CEC Continuing Education Center

DesignNews

Getting Started in TinyML with Arduino

DAY 5: Micro-Speech-Key Word Recognition Project

Sponsored by

DigiKey

informa markets

Webinar Logistics

- Turn on your system sound to hear the streaming presentation.
- If you have technical problems, click “Help” or submit a question asking for assistance.
- Participate in ‘Attendee Chat’ by maximizing the chat widget in your dock.

Dr. Don Wilcher

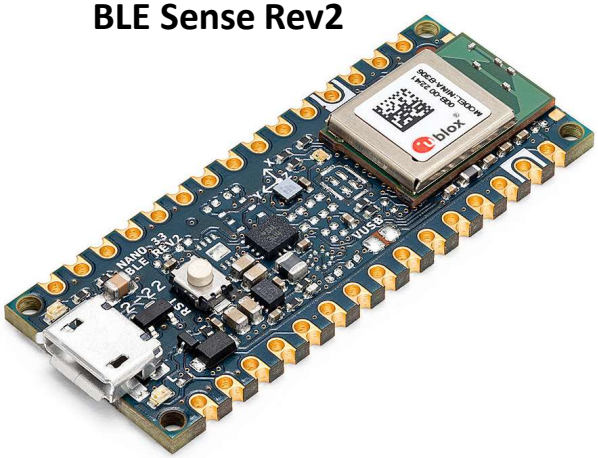
Visit ‘Lecturer Profile’ in your console for more details.

LinkedIn Page:
https://www.linkedin.com/in/dr-don-wilcher-ed-d-mseit-ee-ceta-2735151/

Patreon Page:
https://www.patreon.com/c/DrDon683

Course Kit and Materials

Arduino Nano 33 BLE Sense Rev2



Research Literature and Documentation

Tiny Machine Learning: Progress and Futures

Ji Lin Ligeng Zhu Wei-Ming Chen Wei-Chen Wang Song Han

Massachusetts Institute of Technology

https://tinyml.mit.edu

Abstract—Tiny Machine Learning (TinyML) is a new frontier of machine learning. By squeezing deep learning models into billions of IoT devices and microcontrollers (MCUs), we expand the scope of AI applications and enable ubiquitous intelligence. However, TinyML is challenging due to hardware constraints: the tiny memory resource makes it difficult to hold deep learning models designed for cloud and mobile platforms. There is also limited compiler and inference engine support for bare-metal devices. Therefore, we need to co-design the algorithm and system stack to enable TinyML. In this review, we will first discuss the definition, challenges, and applications of TinyML. We then survey the recent progress in TinyML and deep learning on MCUs. Next, we will introduce MCUNet, showing how we can achieve ImageNet-scale AI applications on IoT devices with *system-algorithm co-design*. We will further *extend* the solution from *inference to training* and introduce tiny on-device training techniques. Finally, we present future directions in this area. Today's "large" model might be tomorrow's "tiny" model. The scope of TinyML should evolve and adapt over time.

Index Terms—TinyML, Efficient Deep Learning, On-Device Training, Learning on the Edge

I. Overview of Tiny Machine Learning

Machine learning (ML) has made significant impacts on various fields, including vision, language, and audio. However, state-of-the-art models often come at the cost of high computation and memory, making them expensive to deploy. To address this, researchers have been working on efficient algorithms, systems, and hardware to reduce the cost of machine learning models in various deployment scenarios. There are two main subdomains of efficient ML: EdgeML and CloudML (Figure 1). While CloudML focuses on improving latency and throughput on cloud servers, EdgeML focuses on improving energy efficiency, latency, and privacy on edge devices. These two domains also intersect in areas such as hybrid inference [1, 2], over-the-air (OTA) updates, and federated learning between the edge and cloud [3]. In recent years, there has been significant progress in extending the scope of EdgeML to ultra-low-power devices such as IoT devices and microcontrollers, known as **TinyML**.

TinyML has several key advantages. It enables machine learning using only a few hundred kilobytes of memory which greatly reduces the cost. With billions of IoT devices producing more and more data in our daily lives, there is a growing need for low-power, always-on, on-device AI. By performing on-device inference near the sensor, TinyML enables better responsiveness and privacy while reducing the energy cost associated with wireless communication. On-device processing of data can be beneficial for applications where real-time decision-making is crucial, such as autonomous vehicles.

