

tinyML[®] Talks

Enabling Ultra-low Power Machine Learning at the Edge

“Introduction to Edge Computing”

Archana Vaidheeswaran - Women Who Code

Singapore Area Group – February 14, 2021



www.tinyML.org



tinyML Talks Sponsors



tinyML Strategic Partner



EDGE IMPULSE



maxim
integrated™



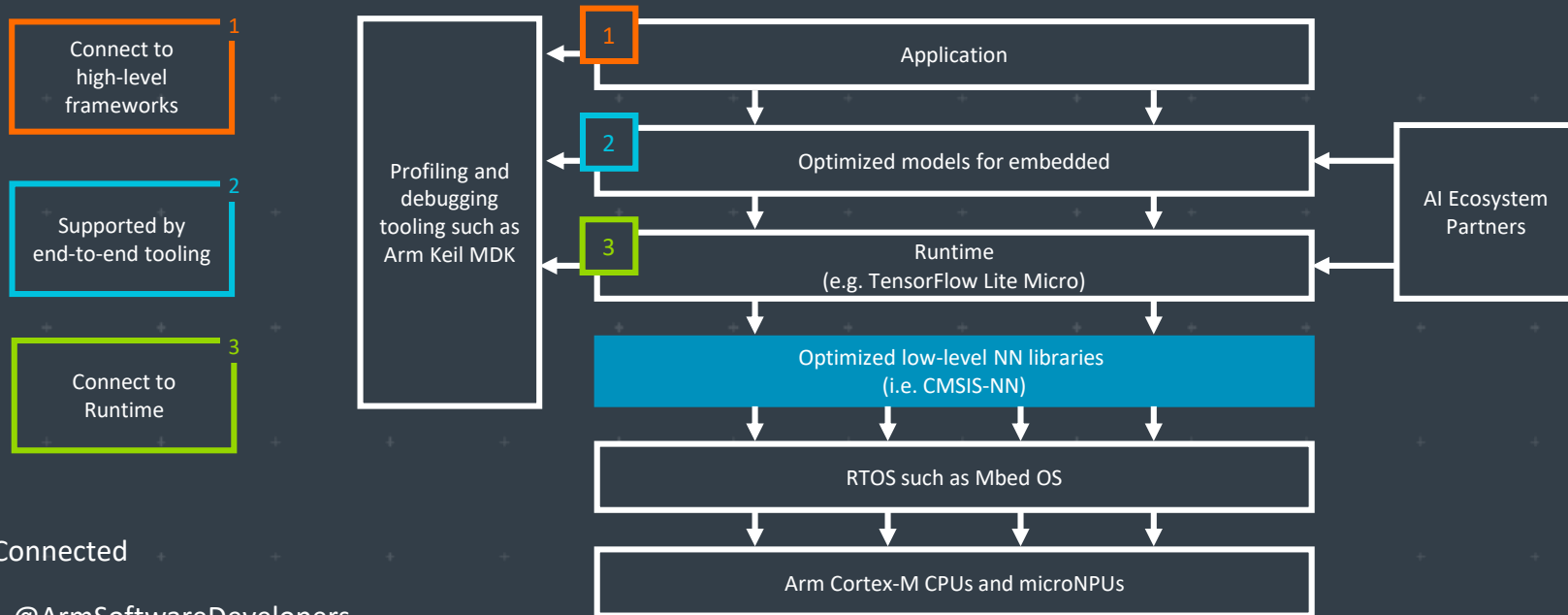
Reality AI®



SynSense

Additional Sponsorships available – contact Bette@tinyML.org for info

Arm: The Software and Hardware Foundation for tinyML



Stay Connected



@ArmSoftwareDevelopers



@ArmSoftwareDev

Resources: developer.arm.com/solutions/machine-learning-on-arm



WE USE AI TO MAKE OTHER AI FASTER, SMALLER AND MORE POWER EFFICIENT



Automatically compress SOTA models like MobileNet to <200KB with **little to no drop in accuracy** for inference on resource-limited MCUs



Reduce model optimization trial & error from weeks to days using Deeplite's **design space exploration**



Deploy more models to your device without sacrificing performance or battery life with our **easy-to-use software**

