# Machine Learning for Embedded Software Engineers

## Class 2: Machine Learning Architectures for Embedded Systems

April 23, 2019

Jacob Beningo

Presented by:

DesignNews

CEC CONTINUING EDUCATION CENTER

Digi-Key ELECTRONICS

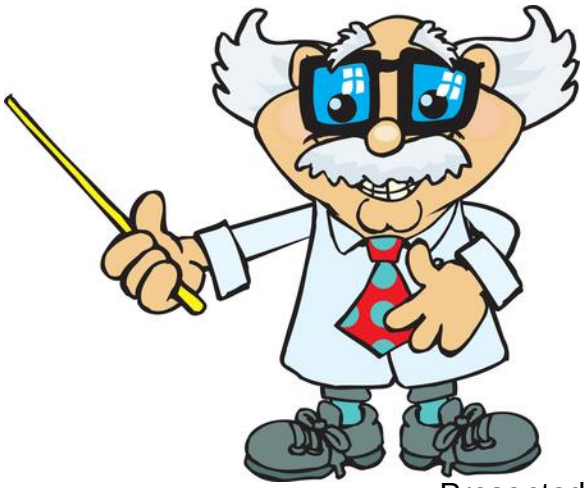
# Course Overview

**Topics:**

- Introduction to Machine Learning
- **Machine Learning Architectures for Embedded Systems**
- Machine Learning Applications: Vision and Speech
- Machine Vision with OpenMV
- Near Real-time Machine Learning using Coral

# Session Overview

- Intelligence in the Cloud
- Intelligence at the Edge
- Datasets
- Software Libraries
- CMSIS-NN



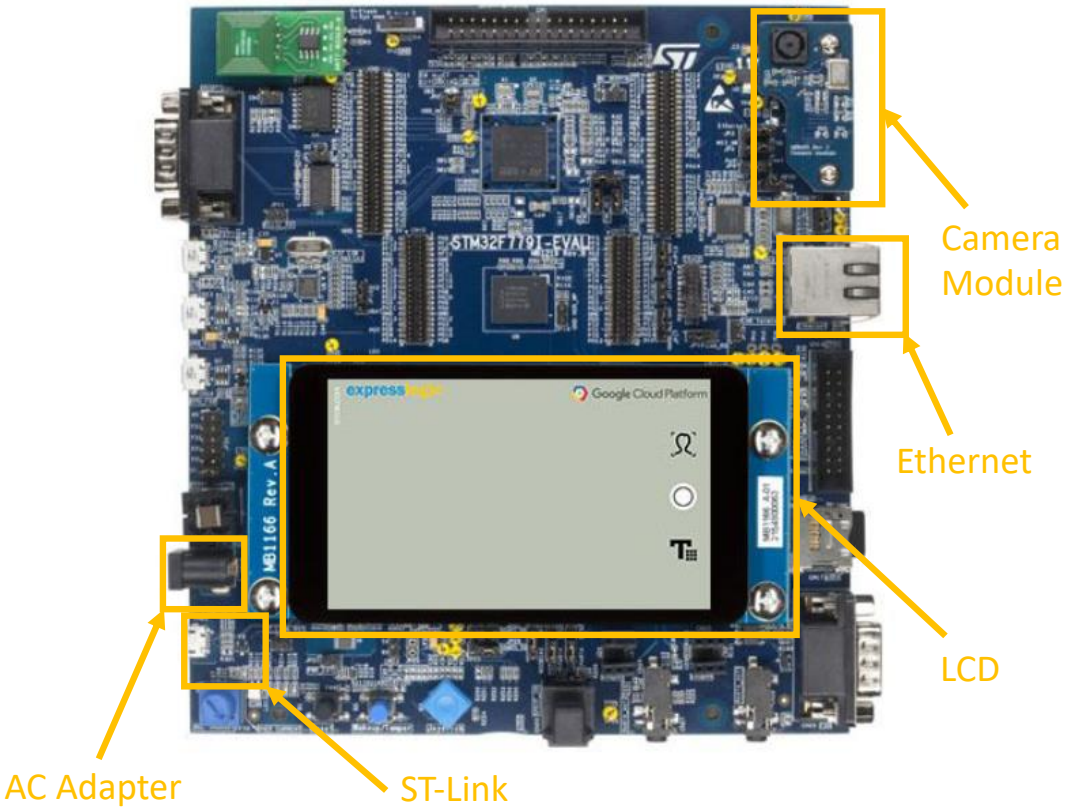
Presented by:

# Intelligent Embedded Architectures

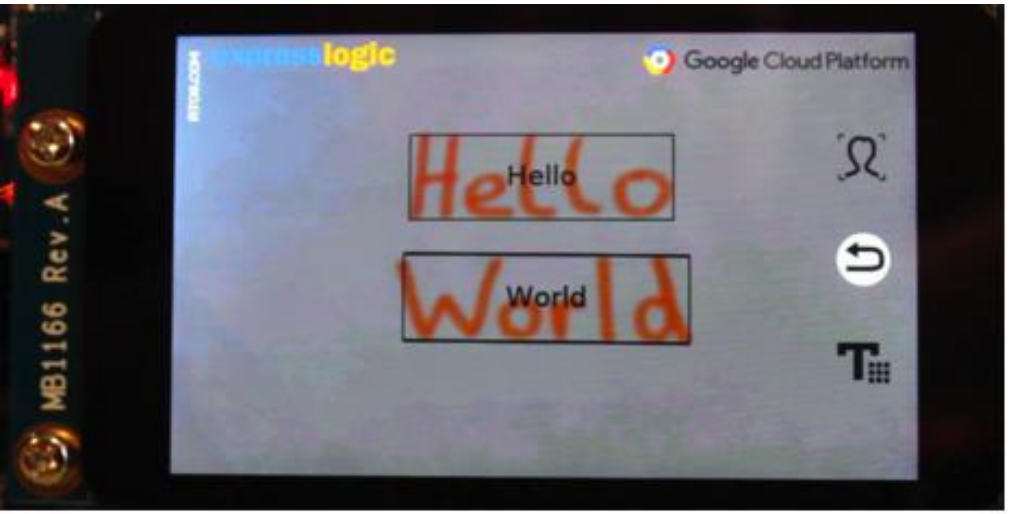
# Intelligence in the Cloud

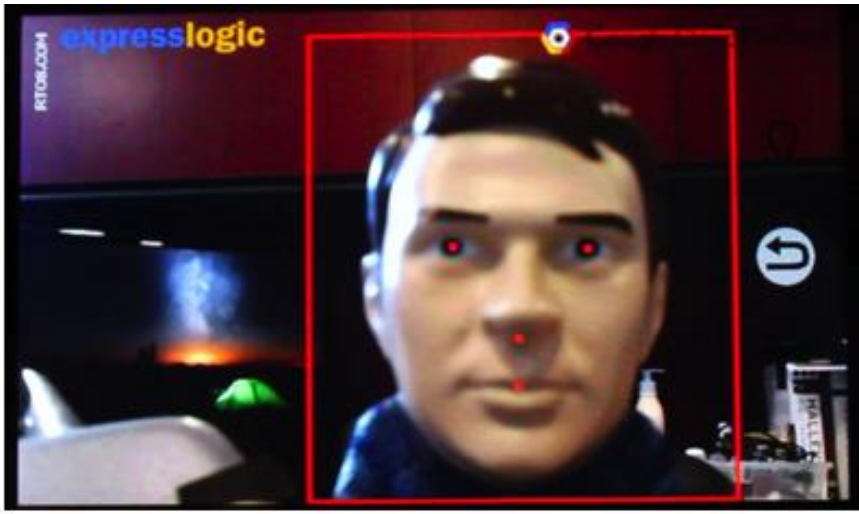
Experiment Setup

- STM32F779I-Eval
- Google Cloud Vision API's
- Express Logic
  - X-Ware IoT Platform
    - ThreadX
    - NetX HTTPS Client
    - NetX Secure TLS
    - etc



# Intelligence in the Cloud




```
Output:                                                    Log file: Off
DHCP client running...
IP address: 10.0.0.221
Network mask: 255.255.255.0
Google Vision API is ready
DNS Lookup for GOOGLE APIs Domain:vision.googleapis.com
GOOGLE APIs Domain IPv4 address: 172.217.0.10
TCP session established.
TLS session established.
Text[(258, 111), (541, 111), (541, 203), (258, 203)]: Hello
Text[(252, 240), (558, 242), (558, 336), (252, 334)]: World
```