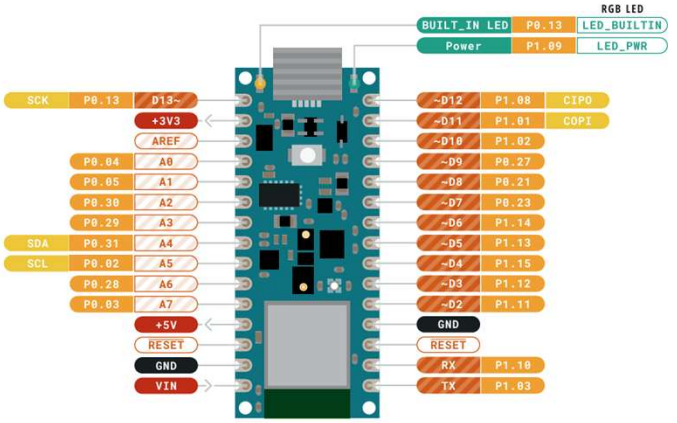
Fig. 1. Efficiency is critical for CloudML, EdgeML, and TinyML. CloudML targets high-throughput accelerators like GPUs, while EdgeML focuses on portable devices like mobile phones. TinyML further pushes the efficiency boundary, enabling powerful ML models to run on ultra-low-power devices such as microcontrollers.

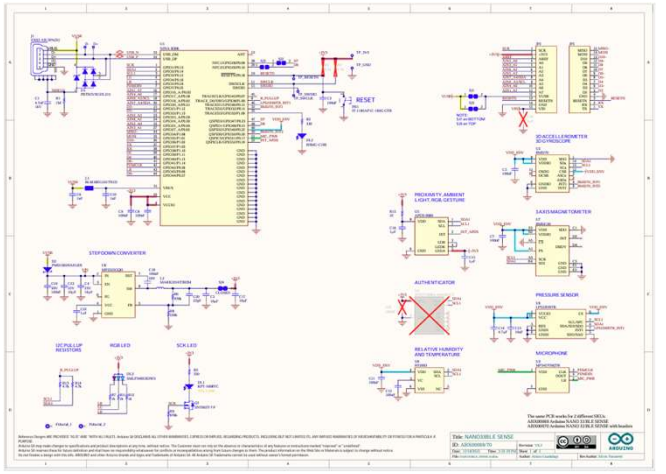
In addition to inference, we push the frontier of TinyML to enable on-device training on IoT devices. Irrevolutionizes EdgeAI through continuous and lifelong learning. Edge device can finetune the model on itself rather than transmitting data to cloud servers, which protects privacy. On-device learning has numerous benefits and a variety of applications. For example, home cameras can continuously recognize new faces, and email clients can gradually improve their prediction by updating customized language models. It also enables IoT applications that do not have a physical connection to the internet to adapt to the environment, such as precision agriculture and ocean sensing.

In this review, we will first discuss the definition and challenges of TinyML, analyzing why we can't directly scale mobile ML or cloud ML models for tinyML. Then we delve into the importance of system-algorithm co-design in TinyML. We will then survey recent literature and the progress of the field, presenting a holistic survey and comparison in Tables 1 and 10. Next, we will introduce our TinyML project, MCUNet, which combines efficient system and algorithm design to enable TinyML for both inference to training. Finally, we will discuss several emerging topics for future research directions in the field.

A. Challenges of TinyML

The success of deep learning models often comes at the cost of high computation, which is not feasible for use in TinyML applications due to the strict resource constraints of devices such as microcontrollers. Deploying and training AI models on MCU is extremely hard: No DRAM, no operating systems (OS), and strict memory constraints (SRAM is smaller than 256kB, and FLASH is read-only). The available resources on these devices are orders of magnitude smaller than those





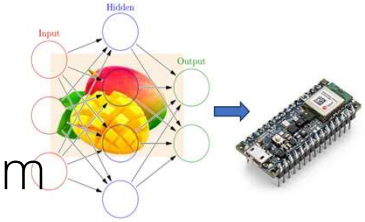
Research Perspective

"Today's large model might be tomorrow's *tiny model*."
[1] Lin et al., 2024

Agenda:

- What is The Micro-Speech Application?
- How Micro-Speech Works?
- Why Micro-Speech Is Important for TinyML?
- Lab: Hands-On With The Micro-Speech Application

What is The Micro-Speech Application?



Micro-Speech is a highly optimized keyword-spotting (KWS) example from TensorFlow Lite for Microcontrollers (TFLM).