BECOME BETA USER bit.ly/testdeeplite

mobilityXlab

arm



TinyML for all developers



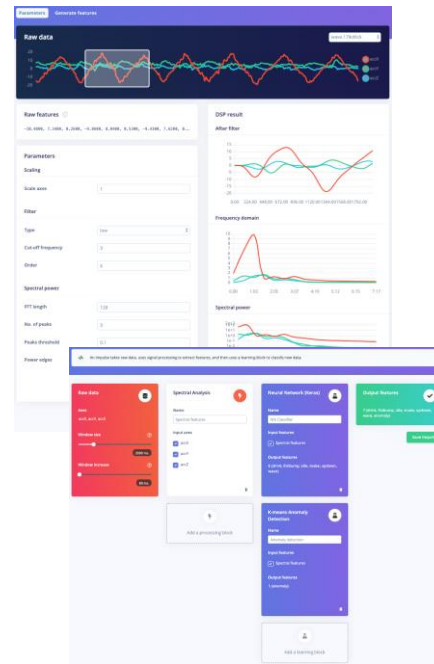
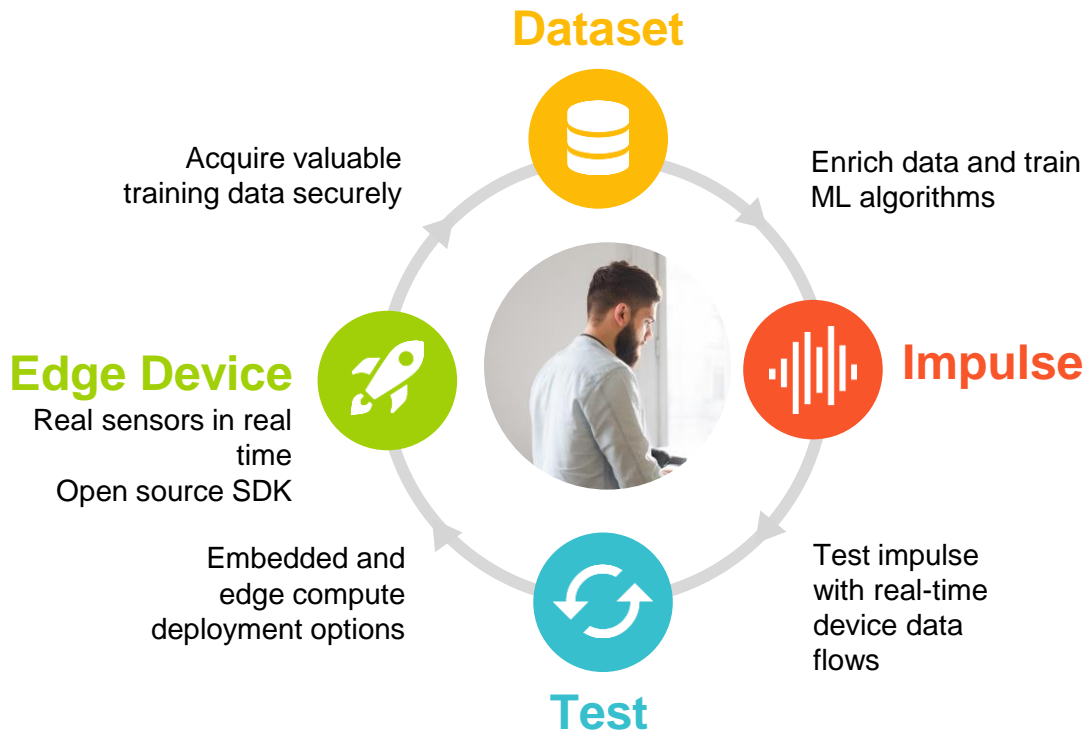
C++ library



Arduino library



WebAssembly



Maxim Integrated: Enabling Edge Intelligence

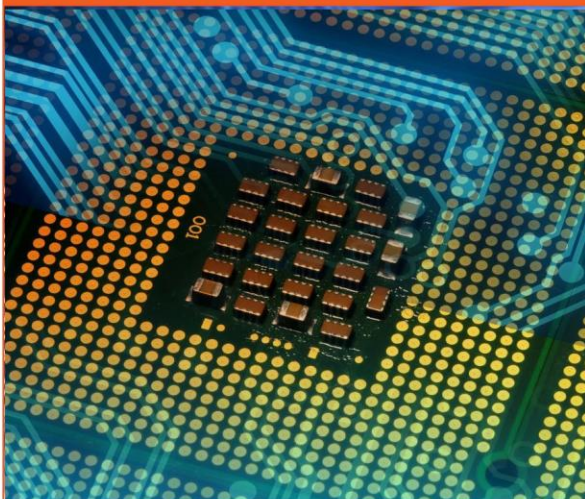
www.maximintegrated.com/ai

Sensors and Signal Conditioning



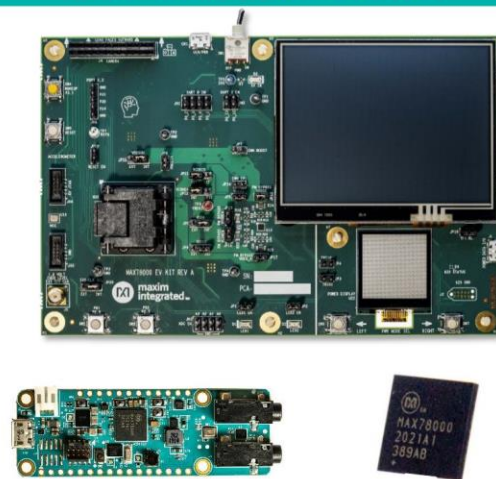
Health sensors measure PPG and ECG signals critical to understanding vital signs. Signal chain products enable measuring even the most sensitive signals.

Low Power Cortex M4 Micros



The biggest (3MB flash and 1MB SRAM) and the smallest (256KB flash and 96KB SRAM) Cortex M4 microcontrollers enable algorithms and neural networks to run at wearable power levels

Advanced AI Acceleration



The new MAX78000 implements AI inferences at over 100x lower energy than other embedded options. Now the edge can see and hear like never before.

Qeexo AutoML for Embedded AI

Automated Machine Learning Platform that builds tinyML solutions for the Edge using sensor data



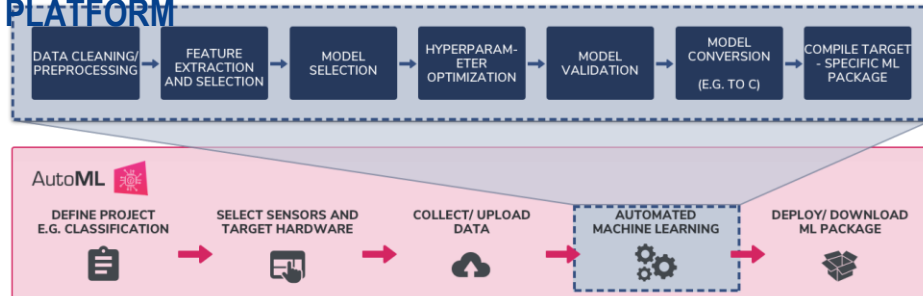
Key Features

- Wide range of ML methods: GBM, XGBoost, Random Forest, Logistic Regression, Decision Tree, SVM, CNN, RNN, CRNN, ANN, Local Outlier Factor, and Isolation Forest
- Easy-to-use interface for labeling, recording, validating, and visualizing time-series sensor data
- On-device inference optimized for low latency, low power consumption, and a small memory footprint
- Supports Arm® Cortex™- M0 to M4 class MCUs
- Automates complex and labor-intensive processes of a typical ML workflow – no coding or ML expertise required!

