

Tech Talks LIVE Schedule – Presentation will begin shortly



NEW Wireless Connectivity Tech Talks



Tuesday, July 13	Learn to add Speech Recognition with Machine Learning
Tuesday, July 27	Simplify your Bluetooth Designs using Python Scripts
Tuesday, August 10	Quick Start your Bluetooth Designs for Pulse Oximetry and Electric Shelf Labels
Tuesday, August 24	Works With: Make the Most of WW 2021

**Respond to the poll to enter to win a
Thunderboard Sense 2**

Recording and slides will be posted to:
www.silabs.com/training

We will begin in **0:00**



tech **t▶lks**

WELCOME

Learn to add Speech Recognition with
Machine Learning

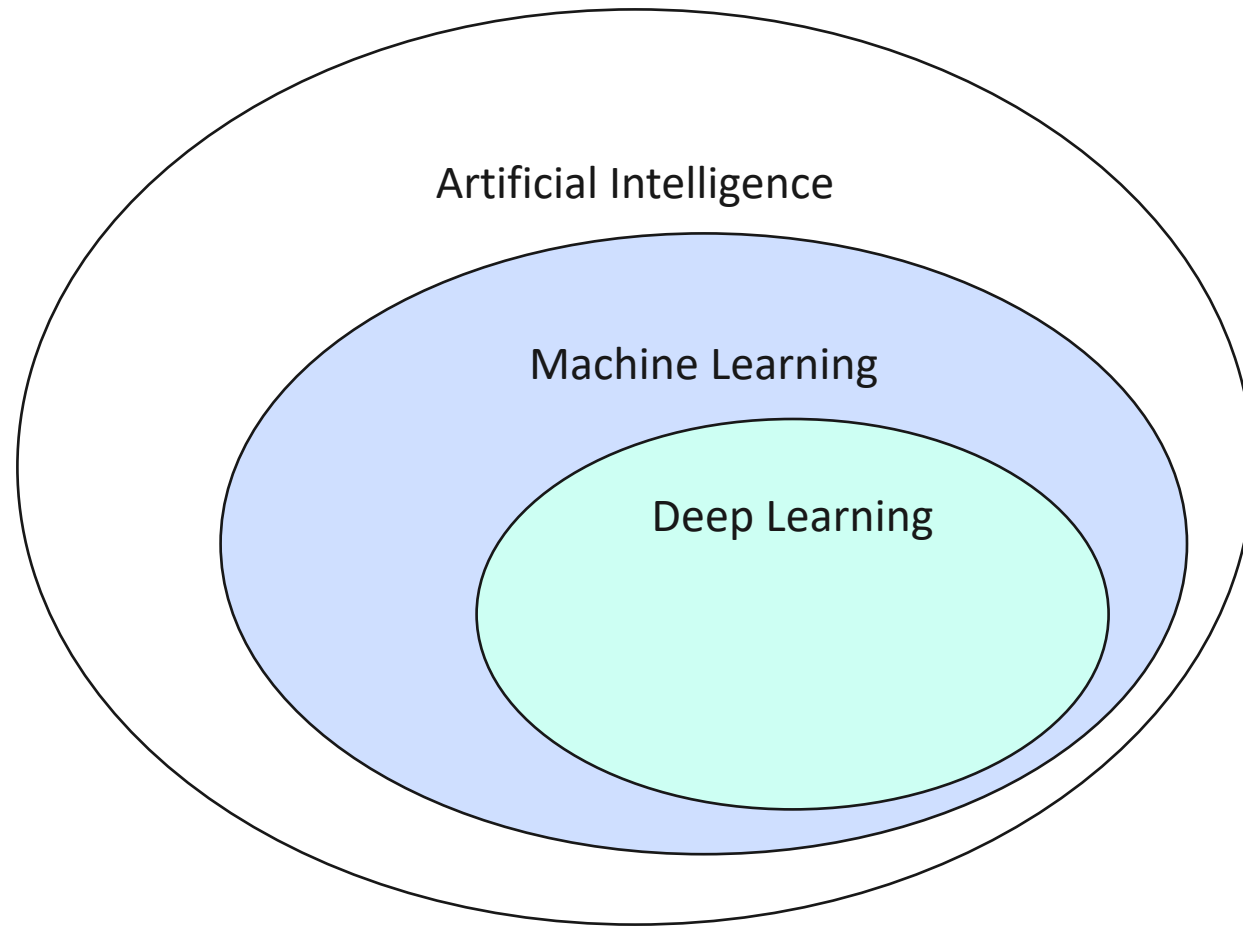
Andrew Krenz



Agenda

- **A brief introduction to Machine Learning on Embedded Devices**
- **Introduction to Edge Impulse and a high level overview on how to use their platform to create a neural network**
- **Demonstration: “wireless gecko” running on a Thunderboard Sense 2**
- **Wrap-up and Q&A**

Artificial Intelligence, Machine Learning, and Deep Learning



- **Artificial Intelligence:** Broadly defined as the effort to automate intellectual tasks normally performed by humans
- **Machine Learning:** A system which outputs predictions based on previous observations
- **Deep Learning:** A subset of Machine Learning in which the learning algorithmic architecture is based on approximations of how a human brain might work.