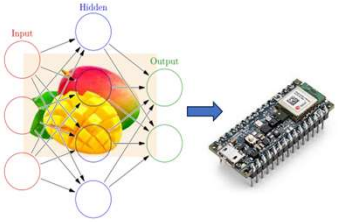
It recognizes simple speech commands, typically:

a) "Yes"
b) "No"
c) Unknown word (anything that is not yes/no)
d) Silence

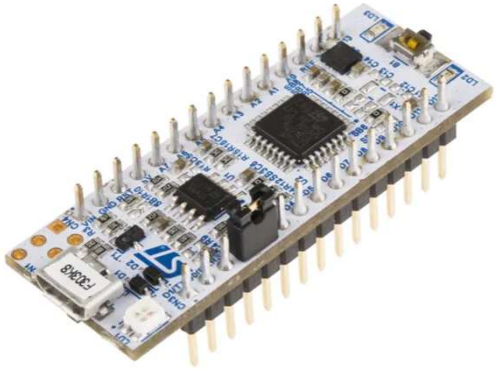
It was designed for very low-power microcontrollers, such as:

a) Arduino Nano 33 BLE Sense
b) STM32 boards
c) ESP32
d) NRF52 chips

What is The Micro-Speech Application?. . .

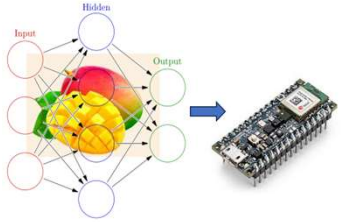


STM32 Boards



Nordic Semiconductor NRF52833 SoC

How Micro-Speech Works?



The following information will describe the working operation of how the Micro-Speech Works.

1. Audio Capture (Real-Time Streaming)

   Micro-Speech continuously reads audio from the microphone Analog-Digital Converter (ADC).

   Typical parameters:

   a) 16 kHz sampling

   b) 16-bit PCM

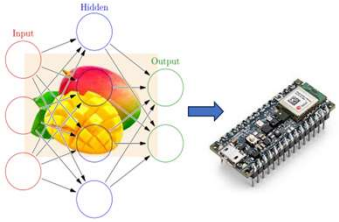
   c) 20–40 ms frames

Question 1

Which parameter is incorrect for the Audio Capture Step?

a) 32 KHz
b) 16-bit PCM
c) 20 – 40 ms frames
d) none of the above

How Micro-Speech Works?. . .

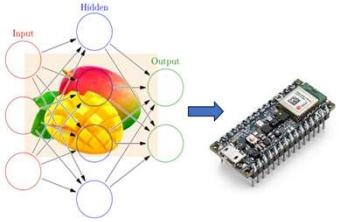


An example C++ code may use lines of instruction to capture audio as shown below:

```
audio_provider.cc
microphone audio acquisition
```

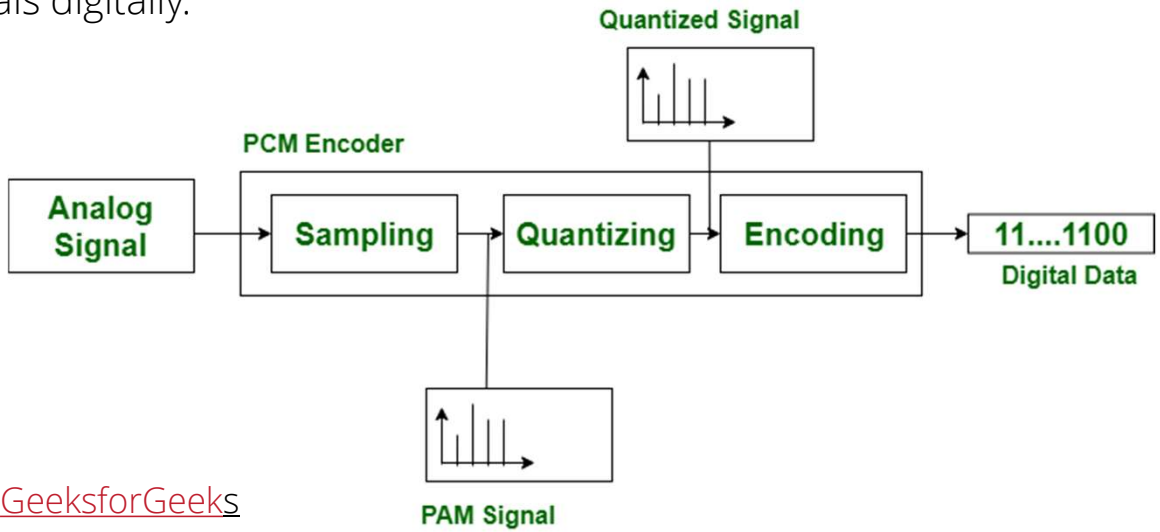
The raw Pulse Code Modulation PCM samples are pushed into a rolling circular buffer of about 1 second of audio.

How Micro-Speech Works?. . .



What is PCM?