Target Markets/Applications

- Industrial Predictive Maintenance
- Automotive
- Smart Home
- Mobile
- Wearables
- IoT

QEE XO AUTO ML: END-TO-END MACHINE LEARNING PLATFORM



For a limited time, sign up to use Qeexo AutoML at automl.qeexo.com for FREE to bring intelligence to your devices!



is for
building products

<https://reality.ai>



info@reality.ai



[@SensorAI](https://twitter.com/SensorAI)



[Reality AI](#)

Reality AI Tools[®] software

Automated Feature
Exploration and
Model Generation

Bill-of-Materials
Optimization

Automated Data
Assessment

Edge AI / TinyML
code for the smallest
MCUs

Reality AI solutions

Automotive sound recognition & localization

Indoor/outdoor sound event recognition

RealityCheck™ voice anti-spoofing



SynSense

SynSense builds **ultra-low-power** (sub-mW) **sensing and inference** hardware for **embedded, mobile and edge** devices. We design systems for **real-time always-on smart sensing**, for audio, vision, IMUs, bio-signals and more.

<https://SynSense.ai>



Next tinyML Talks

Date	Presenter	Topic / Title
Tuesday, February 16	Mohammed Zubair PhD, SMIEEE Associate Professor / Consultant Department of Electrical Engineering / Center for Artificial Intelligence King Khalid University	Oral Tongue Lesion Detection using TinyML on Embedded Devices

Webcast start time is 8 am Pacific time

Please contact talks@tinymml.org if you are interested in presenting

Local Committee in Singapore

- Soham Chatterjee
 - Machine Learning Engineer at Sleek, Singapore
- Archana Vaidheeswaran
 - AI Engineer at Continental Automotive, Singapore



Reminders

Slides & Videos will be posted
tomorrow

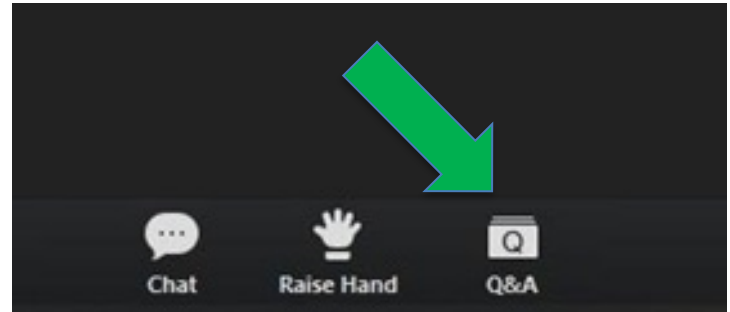


tinymml.org/forums



youtube.com/tinymml

Please use the Q&A window for your
questions





Archana Vaidheeswaran



Archana is an AI Engineer. She works in the field of automation, NLP and edge computing. She has co-created a nanodegree with Intel and Udacity: Intel AI on Edge Nanodegree.

Introduction to Deep Learning for Edge Devices

Vaidheeswaran Archana
&
Soham Chatterjee

OUR MISSION

Inspiring women to
excel in technology
careers.

WOMEN WHO
CODE



OUR VISION

A world where women are representative as technical executives, founders, VCs, board members and software engineers.

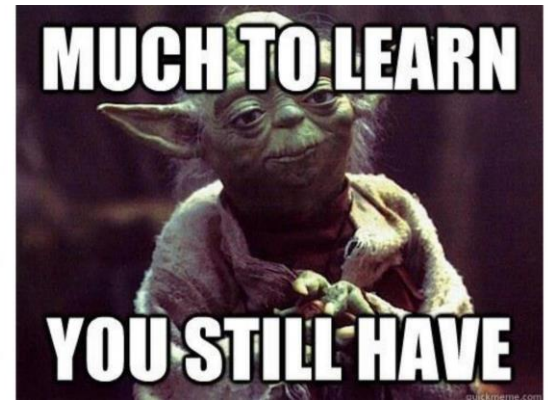
About Me

1. AI Engineer Continental
2. Graduate Student at NUS
3. Deep Learning and Smart grids Research
4. Previously, Research Engineer at Saama Technologies
5. Instructor at Udacity, Intel Edge AI for IoT Developers



What will you learn

- What is Edge Computing?
- Why is Edge Computing important?
- Edge Computing Hardware
- Edge Computing Algorithms
- Edge Computing Software
- Why learn Edge Computing?



ROADMAP OF THE STUDY GROUP!