# Why is ML Moving to the Edge?


Bandwidth


Power


Cost

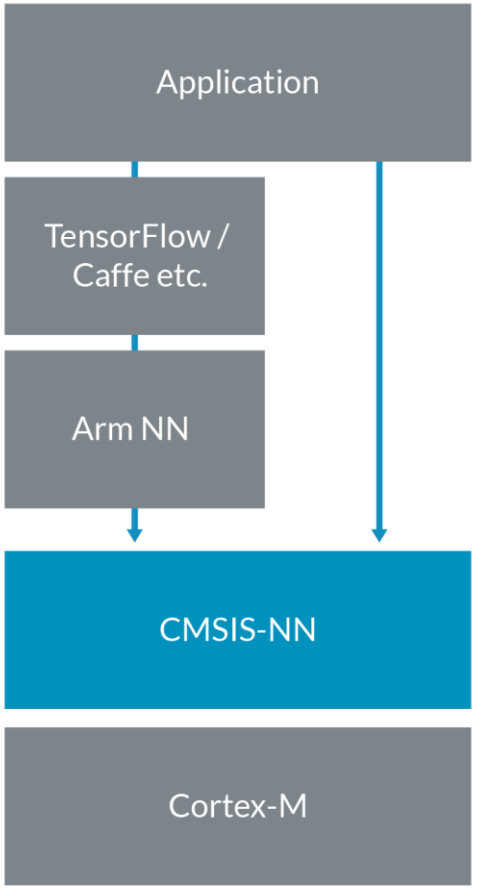

Latency


Reliability


Security

# Model Deployment on Cortex-M MCU's

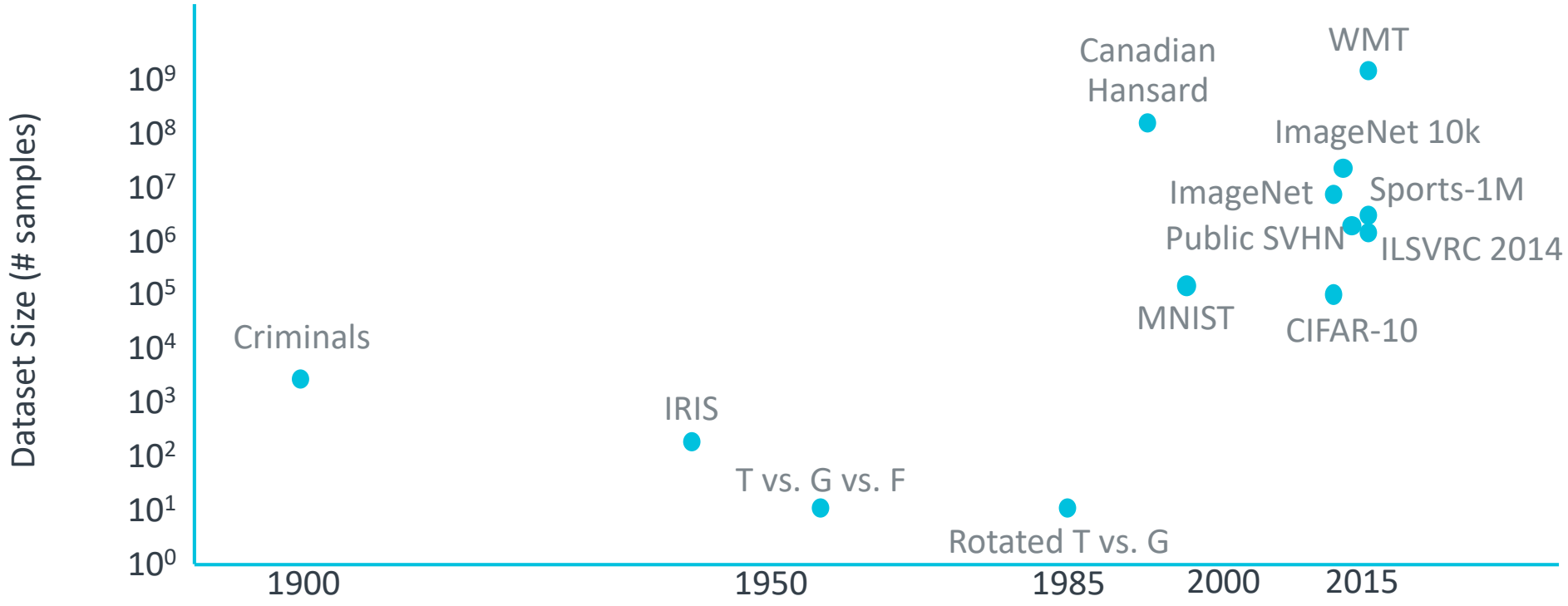
- Running ML framework on Cortex-M systems is impractical
- Need to run bare-metal code to efficiently use the limited resources
- Arm NN translates trained model to the code that runs on Cortex-M cores using CMSIS-NN functions
- **CMSIS-NN:** optimized low-level NN functions for Cortex-M CPUs
- CMSIS-NN APIs may also be directly used in the application code