Embedded Machine Learning Use Cases



AUDIO

Wake Word / Key Phrase Detection
Glass Breaking
Intrusion



SENSING

Motion Sensing
Intelligent Sensor Fusion

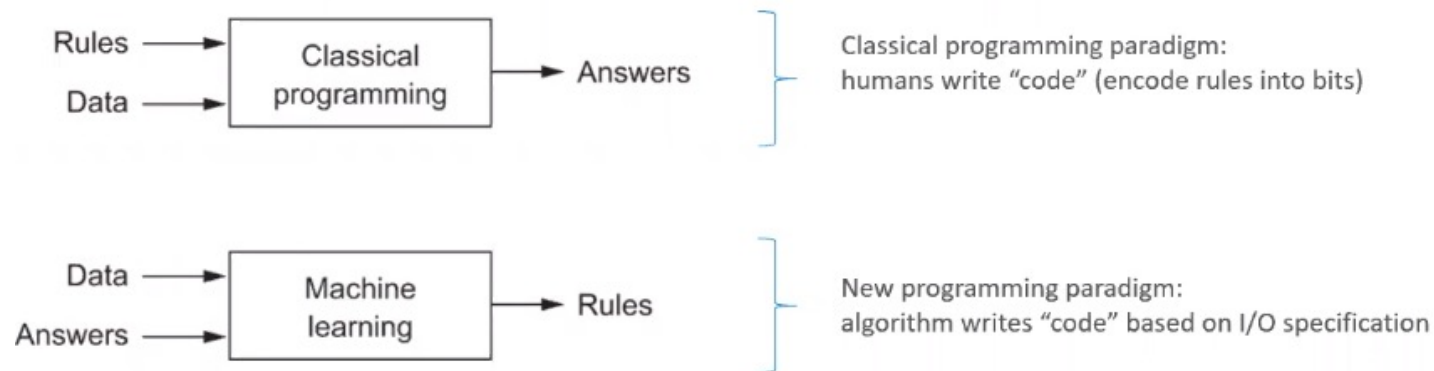


INDUSTRIAL

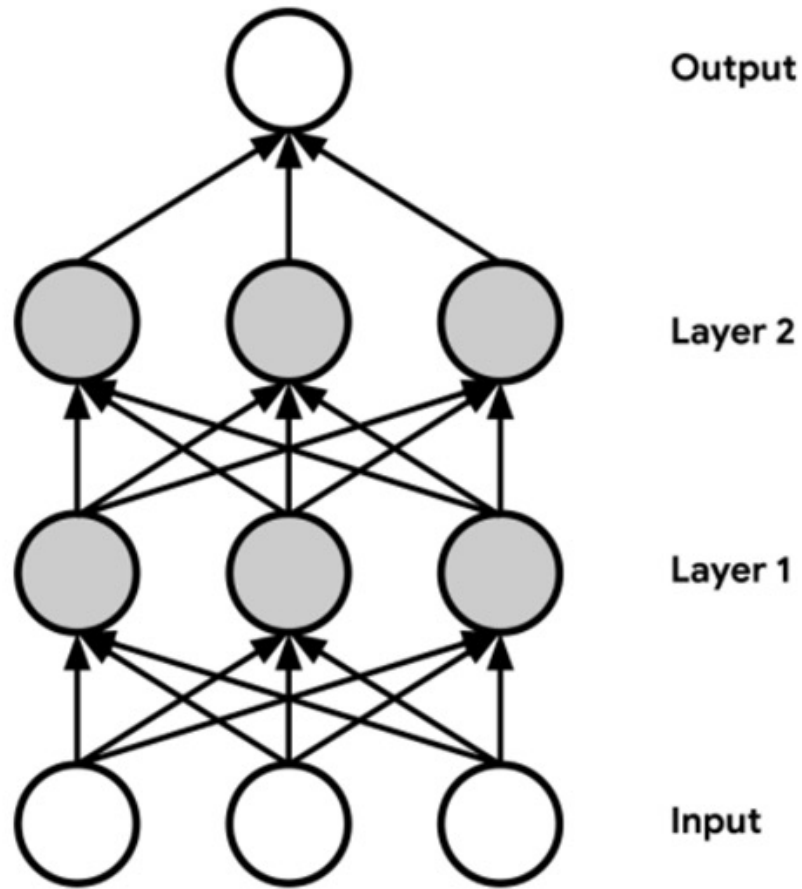
Predictive Maintenance

What is Machine Learning?

- **Definition - Technique for using computers to predict things based on past observations**
- **Collect data – paired input/output**
- **Design Model**
 - Create a computer program that analyzes that data, learns patterns
 - Computer uses the analysis to predict future states from new input
- **Training: Machine Learning model is trained rather than explicitly programmed with rules**
 - Model is presented with many examples of input/output sets
- **Inference: Model finds statistical structure in these examples and produces rules for automating the task**