**Session 2: Basics of
Running Neural Network
at the Edge**

**Session 4:
Hardware at the
Edge**

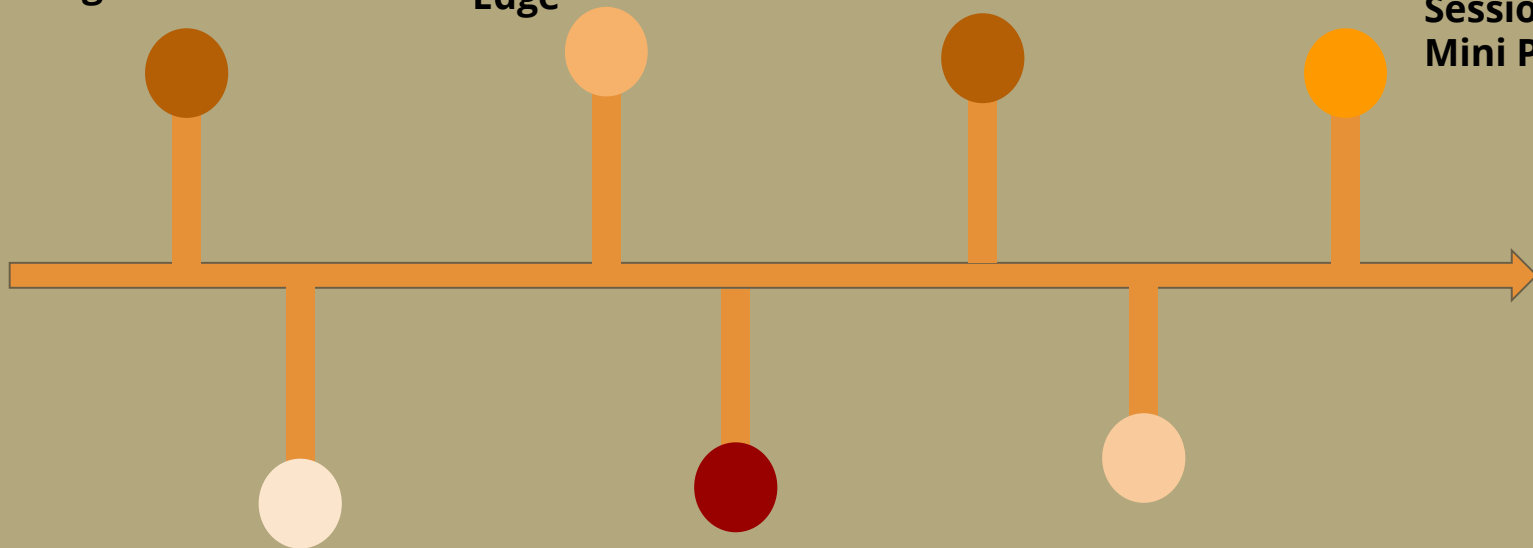
**Session 6: Research
Topic-Early Exits for
Neural Network**

**Session 8:
Mini Project**

**Session 3:
Quantization of
Neural Networks**

Session 5: Pruning

**Session 7: Edge TPU and
Edge TPU Accelerator**

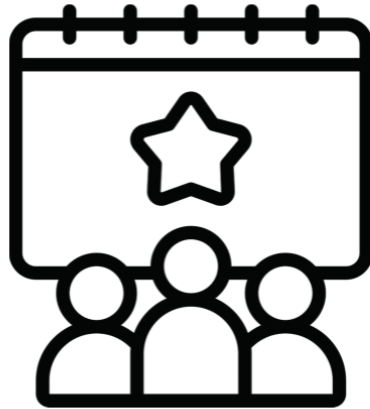


Join our slack group?

CHECK OUT THE SLACK LINK ON THE CHAT!



Research Paper



Events and Recording



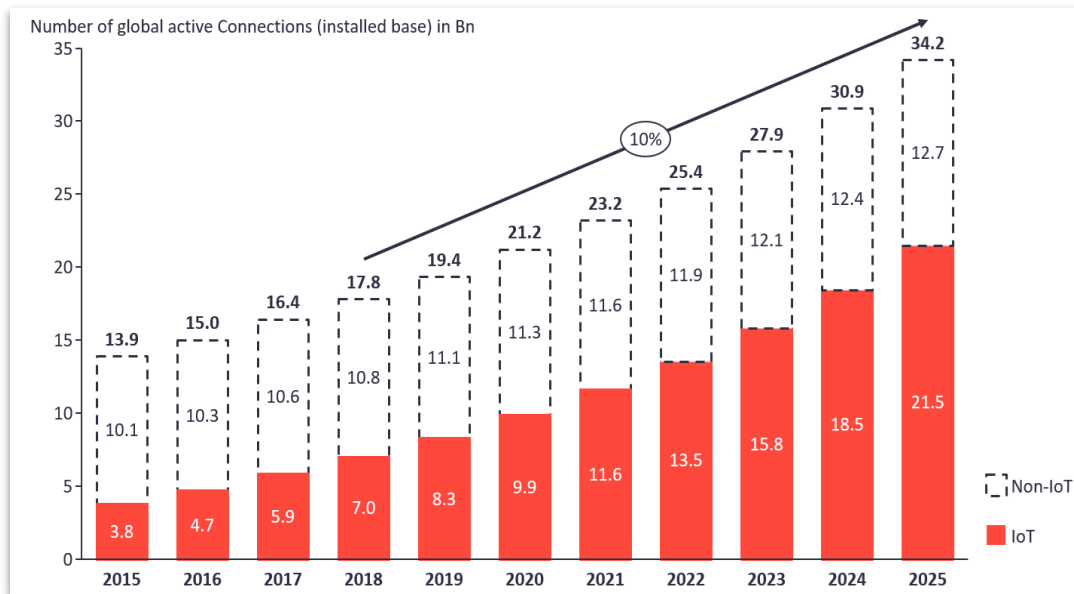
Discussions



Join our slack group?

