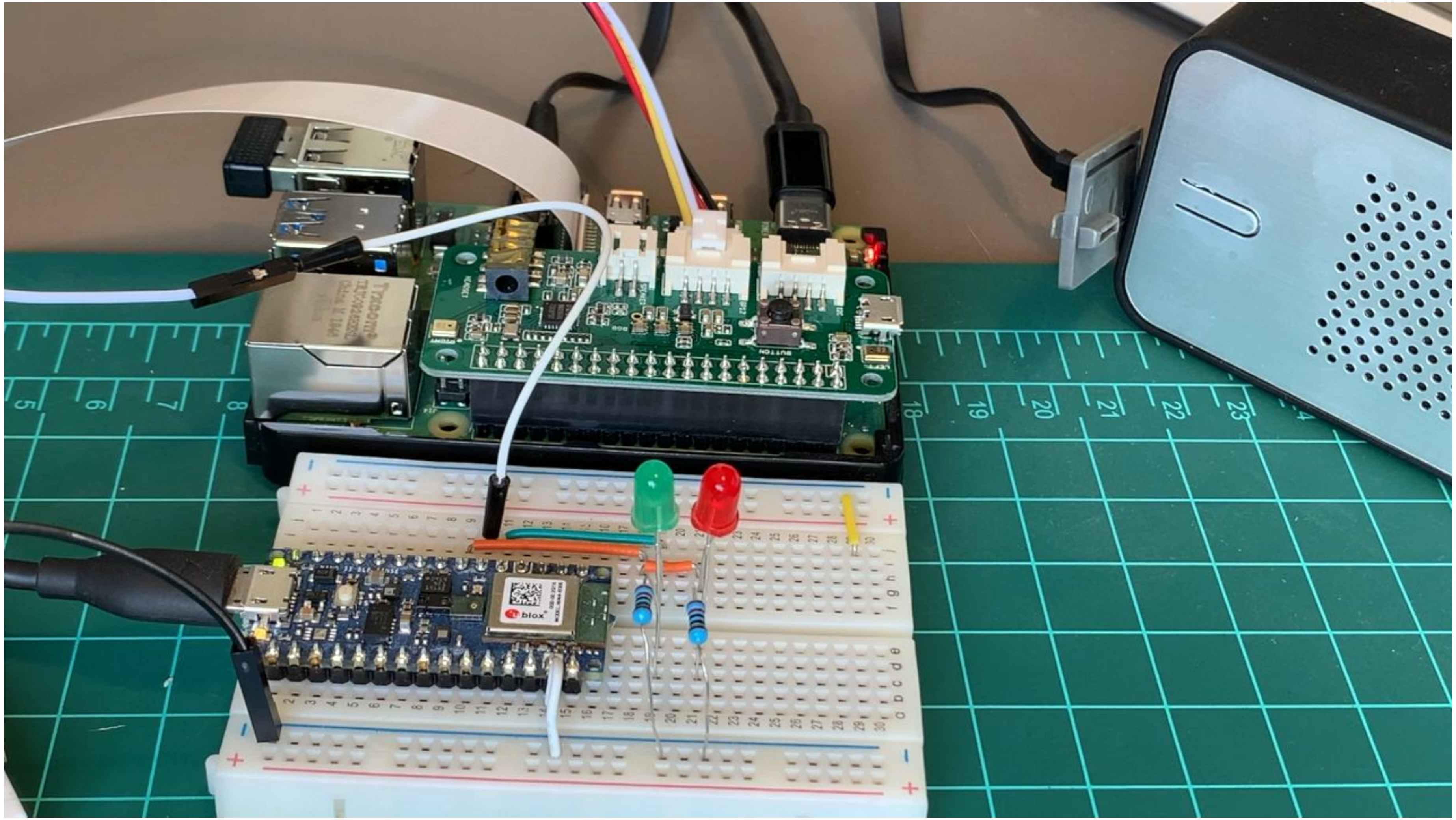
2 Process the data with **TinyML** at the edge