Train the Model



A simple deep learning network with two layers

Source: TinyML by Daniel Situnayake; Pete Warden

▪ **Selecting Data**

- Train your model only using information relevant to solving the problem

▪ **Collecting Data**

- The more, the more varied, the better!
- 80% for training; 20% for testing

▪ **Labelling Data**

- Needed to classify the data

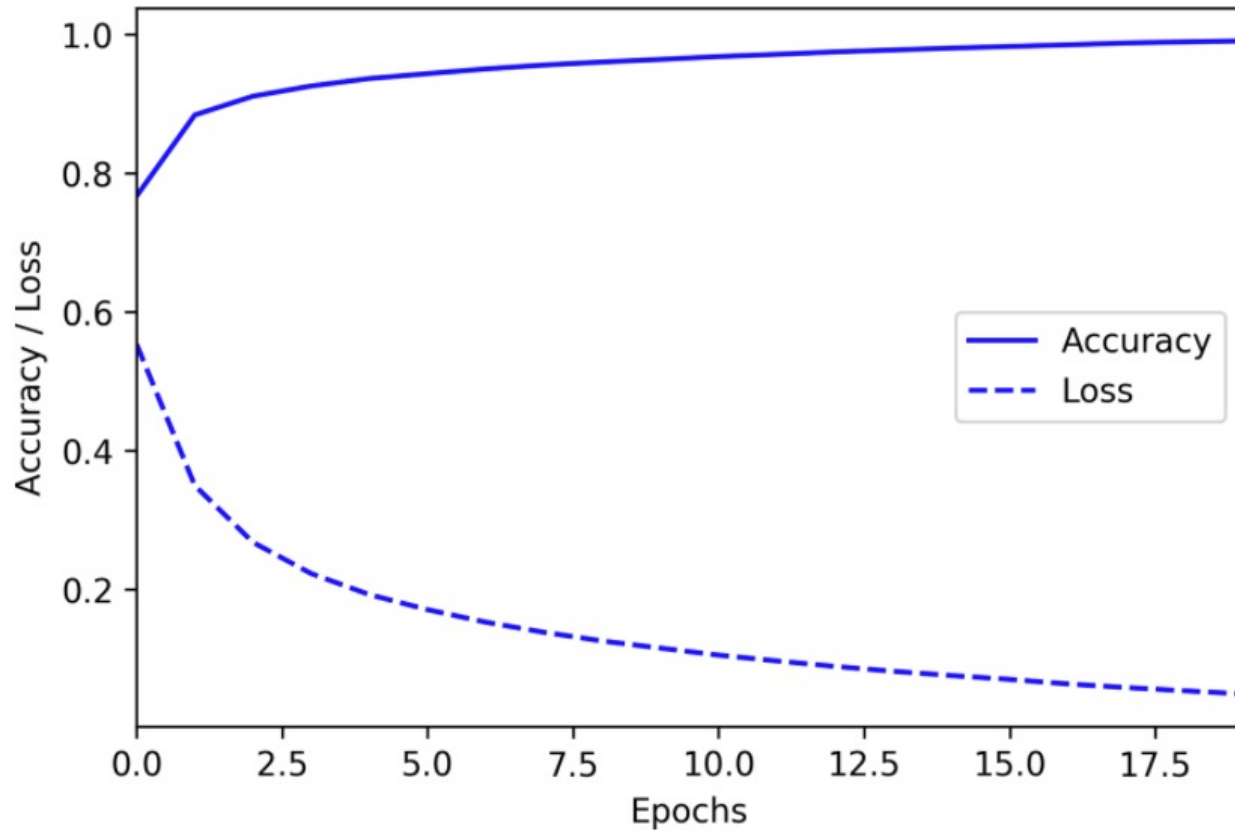
▪ **The idea is to feed training data through a model and make small adjustments**

- Weights and biases of each node are initialized with random values

▪ **Training is done in iterations (or epochs) via an algorithm called backpropagation**

- Weights and biases are adjusted iteratively
- Stops once the model's performance stops improving: The model "converges"

Training the Model Cont'd – Accuracy and Loss



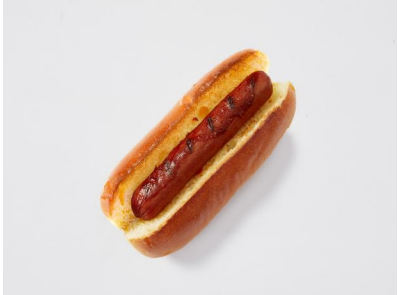
A graph showing model convergence during training

Source: TinyML by Daniel Situnayake; Pete Warden

- How to determine if a model has “converged”?
 - Loss and Accuracy
- Loss: gives a numerical estimate of how far the model is from producing the expected answers
- Accuracy: percentage of time that the model chooses the correct prediction
- Perfect Model: Loss of 0.0 and Accuracy of 100%
- As training progresses, accuracy increases and loss is reduced until the model no longer improves

Training Data Sets

Hot Dog



Not Hot Dog (Unknown)