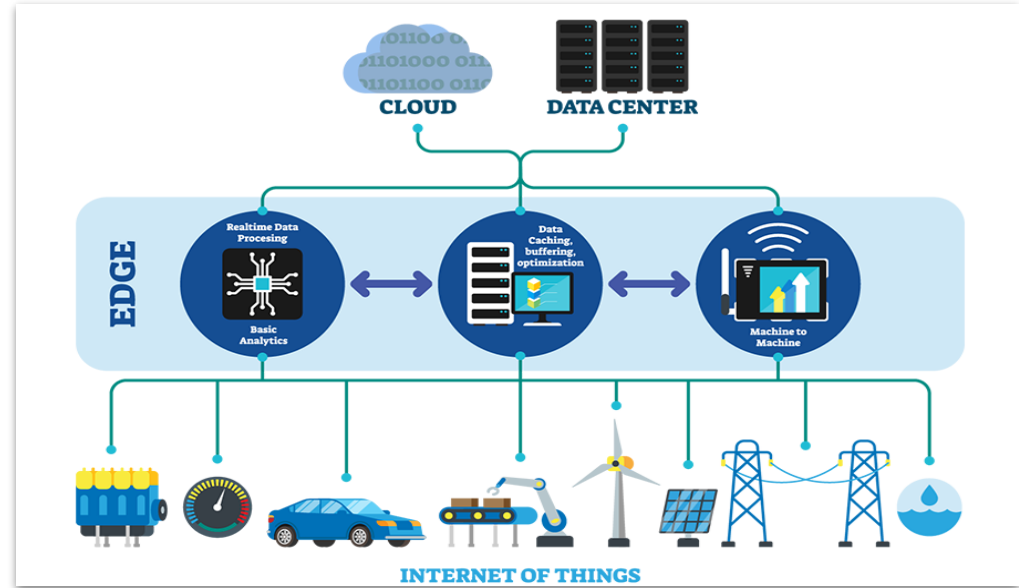
Internet of Things

- IoT devices have a microcontroller and sensors
- They can collect and transfer data, as well as perform simple tasks
- IoT devices have become popular over the last few years
- They are used as smart home and fitness devices, but are also being used in industrial settings
- However **99% of IoT sensor data is discarded**

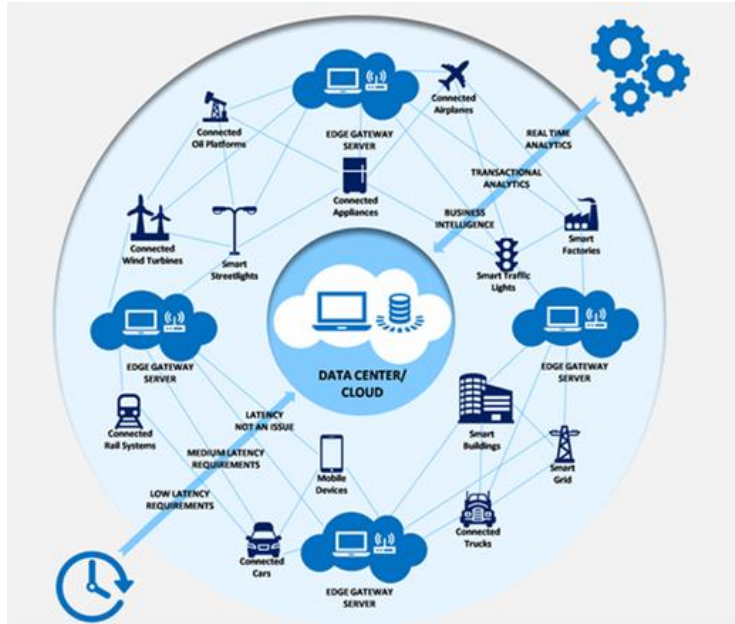


Internet of Things and Edge Computing

- Most DL models are deployed on cloud servers
- Running models closer to where the data is being generated is called edge computing



Edge Computing



1- Increase in IoT devices causes an increase in cloud dependency

2-Need edge devices which have their own data centres

3-The computing that is performed in those data centers is Edge computing

4- Application: Security Cameras, Self Driving Cars

Advantages of Edge Computing

- Reduced Latency
- Reduced Internet Bandwidth
- Increased Security
- Reduction in Dependence on Cloud Services



Edge Computing Hardware

- **Microprocessors**

- General Purpose
- More Powerful
- Consume More Power

- **Microcontrollers**

- For Simpler Tasks
- Less Computationally Powerful
- Consumes Less Power

- **Accelerators**

- Specialised for one task
- Smaller and more power efficient
- Computationally powerful, but can only perform a specific task

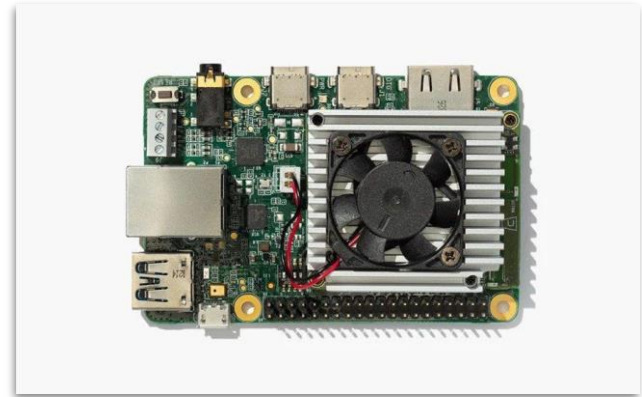


Microprocessors



Raspberry Pi 4B

- General Purpose Microprocessor
- 15W Power Consumption
- Quad core 64-bit ARM-Cortex A72 running at 1.5GHz
- 13.5 GFLOPS
- Multiple Programming Languages and Frameworks are supported