3 Process on a secondary larger model on a larger local device

4 Send the data to the cloud when triggered

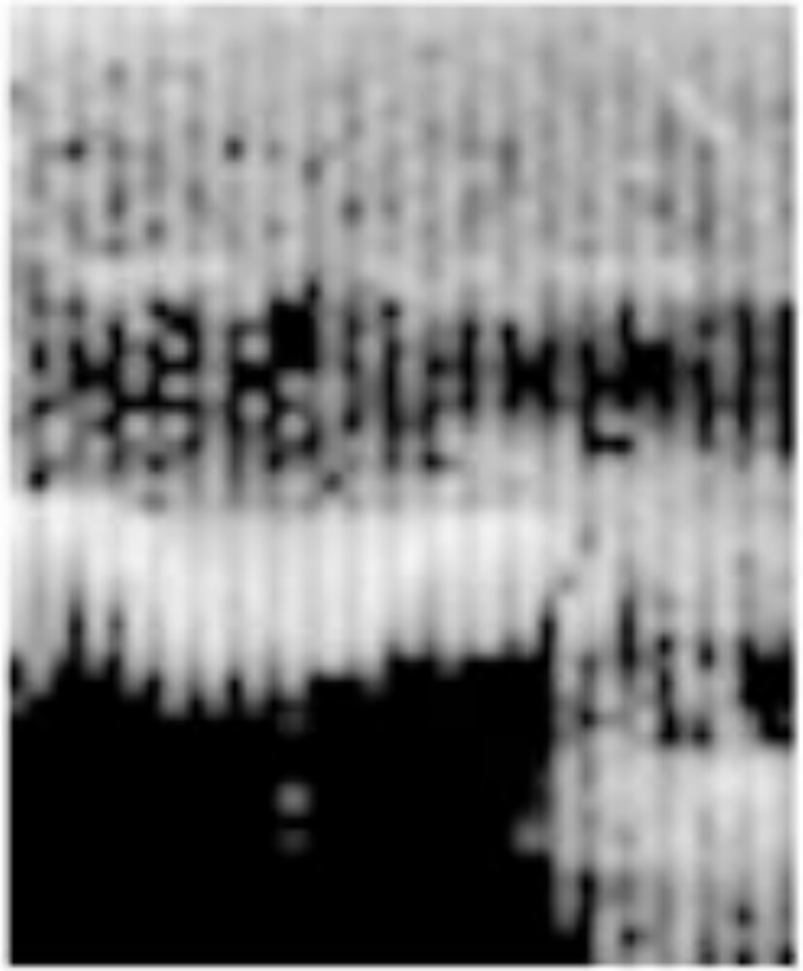
5 Process the full speech data with a large model in the cloud

KeyWord Spotting (KWS)



https://mjrobot.org/2021/01/27/building-an-intelligent-voice-assistant-from-scratch/