Testing and Inferencing



hot dog



NOT hot dog



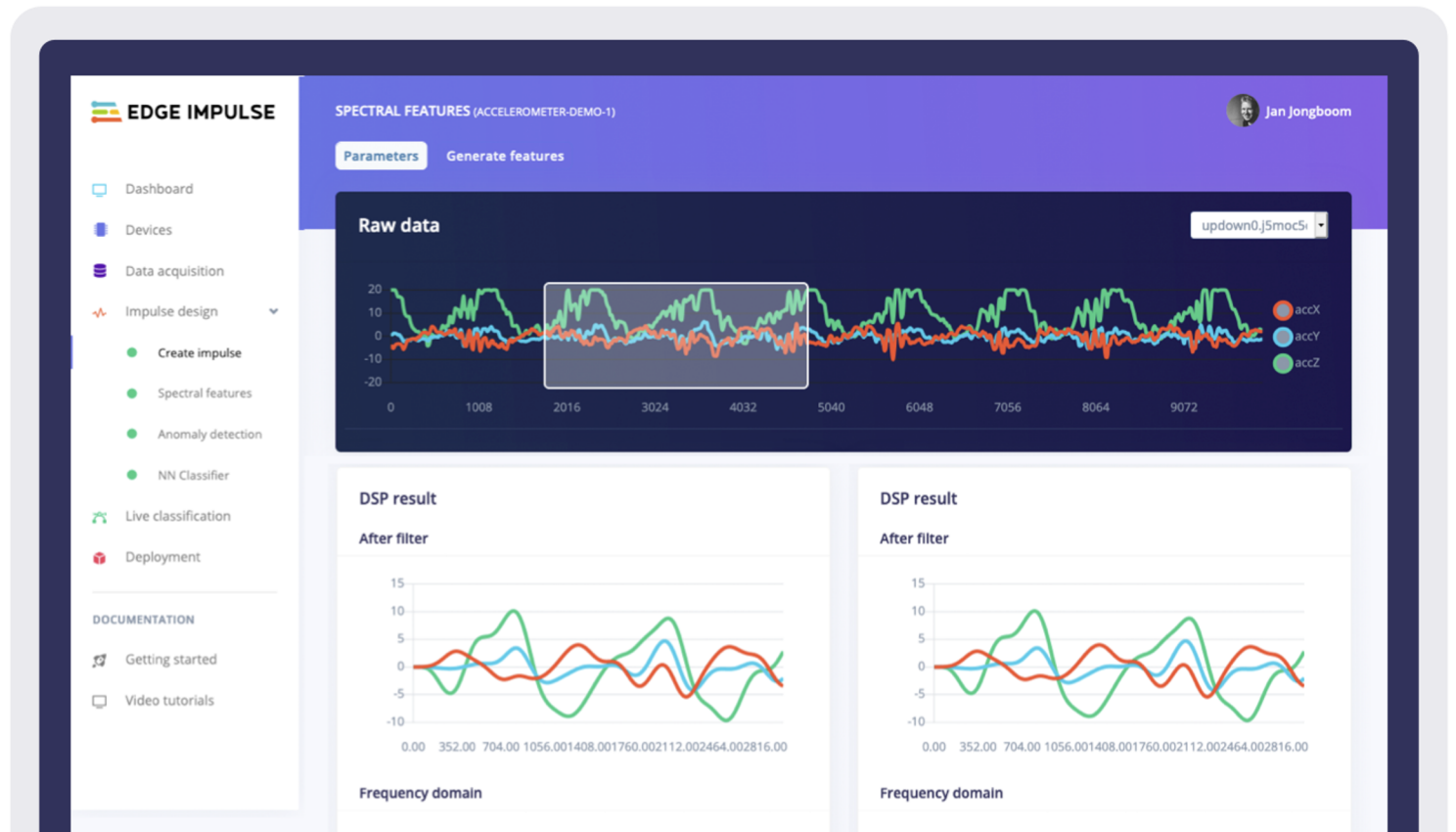
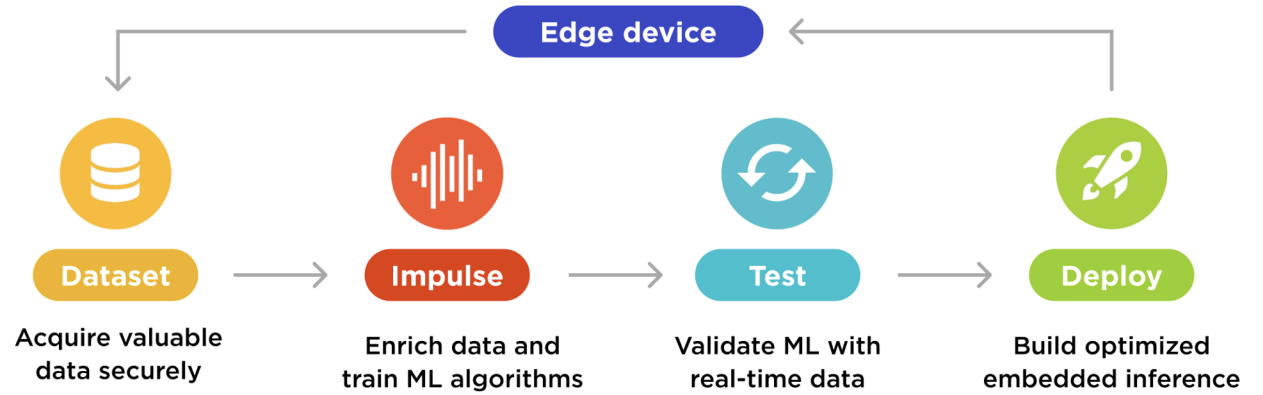
hot dog