EdgeTPU Dev Board

- Raspberry Pi form-factor
- 10W Power Consumption
- Quad-core Arm Cortex-A53 and EdgeTPU Accelerator
- 4 TOPS of performance (EdgeTPU)
- TFLite Framework

Microcontrollers



Arduino Uno

- Based on the ATmega328 microcontroller
- 32KB flash memory, 2KB SRAM, 1KB EEPROM
- 16Mhz clock speed
- 14 digital I/O pins
- Compatible with the Arduino IDE
- Works with Arduino shields and hats



ESP32 - Adafruit Huzzah32

- ESP32-based microcontroller
- Small form factor
- GPIOs, BLE and Wi-Fi
- 4MB SPI Flash memory

Neural Network Accelerators



Neural Compute Stick

- Pen Drive form-factor
- 1W Power Consumption
- MyriadX Vision Processing Unit and 16 Vector Processors
- 4 TOPS of performance in total
- OpenVINO Toolkit



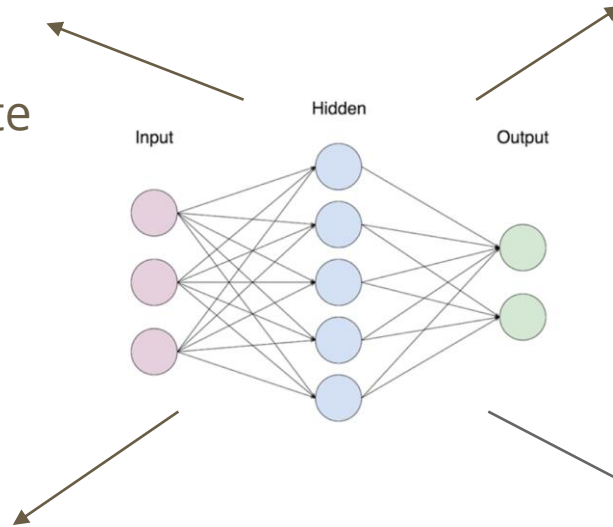
EdgeTPU Accelerator

- Pen Drive form-factor
- 5V; 100-500 mA
- EdgeTPU Accelerator
- 4 TOPS of performance (EdgeTPU)
- TFLite Framework

Need for Edge Computing Algorithms

Power constraints: Neural networks require massive amount of computational power and energy to execute on CPU and GPUs

Composed of Floating Point Values: Neural Networks are generally trained to preserve accuracy and not speed



Memory Constraints:

Laptops and PC come with at least 4GB of RAM whereas Raspberry Pi 3 has 1GB of RAM

Inference Efficiency:
We need model that takes less time for inference

Edge Computing Algorithms

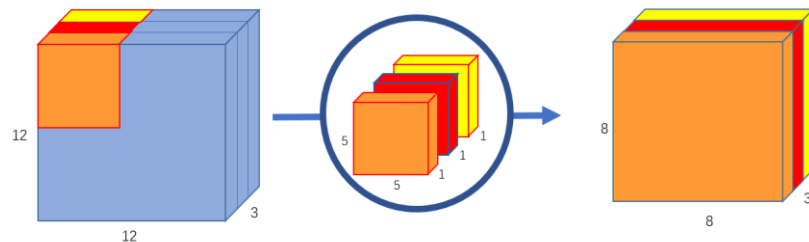
Problem: Less Compute Power, Less Memory, High Accuracy

Algorithms:

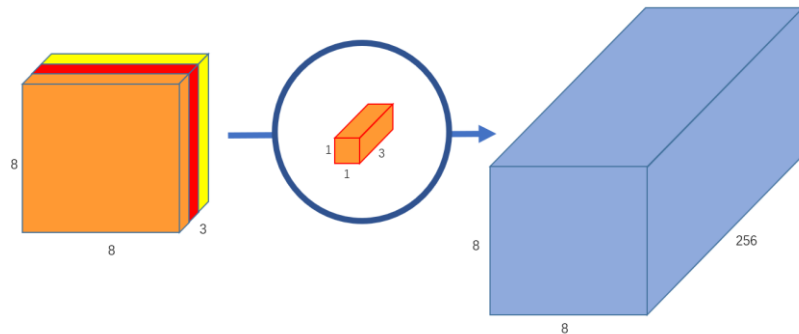
- Separable Convolutions
- Quantization
- Pruning
- Knowledge Distillation

Separable Convolutions

- Changes the standard convolution operation into two separate convolutions
- Reduces the number of weights needed to be stored
- Reduces the number of operations required to execute the layer



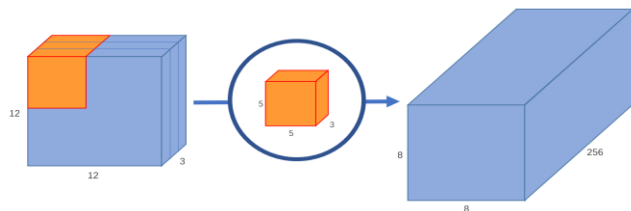
Depthwise Convolution



Pointwise Convolution

How do Separable Convolutions Reduce Computation?