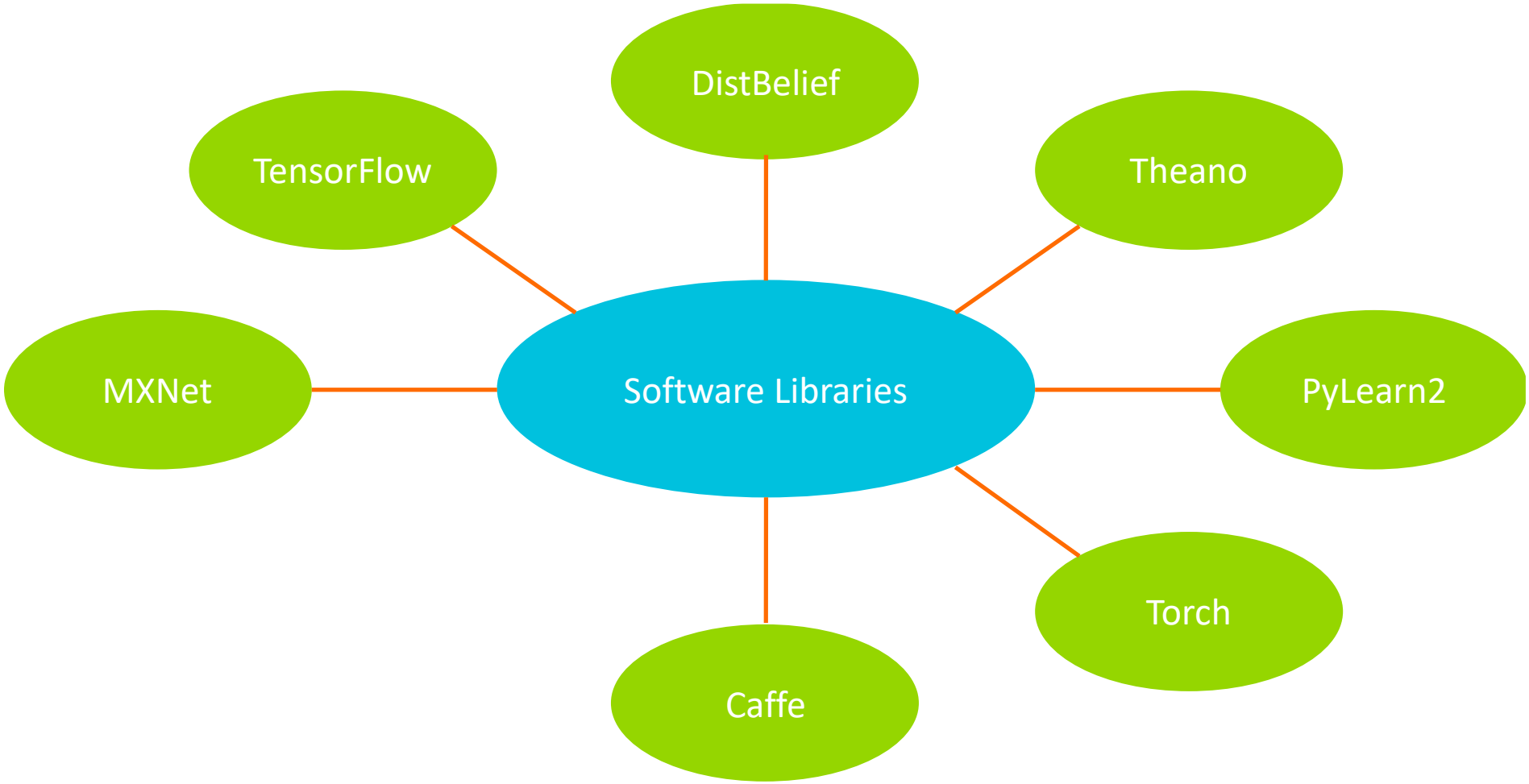
# Intelligence at the Edge

- What do you need to do machine learning at the edge?
- Hardware Floating Unit (FPU)
- ML Libraries
- Enough CPU cycles
- Training Dataset
  - 5,000 labeled examples per category for acceptable performance
  - 10,000,000 labeled examples to achieve human performance
- Time and patience