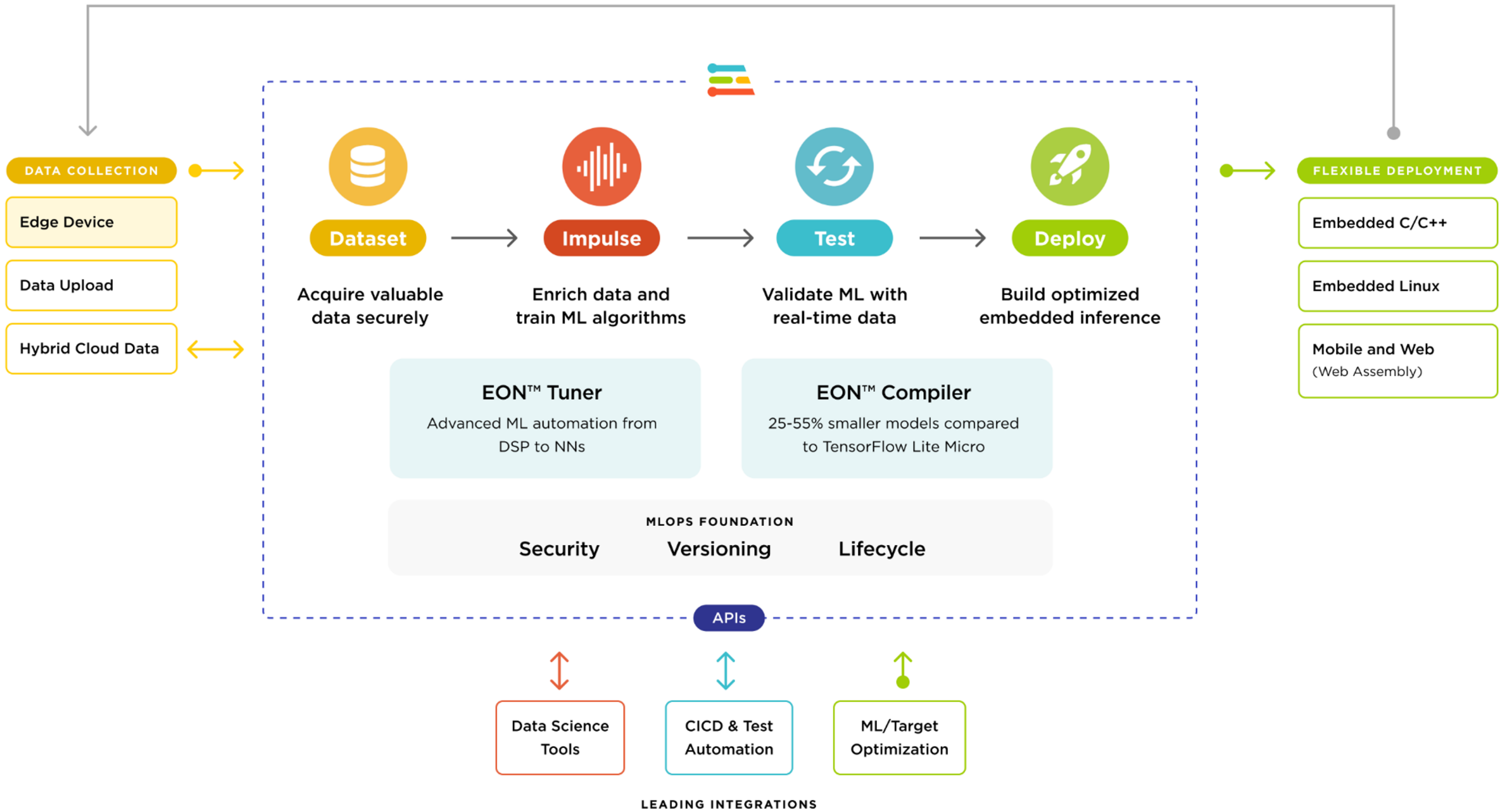
- Reserve 20% of your data set for testing
- A working model will won't be perfect, but should return acceptable results a high percentage of the time
- Poor results?
 - Check the input dataset for miscategorized data
 - Provide more input data
 - 30+ minutes of voice samples not uncommon for typical voice recognition

The leading embedded ML platform

Learn more at:

<http://edgeimpulse.com>





LEADING INTEGRATIONS

Closeup: Data Acquisition and Audio Sampling

EDGE IMPULSE

DATA ACQUISITION (WIRELESS GECKO) Andrew

Training data | Test data

Did you know? You can capture data from any device or development board, or upload your existing datasets - [Show options](#)

DATA COLLECTED: 25m 39s

LABELS: 2

Record new data [Connect using WebUSB](#)

No devices connected to the remote management API.

Collected data

SAMPLE NAME	LABEL	ADDED	LENGTH
wirelessgecko.299g...	wirelessgecko	Jun 29 2021, 14...	1s
wirelessgecko.299g...	wirelessgecko	Jun 29 2021, 14...	1s
wirelessgecko.299g...	wirelessgecko	Jun 29 2021, 14...	1s
wirelessgecko.299g...	wirelessgecko	Jun 29 2021, 14...	1s
wirelessgecko.299g...	wirelessgecko	Jun 29 2021, 14...	1s
wirelessgecko.299g...	wirelessgecko	Jun 29 2021, 14...	1s
wirelessgecko.299g...	wirelessgecko	Jun 29 2021, 14...	1s
wirelessgecko.299g...	wirelessgecko	Jun 29 2021, 14...	1s
wirelessgecko.299g...	wirelessgecko	Jun 29 2021, 14...	1s

RAW DATA

wirelessgecko.299g1ps5.s5

30000
25000
20000
15000
10000
5000
0
-5000
-10000
-15000
-20000

0 103 207 310 414 518 621 725 829 932

audio

0:00 / 0:00

Closeup: Design the Impulse

EDGE IMPULSE

CREATE IMPULSE (WIRELESS GECKO) Andrew Kr

An impulse takes raw data, uses signal processing to extract features, and then uses a learning block to classify new data.