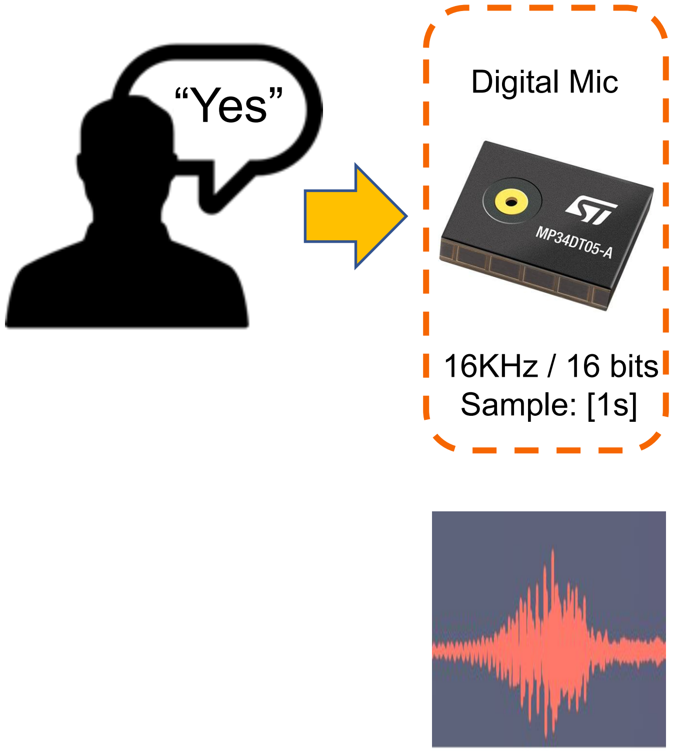
Sound



Image

KeyWord Spotting (KWS) - **Inference**

Obtains an input

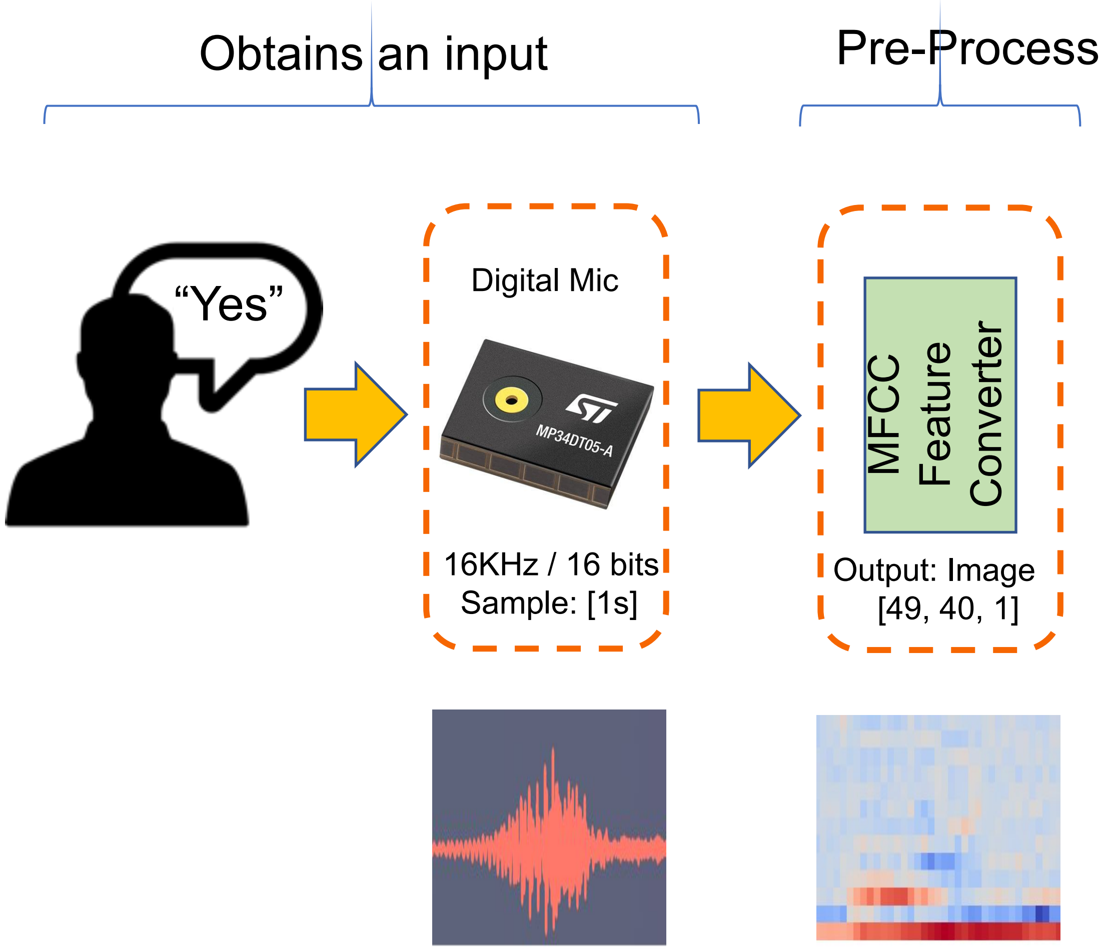


“Yes”

Digital Mic

16KHz / 16 bits
Sample: [1s]