# Datasheets

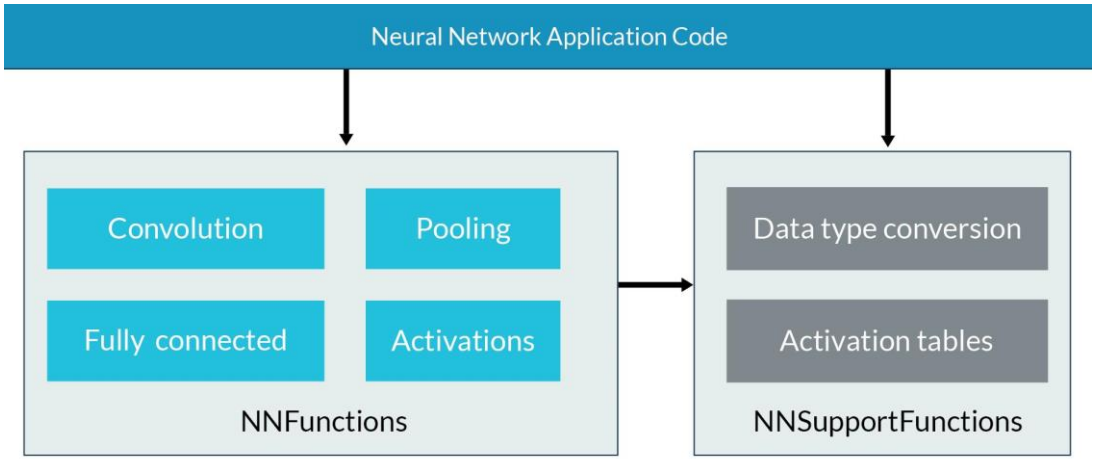

Dataset Size (# samples)
$10^9$
$10^8$
$10^7$
$10^6$
$10^5$
$10^4$
$10^3$
$10^2$
$10^1$
$10^0$
1900
1950
1985
2000
2015
Canadian Hansard
WMT
ImageNet 10k
ImageNet
Sports-1M
Public SVHN
ILSVRC 2014
MNIST
CIFAR-10
Criminals
IRIS
T vs. G vs. F
Rotated T vs. G

# Software Libraries



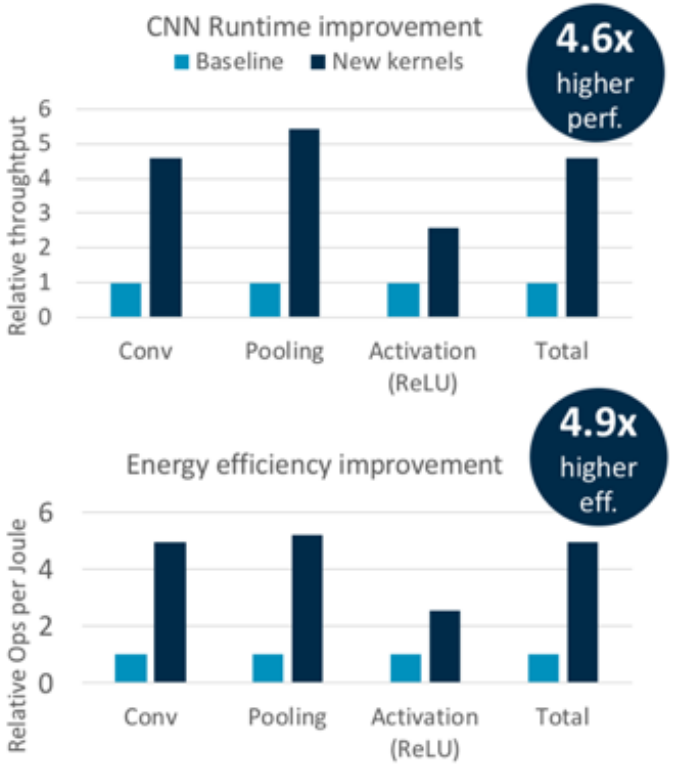
# CMSIS-NN

- CMSIS-NN: collection of optimized neural network functions for Cortex-M CPUs
- Key considerations:
  - Improve performance using SIMD instructions
  - Minimize memory footprint
  - NN-specific optimizations: data-layout and offline weight reordering