- Time series data
 - Axes: audio
 - Window size: 1000 ms.
 - Window increase: 500 ms.
 - Zero-pad data:
- Audio (MFCC)
 - Name: MFCC
 - Input axes: audio
- Neural Network (Keras)
 - Name: NN Classifier
 - Input features: MFCC
 - Output features: 2 (unknown, wirelessgecko)
- Output features
 - 2 (unknown, wirelessgecko)

Save Impulse

GETTING STARTED


Closeup: MFCC and Feature Extraction

MFCC (WIRELESS GECKO) Andrew Kr

Parameters Generate features

Raw data

0:00 / 0:00 wirelessgecko.299g1ps5.s5 (wirelessgec)



audio

Raw features 📄

16, 10, 17, 26, 28, 31, 32, 8, -4, -2, -13, -23, -34, -38, -31, -41, -39, -46,...

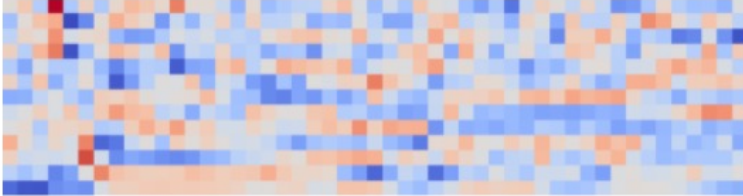
Parameters

Mel Frequency Cepstral Coefficients

Number of coefficients	<input type="text" value="13"/>
Frame length	<input type="text" value="0.02"/>
Frame stride	<input type="text" value="0.02"/>

DSP result

Cepstral Coefficients



Processed features 📄

-2.3155, -0.3726, 0.7762, 1.5181, 0.3246, 0.0234, -1.2595, 0.7454, -0.5266, 0.1...

Closeup: MFCC and Feature Extraction

The screenshot displays the Edge Impulse interface for MFCC (Wireless Gecko) feature extraction. The interface is divided into several sections:

- Left Sidebar:** Contains navigation options such as Dashboard, Devices, Data acquisition, Impulse design, Create impulse, MFCC, NN Classifier, Retrain model, Live classification, Model testing, Versioning, and Deployment. It also includes a 'GETTING STARTED' section with Documentation and Forums.
- Top Header:** Shows the project name 'MFCC (WIRELESS GECKO)' and the user's name 'Andrew K'.
- Parameters Section:** A 'Generate features' button is visible. Below it, the 'Training set' parameters are listed:

Parameter	Value
Data in training set	25m 39s
Classes	2 (unknown, wirelessgecko)
Window length	1000 ms.
Window increase	500 ms.
Training windows	1,391

A green 'Generate features' button is located at the bottom of this section.
- Feature explorer (1,493 samples):** A 3D scatter plot showing the distribution of features. The X, Y, and Z axes are labeled 'Visualization layer'. The plot shows two clusters of points: blue points representing 'unknown' and orange points representing 'wirelessgecko'. A legend on the left identifies these clusters.
- Bottom Right:** A small audio waveform visualization for a sample labeled 'Wireless Gecko.wire...' with a label 'Label: wirelessgecko'. The waveform shows amplitude over time, with a scale from 0 to 10000.

Closeup: Training the NN

EDGE IMPULSE

- Dashboard
- Devices
- Data acquisition
- Impulse design
- Create impulse
- MFCC
- NN Classifier
- Retrain model
- Live classification
- Model testing
- Versioning
- Deployment

GETTING STARTED

- Documentation
- Forums

Neural Network settings

Training settings

Number of training cycles

Learning rate

Minimum confidence rating

Audio training options

Data augmentation

Add noise None Low High

Mask time bands None Low High

Mask frequency bands None Low High

Warp time axis

Neural network architecture

Architecture presets 1D Convolutional (Default) 2D Convolutional

Input layer (650 features)

Reshape layer (13 columns)

1D conv / pool layer (8 neurons, 3 kernel size, 1 layer)

Dropout (rate 0.25)

Training output

Model

Model version:

Last training performance (validation set)

ACCURACY **99.0%** LOSS **0.03**

Confusion matrix (validation set)

	UNKNOWN	WIRELESSGECKO
UNKNOWN	98.8%	1.2%
WIRELESSGECKO	0.8%	99.2%
F1 SCORE	0.99	0.99

Feature explorer (full training set)

- unknown - correct
- wirelessgecko - correct
- unknown - incorrect
- wirelessgecko - incorrect

Visualization layer 1, Visualization layer 2, Visualization layer 3

On-device performance

INFERRING ... **8 ms.** PEAK RAM US... **4.4K** FLASH USAGE **30.0K**

Initial Test Results

EDGE IMPULSE

- Dashboard
- Devices
- Data acquisition
- Impulse design
 - Create impulse
 - MFCC
 - NN Classifier
- Retrain model
- Live classification
- Model testing**
- Versioning
- Deployment

GETTING STARTED

- Documentation
- Forums

MODEL TESTING (WIRELESS GECKO)

This lists all test data. You can manage this data through Data acquisition.