KeyWord Spotting (KWS) - **Inference**



Obtains an input

Pre-Process

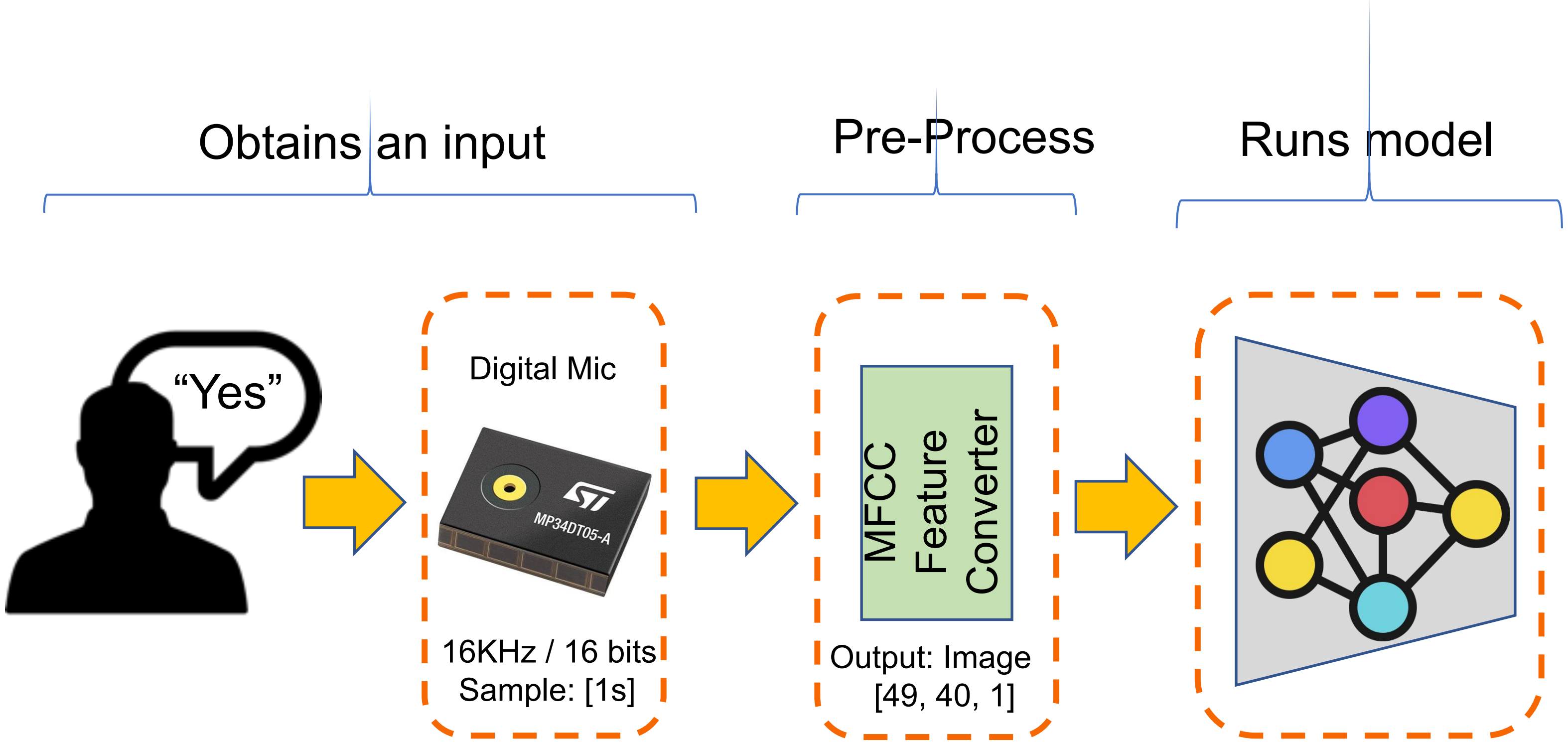
Digital Mic

16KHz / 16 bits
Sample: [1s]

MFCC
Feature
Converter

Output: Image
[49, 40, 1]

KeyWord Spotting (KWS) - **Inference**



Obtains an input

Pre-Process

Runs model

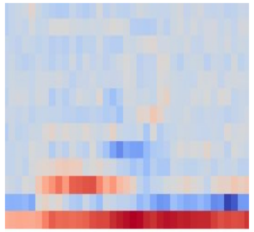
“Yes”

Digital Mic

16KHz / 16 bits
Sample: [1s]

MFCC Feature Converter

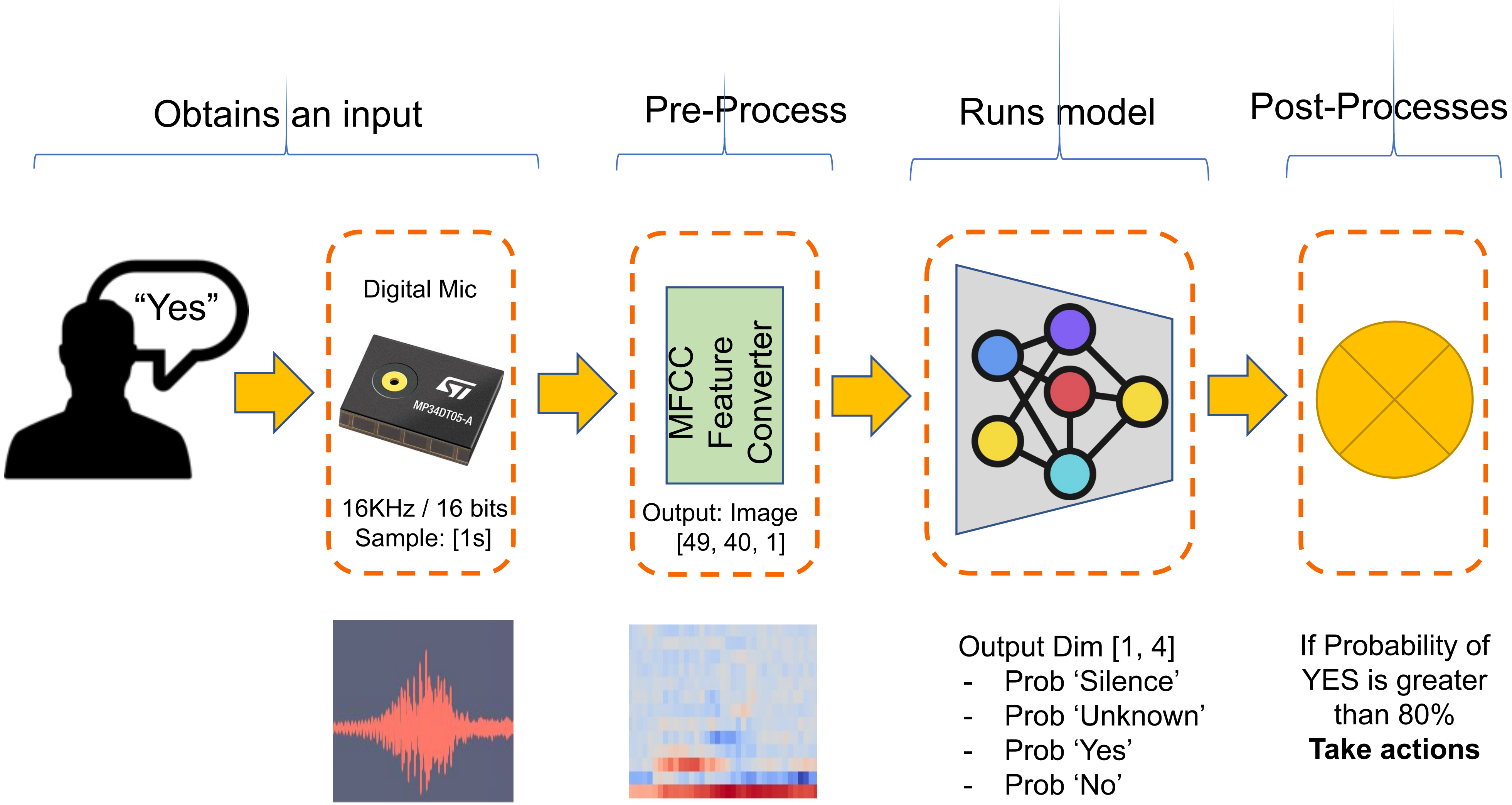
Output: Image
[49, 40, 1]