# CMSIS-NN: Efficient NN Kernels for Cortex-M CPUs

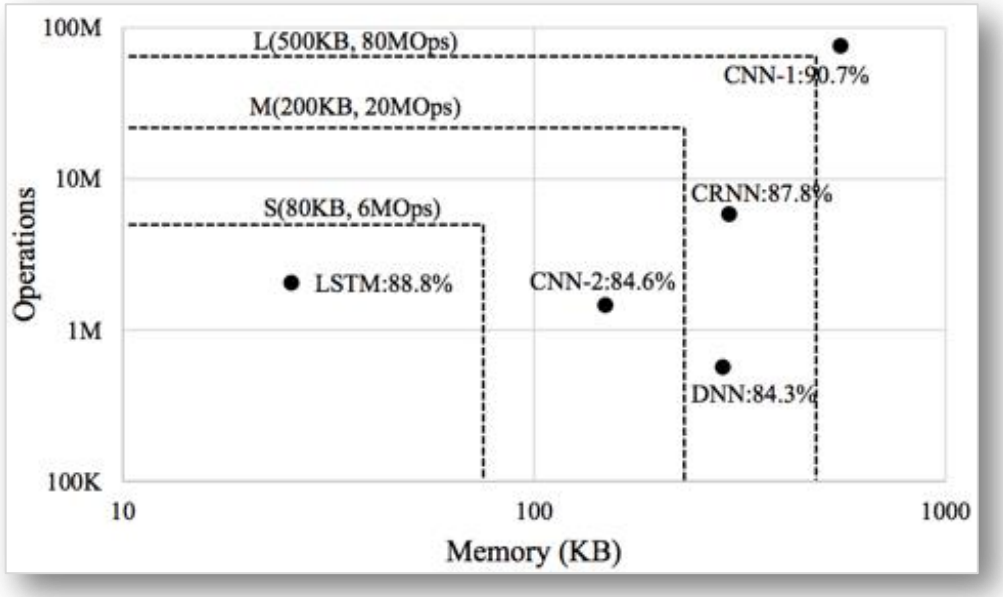
- Convolution
  - Boost compute density with GEMM based implementation
  - Reduce data movement overhead with depth-first data layout
  - Interleave data movement and compute to minimize memory footprint
- Pooling
  - Improve performance by splitting pooling into x-y directions
  - Improve memory access and footprint with in-situ updates
- Activation
  - ReLU: Improve parallelism by branch-free implementation
  - Sigmoid/Tanh: fast table-lookup instead of exponent computation




*Baseline uses CMSIS 1D Conv and Caffe-like Pooling/ReLU

CMSIS-NN paper: https://arxiv.org/abs/1801.06601

# Where are Existing NN Solutions?




NN Models from literature trained on Google speech commands dataset

Many resources available due to the openness of ML community:

- DNN: https://research.google.com/pubs/archive/42537.pdf
- CNN: https://research.google.com/pubs/archive/43969.pdf
- CNN-GRU: https://arxiv.org/abs/1703.05390
- LSTM: https://arxiv.org/abs/1705.02411

Need **compact models**: that fit within the Cortex-M system memory

Need models with **less operations**: to achieve real time performance

# Convolutional Neural Network (CNN) on Cortex-M7

- CNN with 8-bit weights and 8-bit activations
- Total memory footprint: 87 kB weights + 40 kB activations + 10 kB buffers (I/O etc.)
- Example code available in CMSIS-NN github


NUCLEO-F746ZG
216 MHz, 320 KB SRAM

| Layer | Network Parameter | Output activation | Operation count | Runtime on M7 |
|---|---|---|---|---|
| Conv1 | 5x5x3x32 (2.3 KB) | 32x32x32 (32 KB) | 4.9 M | 31.4 ms |
| Pool1 | 3x3, stride of 2 | 16x16x32 (8 KB) | 73.7 K | 1.6 ms |
| Conv2 | 5x5x32x32 (25 KB) | 16x16x32 (8 KB) | 13.1 M | 42.8 ms |
| Pool2 | 3x3, stride of 2 | 8x8x32 (2 KB) | 18.4 K | 0.4 ms |
| Conv3 | 5x5x32x64 (50 KB) | 8x8x64 (4 KB) | 6.6 M | 22.6 ms |
| Pool3 | 3x3, stride of 2 | 4x4x64 (1 KB) | 9.2 K | 0.2 ms |
| ip1 | 4x4x64x10 (10 KB) | 10 | 20 K | 0.1 ms |
| Total | 87 KB weights | Total: 55 KB<br>Max. footprint: 40 KB | 24.7 M Ops | 99.1 ms |

# Additional Resources

- Download Course Material for
  - C/C++ Doxygen Templates
  - Example source code
  - Blog
  - YouTube Videos
- Embedded Bytes Newsletter
  - http://bit.ly/1BAHYXm


From www.beningo.com under

- Blog > CEC – Machine Learning for Embedded Software Engineers