Test data Classify all

Set the 'expected outcome' for each sample to the desired outcome to automatically score the impulse.

SAMPLE NA...	EXPECTED OUT...	LENG...	ACCURACY	RESULT
wirelessg...	wirelessgecko	1s	100%	1 wirelessgecko
wirelessg...	wirelessgecko	1s	100%	1 wirelessgecko
wirelessg...	wirelessgecko	1s	100%	1 wirelessgecko
wirelessg...	wirelessgecko	1s	100%	1 wirelessgecko
wirelessg...	wirelessgecko	1s	100%	1 wirelessgecko
wirelessg...	wirelessgecko	1s	100%	1 wirelessgecko
wirelessg...	wirelessgecko	1s	100%	1 wirelessgecko
wirelessg...	wirelessgecko	1s	100%	1 wirelessgecko
wirelessg...	wirelessgecko	1s	100%	1 wirelessgecko
wirelessg...	wirelessgecko	1s	100%	1 wirelessgecko
wirelessg...	wirelessgecko	1s	100%	1 wirelessgecko
wirelessg...	wirelessgecko	1s	100%	1 wirelessgecko
wirelessg...	wirelessgecko	1s	100%	1 wirelessgecko
wirelessg...	wirelessgecko	1s	100%	1 wirelessgecko
wirelessg...	wirelessgecko	1s	100%	1 wirelessgecko
wirelessg...	wirelessgecko	1s	100%	1 wirelessgecko

Feature explorer ?

- unknown - correct
- wirelessgecko - correct
- unknown - incorrect
- wirelessgecko - incorrect

Model te

FILES HAVE
Generating

Classifyin
Copying fe
Copying fe
Classifyin
Scheduling
Job starte
Classifyin

Job comple

Model te

ACCURACY %

97.85%

	UNKNOWN	WIRELESS...	UNKNOW...	WIRELESS...	UNC
UNKNOWN	97.3%	1.6%	0%	0%	1.1%
WIRELESSGE...	0.6%	55.5%	0%	43.9%	0%
UNKNOWN (...)	-	-	-	-	-
WIRELESSGE...	-	-	-	-	-

Deployment

EDGE IMPULSE

Dashboard

Devices

Data acquisition

Impulse design

Create impulse

MFCC

NN Classifier

Retrain model

Live classification

Model testing

Versioning

Deployment

GETTING STARTED

Documentation

Select optimizations (optional)

Model optimizations can increase on-device performance but may reduce accuracy. Click below to analyze optimizations and see the recommended choices for your target. Or, just click Build to use the currently selected options.



Enable EON™ Compiler

Same accuracy, up to 50% less memory. Open source.



Available optimizations for NN Classifier

	RAM USAGE	LATENCY	CONFUSION MATRIX		
Quantized (int8) ★ Currently selected	4.4K	8 ms	97.3	1.6	1.1
	FLASH USAGI	ACCURACY	0.8	99.2	0
	30.0K	98.06%			
This optimization is recommended for best performance.					
Unoptimized (float32) Click to select	8.0K	35 ms	97.3	1.6	1.1
	FLASH USAGI	ACCURACY	0.8	99.2	0
	32.4K	98.06%			

Estimate for Cortex-M4F 40MHz (SiLabs Thunderboard Sense 2)

Build

Testing on Hardware: Thunderboard Sense 2 – SLTB004A

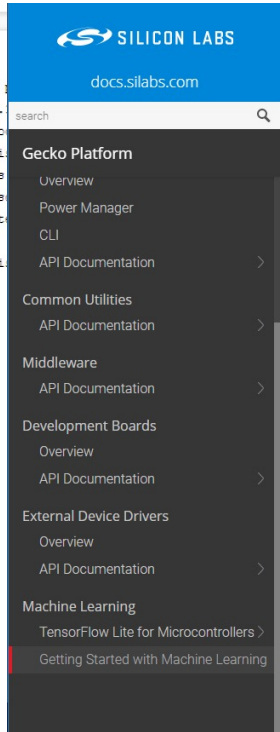
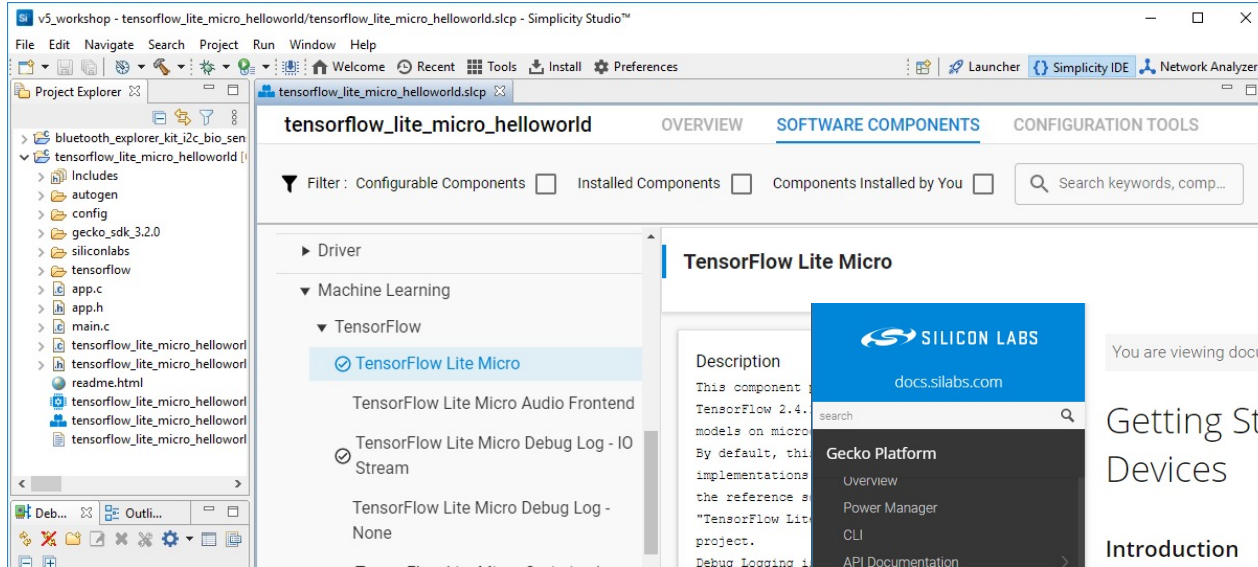