Output Dim [1, 4]

- Prob ‘Silence’
- Prob ‘Unknown’
- Prob ‘Yes’
- Prob ‘No’

KeyWord Spotting (KWS) - **Inference**



Obtains an input

Pre-Process

Runs model

Post-Processes

16KHz / 16 bits
Sample: [1s]

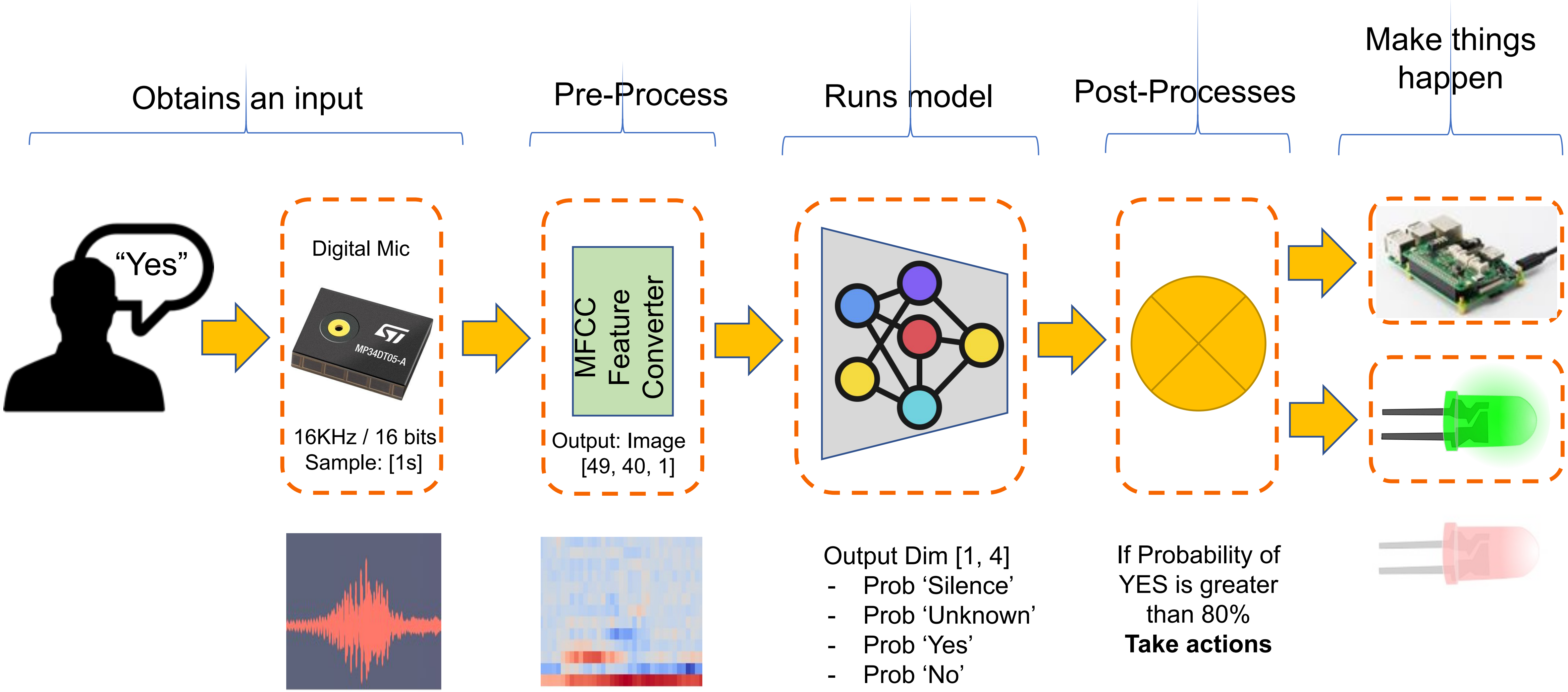
Output: Image
[49, 40, 1]

Output Dim [1, 4]

- Prob ‘Silence’
- Prob ‘Unknown’
- Prob ‘Yes’
- Prob ‘No’

If Probability of
YES is greater
than 80%
Take actions

KeyWord Spotting (KWS) - **Inference**



Obtains an input

Pre-Process

Runs model

Post-Processes

Make things happen

Digital Mic

16KHz / 16 bits
Sample: [1s]

MFCC Feature Converter

Output: Image
[49, 40, 1]

Output Dim [1, 4]

- Prob ‘Silence’
- Prob ‘Unknown’
- Prob ‘Yes’
- Prob ‘No’