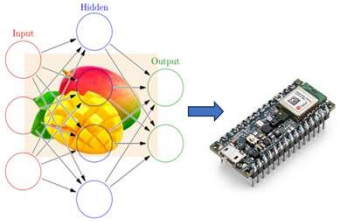
PCM stands for Pulse-Code Modulation — the most common way to represent analog audio signals digitally.



Block Diagram Of PCM

Image courtesy of GeeksforGeeks

How Micro-Speech Works?. . .



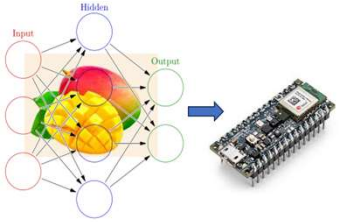
2. Feature Extraction – MFCCs

Speech recognition does **not** use raw audio directly.
Instead, Micro-Speech computes MFCCs – Mel Frequency Cepstral Coefficients.
Steps:

a) Split audio into frames (e.g., 30 ms with 10 ms overlap)
b) Apply Hamming window
c) Perform FFT
d) Apply Mel-scale filter banks
e) Convert power values to log scale
f) Compute Discrete Cosine Transform (DCT) → MFCC coefficients

The output is a 2D spectrogram-like feature map.

How Micro-Speech Works?. . .



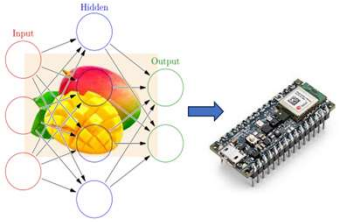
What is MFCCs?

MFCCs stand for Mel-Frequency Cepstral Coefficients.
They are one of the most widely used audio features in:

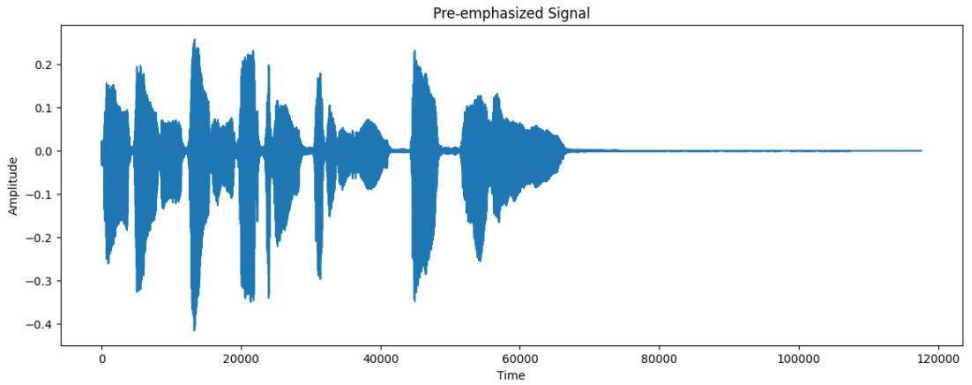
a) Speech recognition
b) Keyword spotting (like Micro-Speech “Yes/No”)
c) Speaker identification
d) Audio classification
e) TinyML audio models

MFCCs take raw PCM audio and convert it into a representation that a neural network can easily learn from.

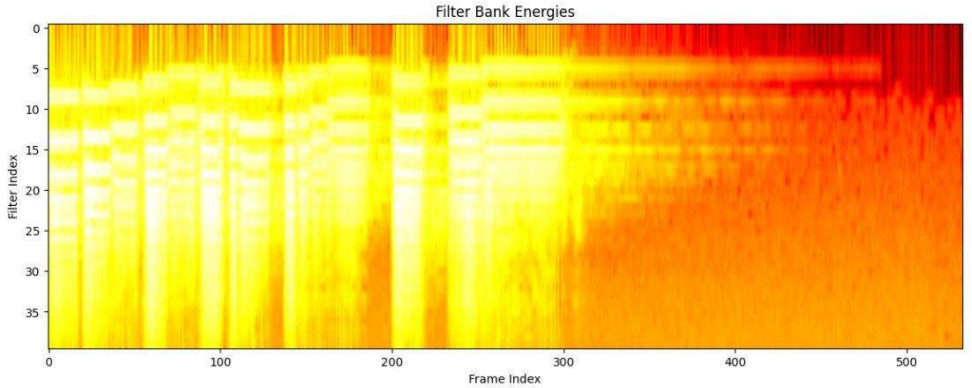
How Micro-Speech Works?. . .