▪ Thunderboard Sense 2

- Wireless SoC with multi-protocol radio
- Direct support with Edge Impulse
- ARM® Cortex® M4 core with 256 kB RAM and 1024 kB Flash
- Broad Range of Sensors
 - 6-axis Inertial Sensor ICM-20648
 - Digital Microphone ICS-43434
 - Pressure Sensor BMP280
 - Indoor Air Quality and Gas Sensor CCS811
 - Relative Humidity and Temperature Sensor Si7021
 - UV and Ambient Light Sensor Si1133
 - Hall-effect Sensor Si7210

DEMO

TensorFlow Lite for Microcontrollers



You are viewing documentation for version: 3.2 (latest) | [Version History](#)

Getting Started with Machine Learning on Silicon Labs Devices

Introduction

Silicon Labs integrates [TensorFlow](#) as a component within our Gecko SDK and [Project Configurator](#) for our EFX32 series microcontrollers, making it simple to add machine learning capability to any application. This guide covers how to get started using TensorFlow Lite for Microcontrollers on Silicon Labs' EFX32 devices.

TensorFlow Lite for Microcontrollers

[TensorFlow](#) is a widely used deep learning framework, with capability for developing and executing neural networks across a variety of platforms. [TensorFlow Lite](#) provides an optimized set of tools specifically catered towards machine learning for mobile and embedded devices.

[TensorFlow Lite for Microcontrollers](#) (TFLM) specifically provides a C++ library for running machine learning models in embedded environments with tight memory constraints. Silicon Labs provides tools and support for loading and running pre-trained models that are compatible with this library.

Gecko SDK TensorFlow Integration

The [Gecko SDK](#) includes TensorFlow as a third-party submodule, allowing for easy integration and testing with Silicon Labs' projects. Note that the included TensorFlow version may differ from the latest release of TensorFlow.

Additionally, [TensorFlow Software Components](#) in the [Project Configurator](#) simplify the process of including the necessary dependencies to use TFLM in a project.

Developing a Machine Learning Model in TFLM

Resources

- **Edge Impulse**
 - <https://www.edgeimpulse.com/>
- **Thunderboard Sense 2**
 - <https://www.silabs.com/development-tools/thunderboard/thunderboard-sense-two-kit>
- **Documentation & Getting Started**
 - https://docs.silabs.com/gecko-platform/latest/machine_learning/tensorflow/overview
 - https://docs.silabs.com/gecko-platform/latest/machine_learning/tensorflow/ml_getting_started_guide
- **Machine Learning Resources at Silicon Labs**
 - <https://www.silabs.com/solutions/artificial-intelligence-machine-learning>
- **TensorFlow Lite**
 - <https://www.tensorflow.org/lite>
- **FREE Online Course: Embedded Machine Learning**
 - <https://www.coursera.org/learn/introduction-to-embedded-machine-learning>

Join our next Tech Talk

A promotional banner for a tech talk. The background is a dark blue geometric pattern overlaid on a photograph of a person with long dark hair sitting at a desk, working on a laptop. The person's hands are on the keyboard. A large monitor in the background displays code. A green cup is on the desk. The text is white and light blue. The date 'JULY 27TH' is underlined. The title 'Simplify your Bluetooth Designs using Python Scripts' is in a large font. The Silicon Labs logo is at the bottom left. The 'tech talks' logo is in the top right, with 'tech' in white and 'talks' in a white speech bubble. A light blue button with 'REGISTER TODAY' is centered below the banner.

JULY 27TH

tech talks

Simplify your
Bluetooth Designs
using Python Scripts

 SILICON LABS

REGISTER TODAY



works with

BY SILICON LABS

VIRTUAL CONFERENCE



When September 14–15th

Where Accessible live and online from anywhere in the world

Who Developer conference that brings together the biggest names in smart home technology

Why Developer's will learn how to develop and deliver IoT devices directly from the engineers who are building the latest advances

What Live. Free. All-online

[Register Today](#)

[View Agenda](#)

workswith.silabs.com





tech **t▶lks**

Q&A





Thank You!