If Probability of YES is greater than 80%
Take actions

KeyWord Spotting (KWS) - **Model**

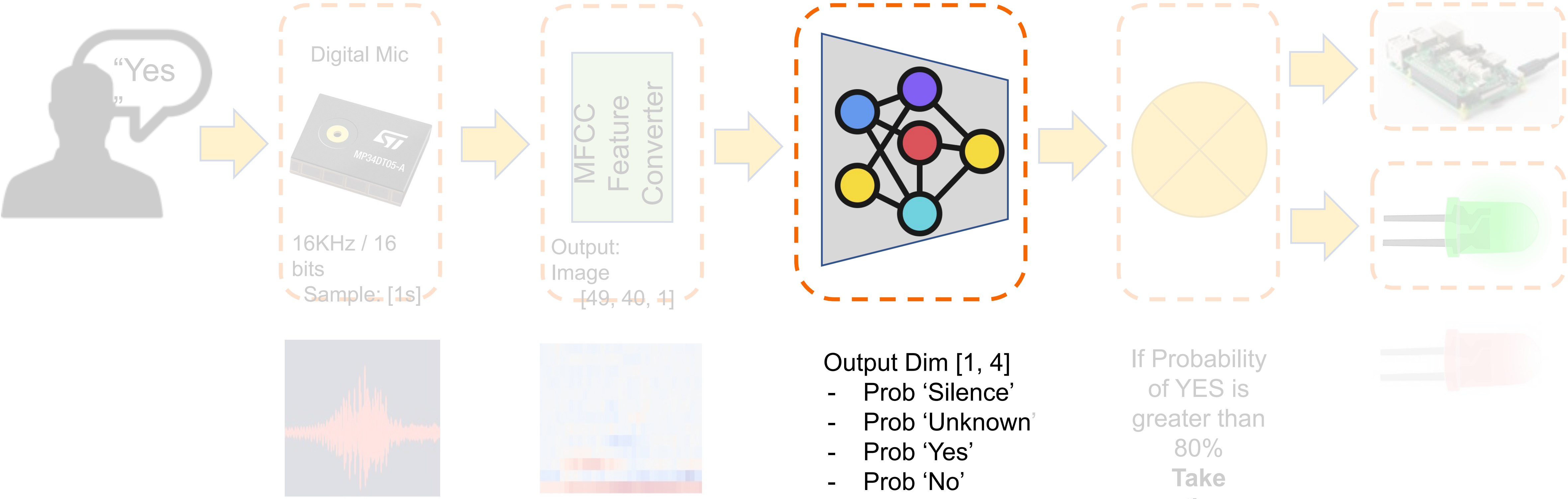
Obtains an input

Pre-Process

Runs model

postprocesses

Make things happen



16KHz / 16 bits Sample: [1s]

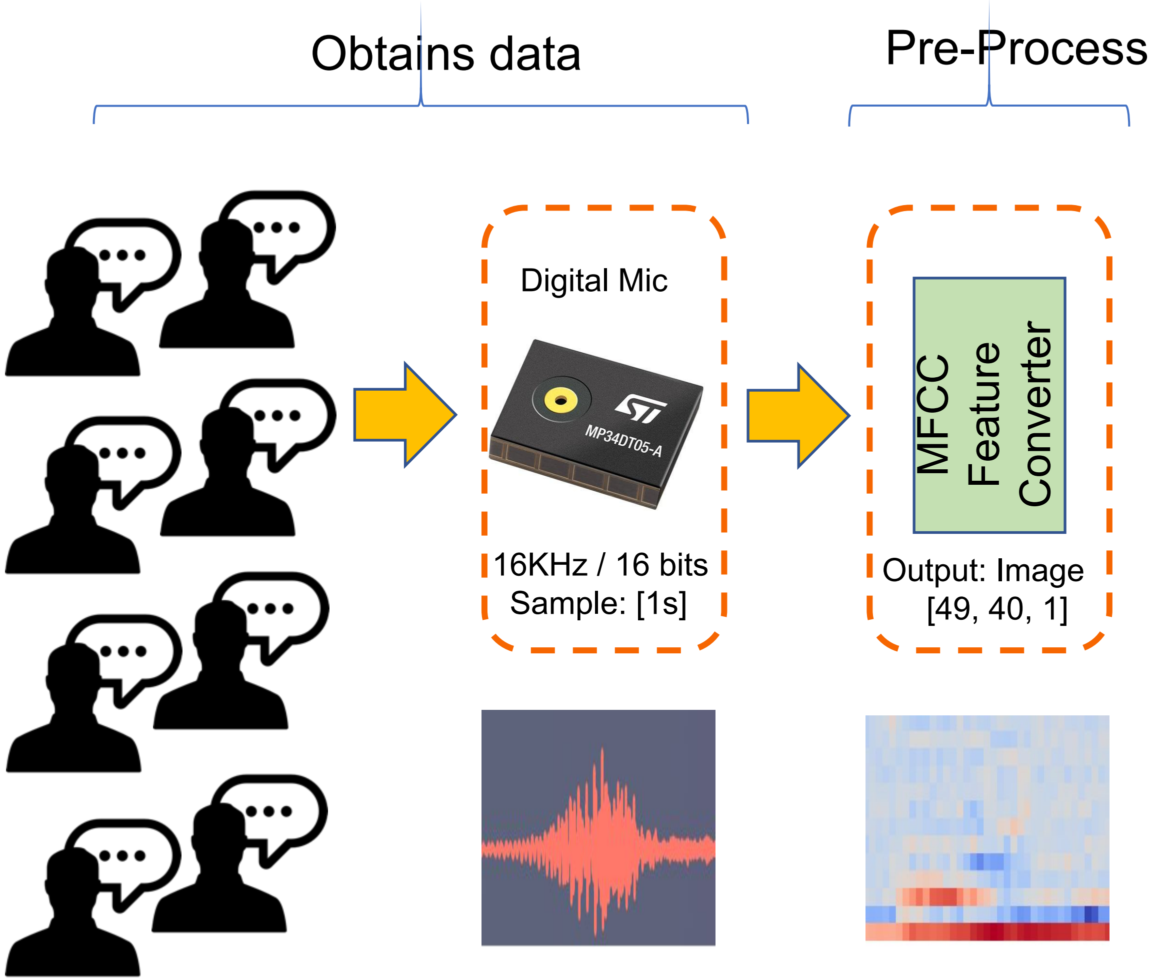
Output: Image [49, 40, 1]