Original Speech Signal



Mel Frequency Equivalent



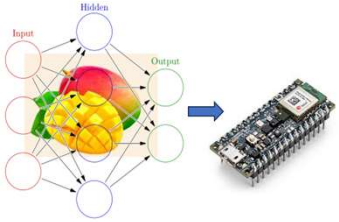
Images courtesy of GeeksforGeeks

Question 2

Which item is correct about MFCC?

a) Split audio into nuggets
b) Perform STT
c) Compute DCT→ MFCC Coefficients
d) none of the above

How Micro-Speech Works?. . .



3. TensorFlow Lite Micro Inference

Micro-Speech loads a small neural network:

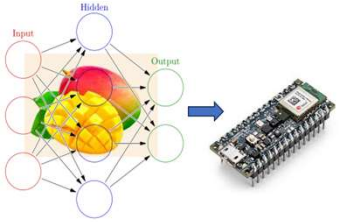
Model Architecture (typical)

A standard model has:

a) 1D Convolution (Conv) layer
b) Depthwise Conv
c) Average Pooling
d) Fully connected layer
e) Softmax output (4 classes)

Model size: ~18 KB

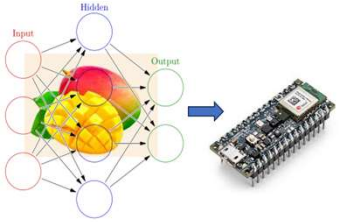
How Micro-Speech Works?. . .



The model is compiled as a static C-array:

```
micro_speech_model_data.cc
```

How Micro-Speech Works?. . .



4. Command Detection Logic (Smoothing & Debouncing)

Neural networks can produce noisy outputs.

Micro-Speech uses a command-recognizer algorithm to filter results:

It checks:

a) Consistent detection over multiple frames

b) Minimum confidence threshold

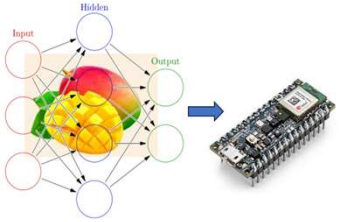
c) Suppresses false positives

d) Avoids “double triggers”

This is done in:

```
recognize_commands.cc
```

How Micro-Speech Works?. . .



5. Output: Trigger an Action

When a command is recognized, the main loop:

a) Turns on an LED
b) Prints “YES” / “NO” over serial
c) Calls a user-defined callback

Example in `main_functions.cc`:

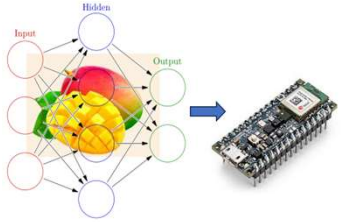
```
if (found_command == "yes") {
    light LED;
}
```

Question 3

For Step 5 "How Micro-Speech Works output Trigger an Action", what other device can be activated?

a) DC motor
b) a small light bulb
c) A seven-segment LED display
d) all of the above

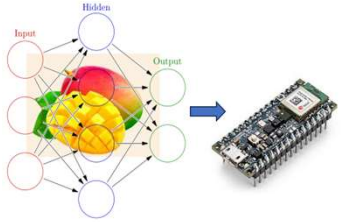
Why Micro-Speech Is Important Tiny ML?



Micro-Speech is the canonical example that demonstrates:

TinyML Challenge	Micro-Speech Feature
Running ML with tiny memory	10–20 KB RAM usage
Real-time signal processing	Audio FFT + MFCC
Embedded inference	TFLite Micro interpreter
Low-power keyword spotting	"always-on" listening
Model compression	Quantized 8-bit NN

Why Micro-Speech Is Important Tiny ML?. . .



It is used in:

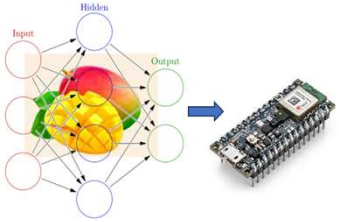
a) TinyML courses
b) Harvard CS249r
c) Edge impulse beginner labs
d) Arduino TensorFlow tutorials