Normal Convolution



$$D_k \times D_k \times C \times N \times W_o \times H_o$$

D_k is the kernel dimension

C is the number of input channels

N is the number of output channels

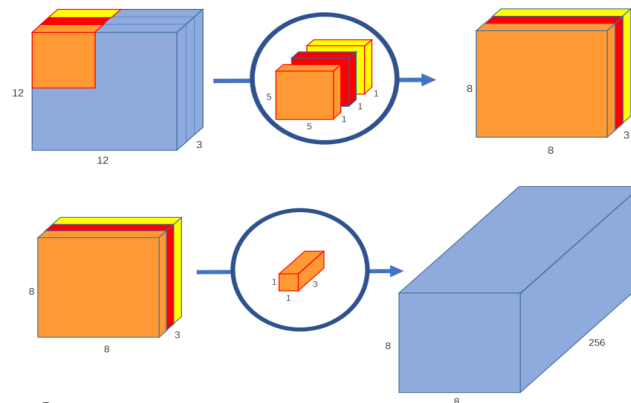
W_o, H_o are the output width and height

□ Total FLOPs:

□ $8 \times 8 \times 5 \times 5 \times 3 \times 256 = 1,228,800$

Separable Convolution

Here we separate the convolutional operation into 2 operations: Depthwise and Pointwise



Total FLOPs:

$$D_k \times D_k \times C \times W_o \times H_o + C \times N \times W_o \times H_o$$

Total

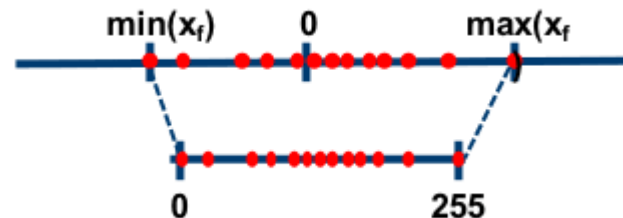
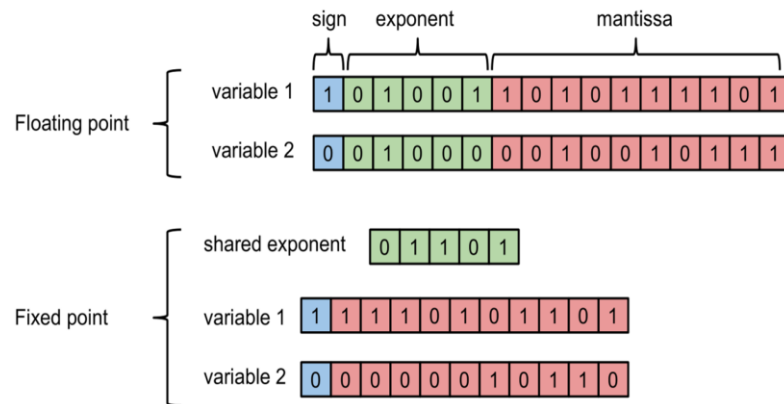
Savings:

$$\frac{1}{N} + \frac{1}{D_k^2}$$

Total FLOPs: $4800 + 49152 = 53,952$ (22x lesser)

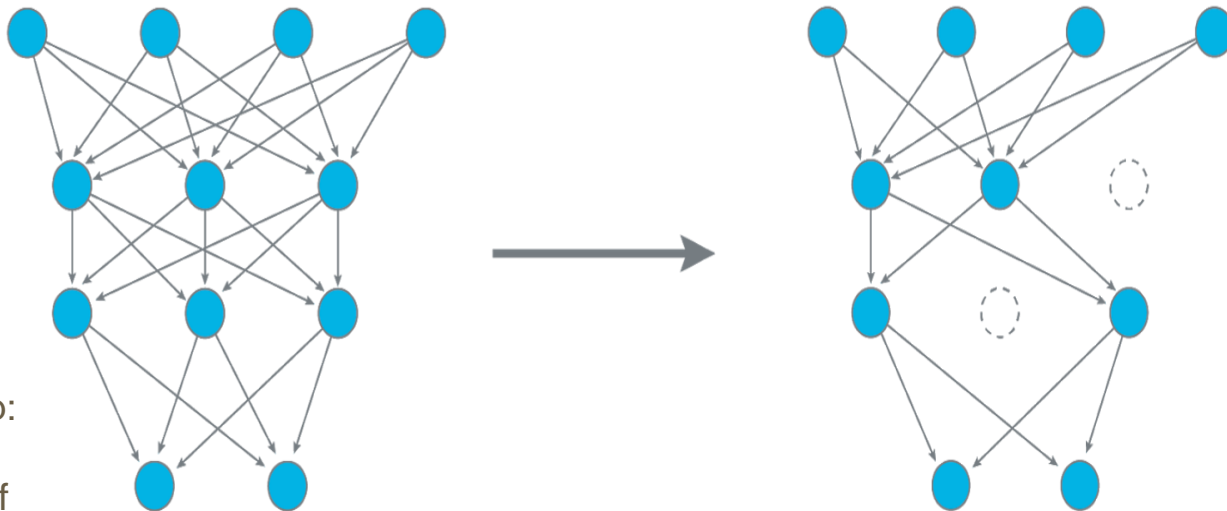
Quantization

- Neural Networks generally use 32 bit floating point weights
- These require more memory to save and custom logic to execute efficiently (which many micro-controllers do not have)
- By converting weights to INT8 (8 bits), we can save memory and compute faster
- However, it leads to a loss in accuracy since we lose precision



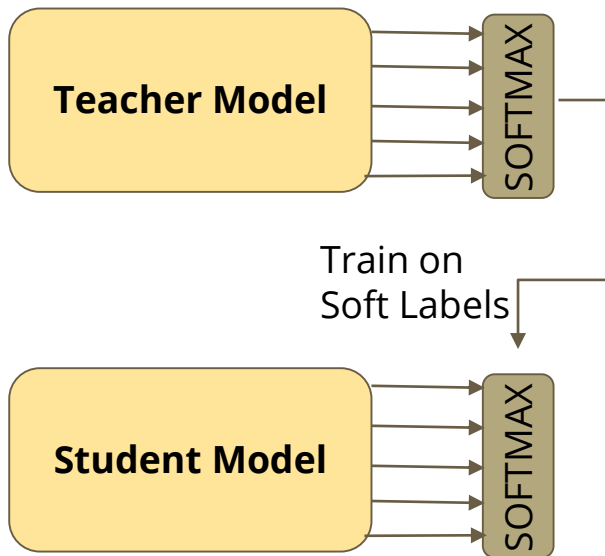
Model Pruning

- Weights can be Pruned
- Neurons can be Pruned
- Executing Pruned Networks Efficiently is Difficult
- Structured Sparsity can help:
Essentially pruning blocks of weights