Output Dim [1, 4]

- Prob ‘Silence’
- Prob ‘Unknown’
- Prob ‘Yes’
- Prob ‘No’

If Probability of YES is greater than 80% **Take actions**

KeyWord Spotting (KWS) – Create Model (Training)



Obtains data

Pre-Process

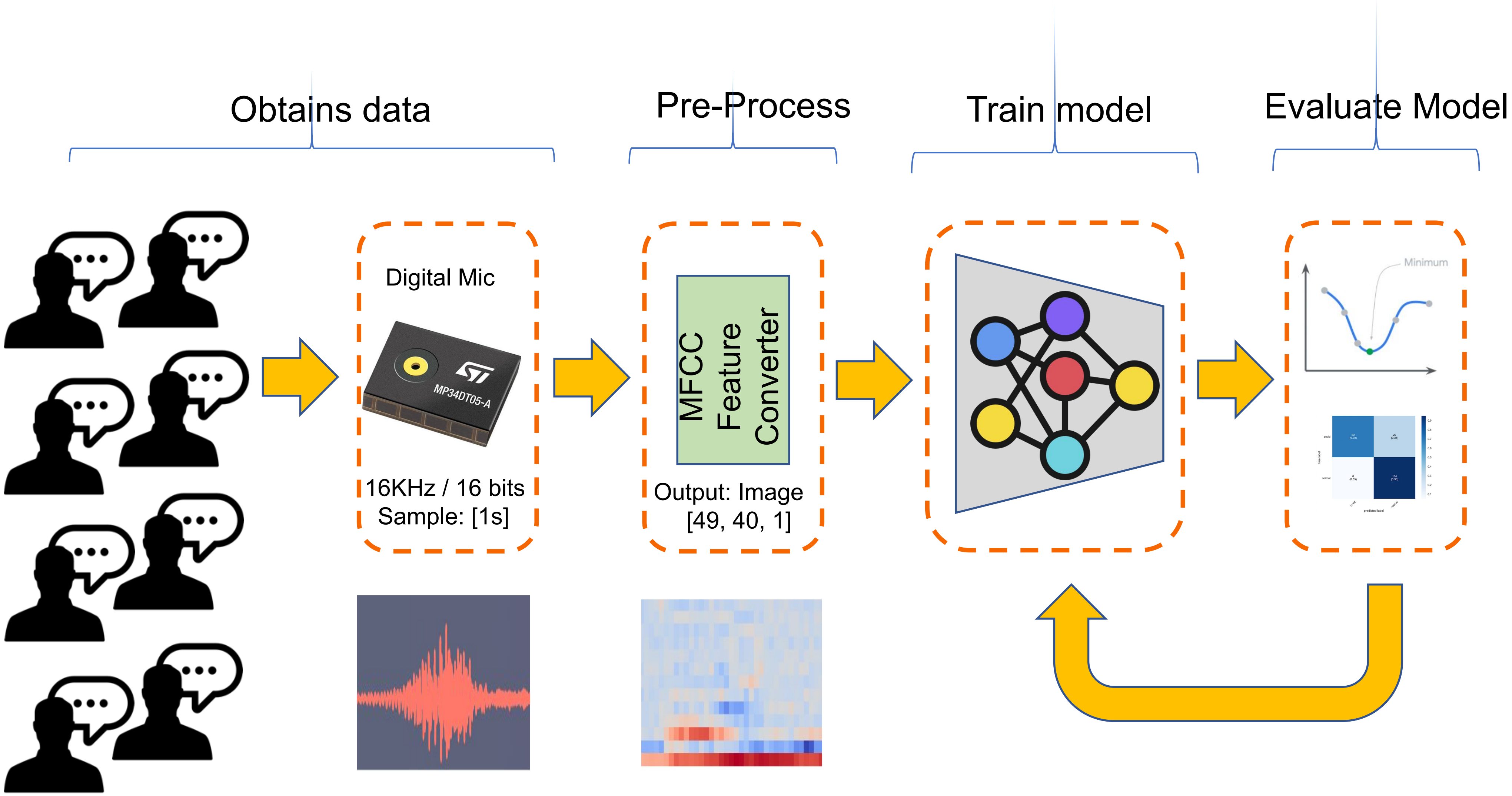
Digital Mic

16KHz / 16 bits
Sample: [1s]