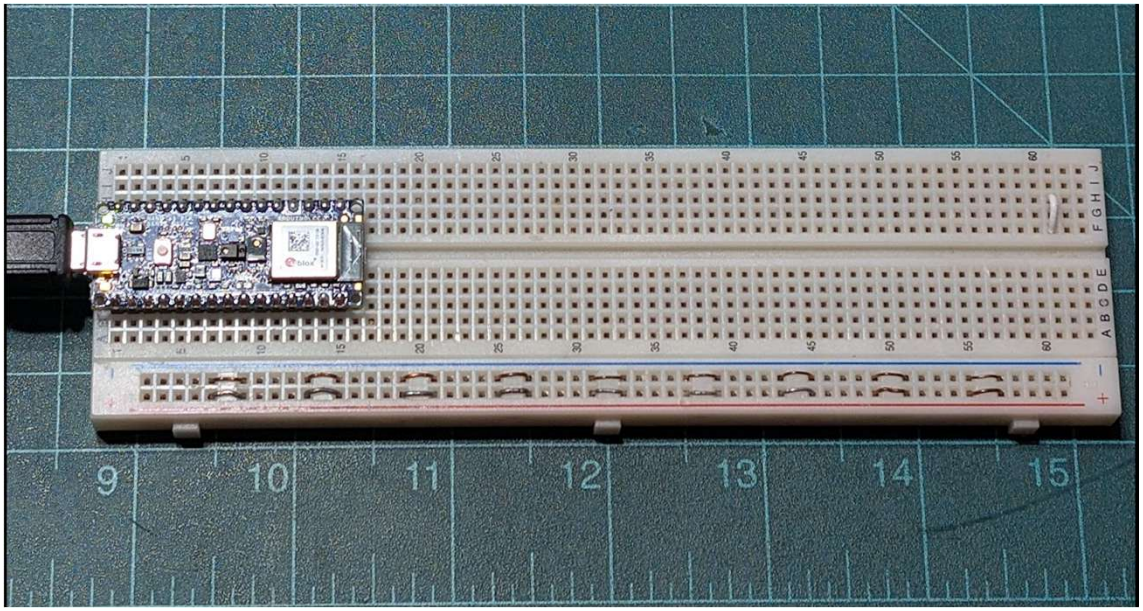
Question 4

Which item is incorrect regarding the importance that Micro-Speech can demonstrate?

a) 15-20 KB RAM usage
b) TFLite Macro interpreter
c) "always on" listening
d) none of the above

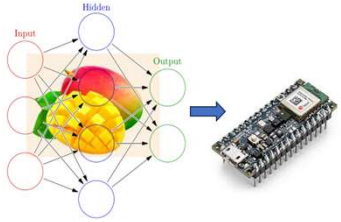
Lab: Hands-On With the Micro-Speech Application





```
Heard no (207) @24704ms
Heard no (203) @27728ms
Heard unknown (207) @29168ms
Heard no (205) @34640ms
Heard no (214) @37744ms
Heard yes (206) @42672ms
Heard no (201) @44640ms
Heard yes (203) @45520ms
Heard yes (217) @49808ms
Heard no (206) @50832ms
Heard no (201) @52656ms
Heard no (201) @54784ms
Heard no (210) @63824ms
```

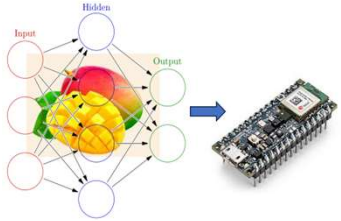
Lab: Hands-On With the Micro-Speech Application. . .



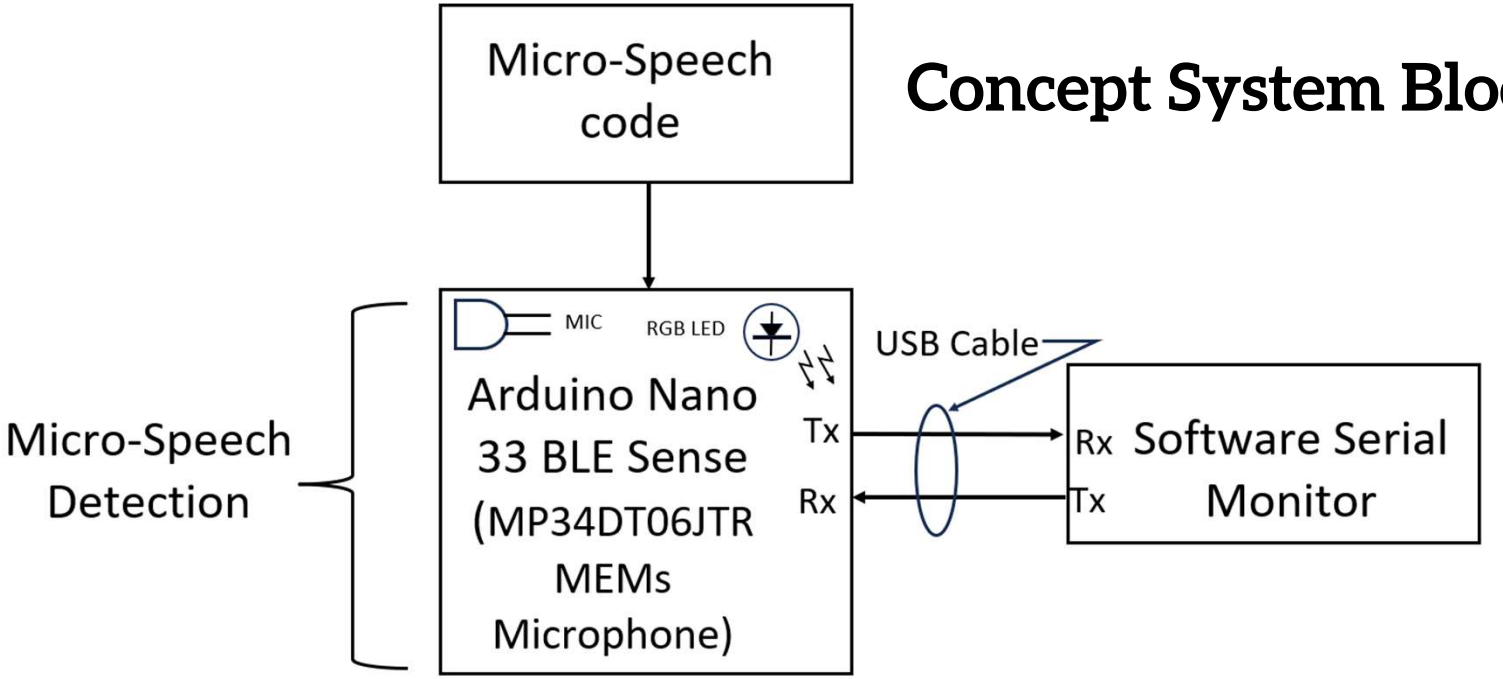
Lab Objectives:

- Participants will learn to set up a Micro-Speech System using the Arduino Nano 33 BLE Sense board and the TinyML classification C++ code.
- Participants will learn to identify the keywords" Yes" and "No" using the Arduino Nano 33 BLE Sense board's RGB LED.
- Participants will learn to visualize the keywords using a software serial monitor.

Lab: Hands-On With the Micro-Speech Application. . .

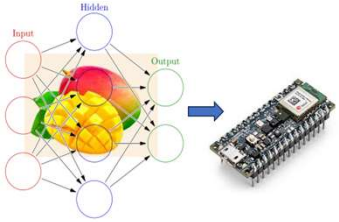


Concept System Block Diagram



Setup-Micro-Speech System

Arduino Nano 33 BLE Sense Overview. . .



Functional Overview

Version 2

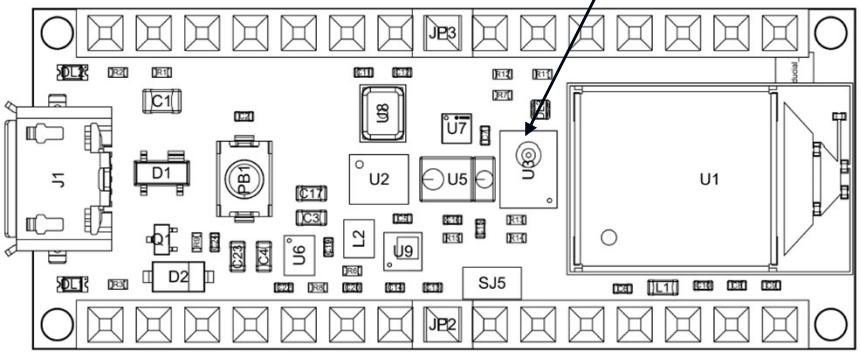
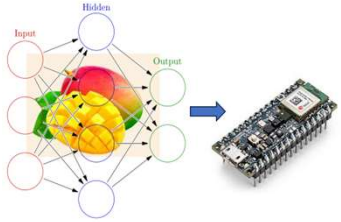


Illustration courtesy of Arduino.cc

Ref.	Description	Ref.	Description
U1	NINA-B306 Module Bluetooth® Low Energy 5.0 Module	U6	MP2322GQH Step Down Converter
U2	BMI270 Sensor IMU	PB1	IT-1185AP1C-160G-GTR Push button
U3	MP34DT06JTR MEMS Microphone	U8	HS3003 Humidity Sensor
U7	BMM150 Magnetometer IC	DL1	Led L
U5	APDS-9660 Ambient Module	DL2	Led Power
U9	LPS22HBTR Pressure Sensor IC		