Knowledge Distillation

- Works by transferring the knowledge learned by a large teacher model to a smaller student model
- The student model is easier to execute
- The student model can be trained with unlabelled data
- Student models can often achieve similar or more accuracy than the teacher



Edge Computing Frameworks

- TensorFlow Lite
 - Quantization
 - Pruning
 - Weight Clustering
 - Support for EdgeTPU
- OpenVINO
 - Quantization
 - Intermediate Representations
 - Support for NCS and other Intel Hardware
- Others
 - ONNX
 - PyTorch
 - DeepStream



Why Learn Edge Computing

- Job Prospects
 - It is an up and coming, but niche field.
 - Edge Computing concepts and optimizations can be applied in other areas
- Research
 - Really active research area with lots of potential
- Projects
 - Edge Computing is a great way to put intelligence in your projects
- Community
 - Large and open community with very helpful people!



Research Papers

Why should we add early exits to neural networks?

Simone Scardapane, Michele Scarpiniti, Enzo Baccarelli, Aurelio Uncini *

June 24, 2020

Abstract

Deep neural networks are generally designed as a stack of differentiable layers, in which a prediction is obtained only after running the full stack. Recently, some contributions have proposed techniques to endow the networks with *early exits*, allowing to obtain predictions at intermediate points of the stack. These multi-output networks have a number of advantages, including: (i) significant reductions of the inference time, (ii) reduced tendency to overfitting and vanishing gradients, and (iii) capability of being distributed over multi-tier computation platforms. In addition, they connect to the wider themes of biological plausibility and layered cognitive reasoning. In this paper, we provide a comprehensive introduction to this family of neural networks, by describing in a unified fashion the way these architectures can be designed, trained, and actually deployed in time-constrained scenarios. We also describe in-depth their application scenarios in

14.12814v2 [cs.NE] 23 Jun 2020

Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference

Benoit Jacob Skirmantas Kligys Bo Chen Menglong Zhu
Matthew Tang Andrew Howard Hartwig Adam Dmitry Kalenichenko
{benoit.jacob, skligys, bochen, menglong,
mttang, howarda, hadam, dkalenichenko}@google.com
Google Inc.

Abstract

The rising popularity of intelligent mobile devices and the daunting computational cost of deep learning-based models call for efficient and accurate on-device inference schemes. We propose a quantization scheme that allows inference to be carried out using integer-only arithmetic, which can be implemented more efficiently than floating point inference on commonly available integer-only hardware. We also co-design a training procedure to preserve end-to-end model accuracy post quantization. As a result, the proposed quantization scheme improves the tradeoff between accuracy and on-device latency. The improvements are significant even on MobileNet, a model family known for run-time efficiency, and are demonstrated in ImageNet

tizes the weights and / or activations of a CNN from 32 bit floating point into lower bit-depth representations. This methodology, embraced by approaches such as Ternary weight networks (TWN [22]), Binary Neural Networks (BNN [14]), XNOR-net [27], and more [8, 21, 26, 33, 34, 35], is the focus of our investigation. Despite their abundance, current quantization approaches are lacking in two respects when it comes to trading off latency with accuracy.

First, prior approaches have not been evaluated on a reasonable baseline architecture. The most common baseline architectures, AlexNet [20], VGG [28] and GoogleNet [29], are all over-parameterized by design in order to extract marginal accuracy improvements. Therefore, it is easy to obtain sizable compression of these architectures, reducing

5877v1 [cs.LG] 15 Dec 2017

ON THE QUANTIZATION OF RECURRENT NEURAL NETWORKS

Jian Li* and Raziel Alvarez
Google, Inc., USA.

ABSTRACT

Integer quantization of neural networks can be defined as the approximation of the high precision computation of the canonical neural network formulation, using reduced integer precision. It plays a significant role in the efficient deployment and execution of machine learning (ML) systems, reducing memory consumption and leveraging typically faster computations. In this work, we present an integer-only quantization strategy for Long Short-Term Memory (LSTM) neural network topologies, which themselves are the foundation of many production ML systems. Our quantization strategy is accurate (e.g. works well with quantization post-training), efficient and fast to execute (utilizing 8 bit integer weights and mostly 8 bit activations), and is able to target a variety of hardware (by leveraging instructions sets available in common CPU architectures, as well as available neural accelerators).

Keywords integer quantization · RNN/LSTM · hardware accelerator

1 Introduction

One important area in the development of machine learning (ML) systems is concerned with the efficient utilization

5453v1 [cs.LG] 14 Jan 2021



Copyright Notice

This presentation in this publication was presented as a tinyML® Talks webcast. The content reflects the opinion of the author(s) and their respective companies. The inclusion of presentations in this publication does not constitute an endorsement by tinyML Foundation or the sponsors.

There is no copyright protection claimed by this publication. However, each presentation is the work of the authors and their respective companies and may contain copyrighted material. As such, it is strongly encouraged that any use reflect proper acknowledgement to the appropriate source. Any questions regarding the use of any materials presented should be directed to the author(s) or their companies.

tinyML is a registered trademark of the tinyML Foundation.

www.tinyML.org