MFCC Feature Converter

Output: Image
[49, 40, 1]

KeyWord Spotting (KWS) – Create Model (Training)



Obtains data

Pre-Process

Train model

Evaluate Model

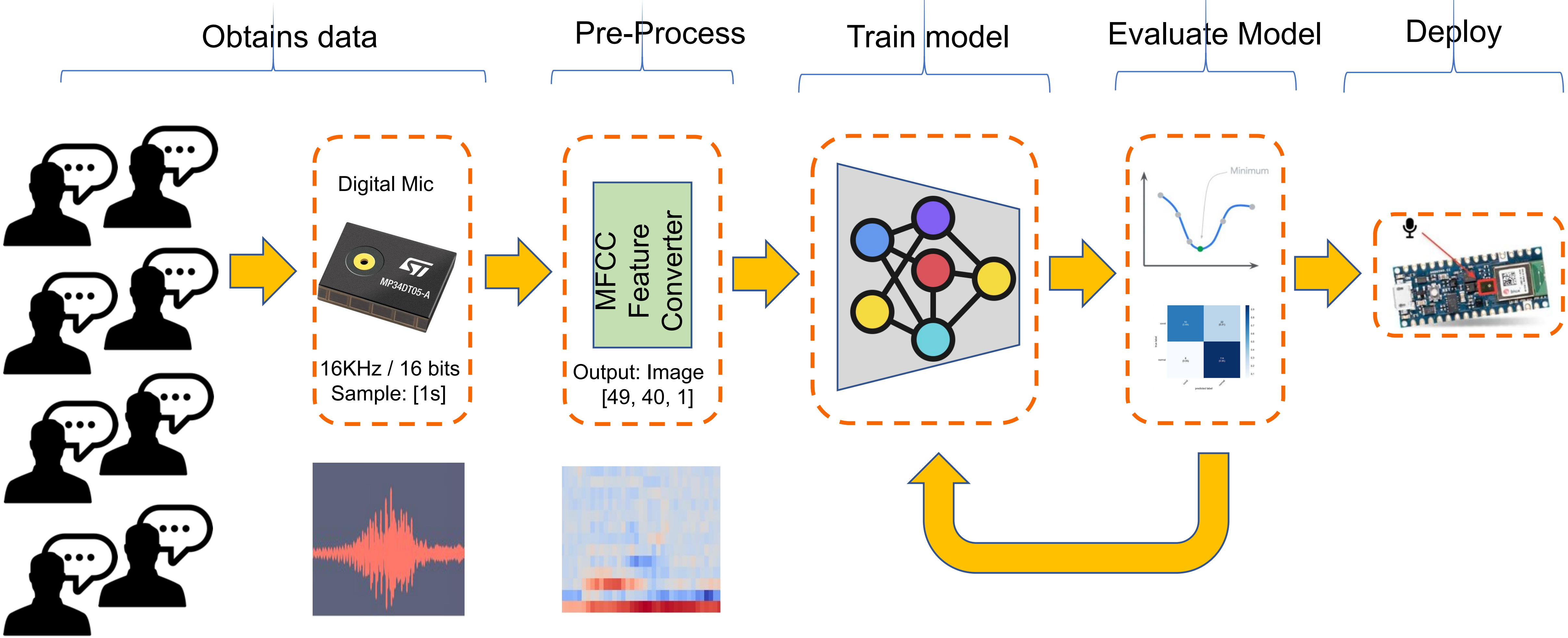
Digital Mic

16KHz / 16 bits
Sample: [1s]

MFCC Feature Converter

Output: Image
[49, 40, 1]

KeyWord Spotting (KWS) – Create Model (Training)



Obtains data

Pre-Process

Train model

Evaluate Model

Deploy

Digital Mic

16KHz / 16 bits
Sample: [1s]

MFCC Feature Converter

Output: Image
[49, 40, 1]

Reading Material

Main references

- Harvard School of Engineering and Applied Sciences - CS249r: Tiny Machine Learning
- Professional Certificate in Tiny Machine Learning (TinyML) – edX/Harvard
- Introduction to Embedded Machine Learning (Coursera)
- Text Book: "TinyML" by Pete Warden, Daniel Situnayake

I want to thank Shawn Hymel and Edge Impulse, Pete Warden and Laurence Moroney from Google, and especially Harvard professor Vijay Janapa Reddi, Ph.D. student Brian Plancher and their staff for preparing the excellent material on TinyML that is the basis of this course at UNIFEI.

The IESTI01 course is part of the TinyML4D, an initiative to make TinyML education available to everyone globally.

Thanks

And stay safe!

UNIFEI