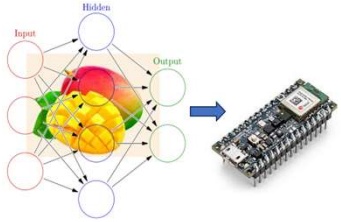
Lab: Hands-On With the Micro-Speech Application. . .




```
20  #ifndef ARDUINO_EXCLUDE_CODE
21
22  #include "Arduino.h"
23  #include "command_responder.h"
24  #include "tensorflow/lite/micro/micro_log.h"
25
26  // Toggles the built-in LED every inference, and lights a colored LED depending
27  // on which word was detected.
28  void RespondToCommand(int32_t current_time, const char* found_command,
29                        uint8_t score, bool is_new_command) {
30    static bool is_initialized = false;
31    if (!is_initialized) {
32      pinMode(LED_BUILTIN, OUTPUT);
33      // Pins for the built-in RGB LEDs on the Arduino Nano 33 BLE Sense
34      pinMode(LEDR, OUTPUT);
35      pinMode(LEDG, OUTPUT);
36      pinMode(LEDB, OUTPUT);
37      // Ensure the LED is off by default.
38      // Note: The RGB LEDs on the Arduino Nano 33 BLE
39      // Sense are on when the pin is LOW, off when HIGH.
40      digitalWrite(LEDR, HIGH);
41      digitalWrite(LEDG, HIGH);
42      digitalWrite(LEDB, HIGH);
43      is_initialized = true;
44    }
45    static int32_t last_command_time = 0;
46    static int count = 0;
```

Partial Arduino Command Responder C++ Code

Lab: Hands-On With the Micro-Speech Application. . .

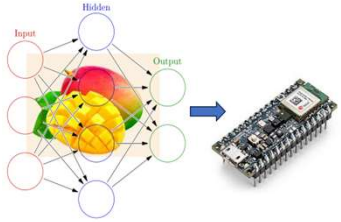


To access the Micro-Speech Application within the Arduino IDE:

Examples>File>Arduino TensorFlow Lite> micro_speech

Note:
To ensure the code will compile correctly, install the Arduino TensorFlow Lite library

Lab: Hands-On With the Micro-Speech Application. . .



Ambient and Keywords will be display on Serial Monitor

Output Serial Monitor ×

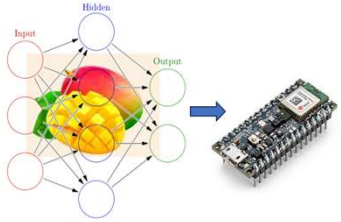
Message (Enter to send message to 'Arduino Nano 33 BLE' on 'COM12')

New Line 9600 baud

```
Heard no (233) @2625040ms
Heard unknown (205) @2627680ms
Heard unknown (208) @2629840ms
Heard unknown (203) @2631344ms
Heard unknown (202) @2635424ms
Heard unknown (205) @2639168ms
Heard unknown (204) @2641728ms
Heard unknown (209) @2644672ms
Heard unknown (214) @2647536ms
Heard unknown (204) @2652768ms
Heard yes (214) @2653728ms
Heard yes (218) @2656032ms
Heard yes (211) @2657936ms
Heard yes (208) @2659760ms
Heard no (212) @2661584ms
Heard no (206) @2663488ms
Heard no (204) @2664992ms
Heard unknown (204) @2671104ms
```

Arduino Nano 33 BLE on COM12 1

Lab: Hands-On With the Micro-Speech Application. . .



The README.md tab will provide additional information on how to train the Micro-Speech to recognize other keywords of choice.

Question 5

Which line of instruction allows changing the output device from the onboard LED to another actuator/display on slide 29?

a) 40
b) 35
c) 32
d) none of the above

Thank you for attending

Please consider the resources below:

[1] J. Lin, L. Zhu, W. M. Chen, W. C.Wang, and S. Han, "Tiny machine learning: Progress and futures," *arXiv*:2403.19076v2 [cs.LG], Jun. 2016. [Online]. Available: https://arxiv.org/abs/2403.19076

[2] R. Mathur, "A detailed intro to neural networks," Aug. 2023. [Online]. Available: https://rikinmathur.substack.com/p/a-detailed-intro-to-neural-networks

[3] S. Heydari, Q. H. Mahmoud, "Tiny machine learning and on-device inference: A survey of applications, challenges, and future directions," May. 2025. [Online]. Available: https://www.mdpi.com/1424-8220/25/10/3191

[4] D. Wilcher, "Designs News December 25 webinar code," GitHub repository, Dec. 2025. [Online]. Available: https://github.com/DWilcher/DesignNews-WebinarCode/blob/main/December_25_Webinar_Code.zip

[5] A. Dehghani, O. Sarbishei, T. Glatard, and E. Shihab, " A quantitative comparison of overlapping and non-overlapping sliding windows for human activity recognition using inertial sensors," Nov. 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/22/5026

CEC Continuing Education Center

DesignNews

Thank You

Sponsored by

DigiKey

informa markets