2021 embedded VISION summit®

VIRTUAL | MAY 25-28

Practical Guide to Implementing ML on Embedded Devices

Nathan Kopp
Chamberlain Group

CHAMBERLAIN GROUP

About the Chamberlain Group

CHAMBERLAIN GROUP

LiftMaster CHAMBERLAIN myQ tend

CPSG SYSTEMS Merlin GRIFCO

Chamberlain Group (CGI) is a global leader in access solutions and products.

Over 8,000 Employees Worldwide

CGI is a global team with solutions and operations designed to serve customers in a variety of markets worldwide.

VISION

Giving the Power of Access and Knowledge

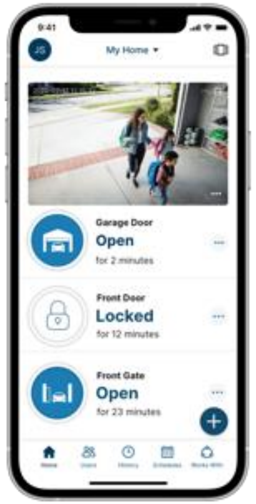
MISSION

People everywhere rely on CGI to move safely through their world, confident that what they value most is secure within reach.

END-MARKETS SERVED

Residential
Commercial
Automotive

About the Chamberlain Group



Garage Door
Open
for 2 minutes

Front Door
Locked
for 12 minutes

Front Gate
Open
for 23 minutes

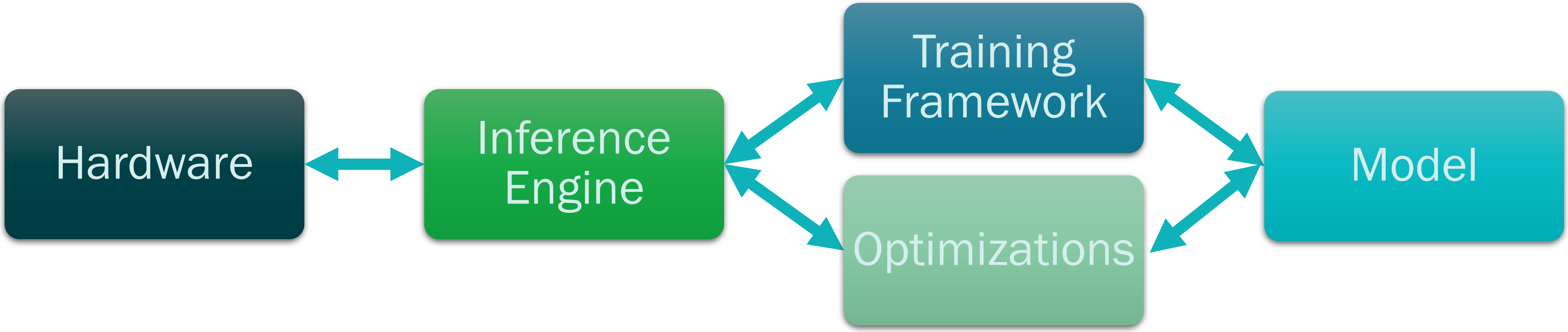
Goals of this talk

More is possible than you think!

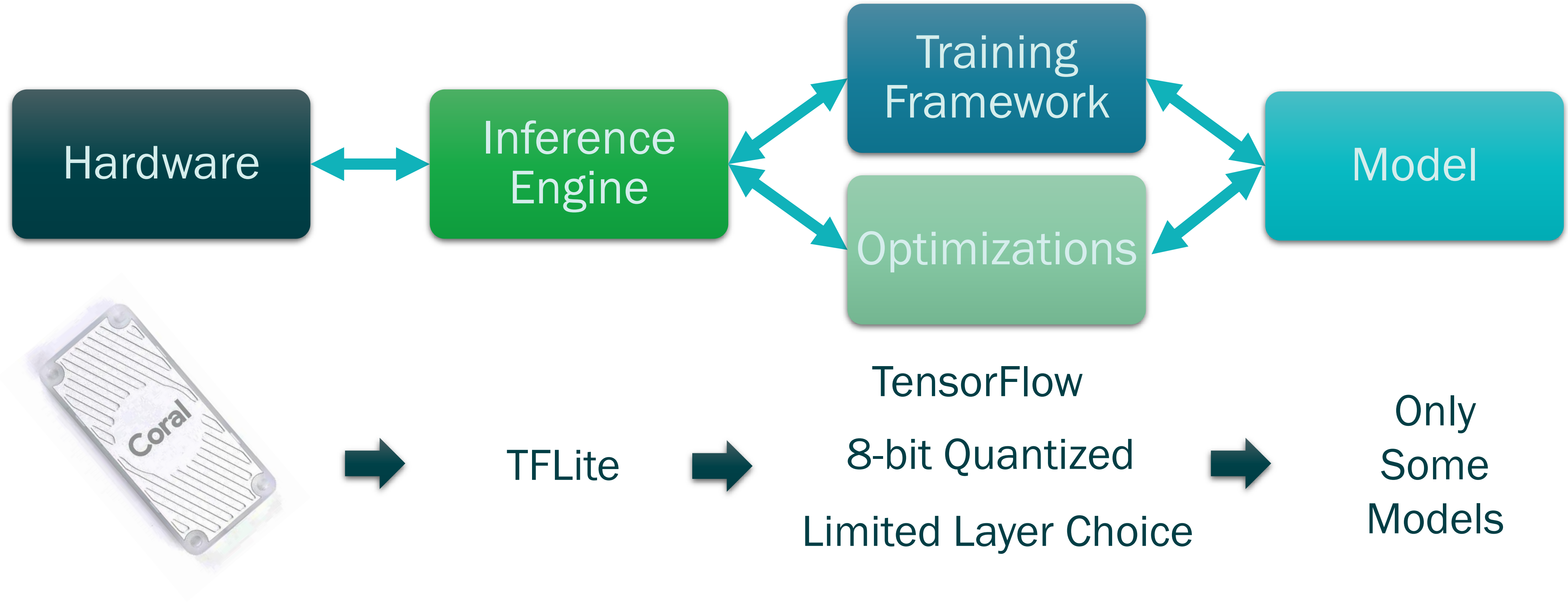
- Survey the landscape of edge inference implementation
- Explore software & hardware choices
- Examine model & optimization choices

Choices

Everything is intertwined.



Choices



Hardware

Inference Engine

Training Framework

Optimizations

Model

TFLite

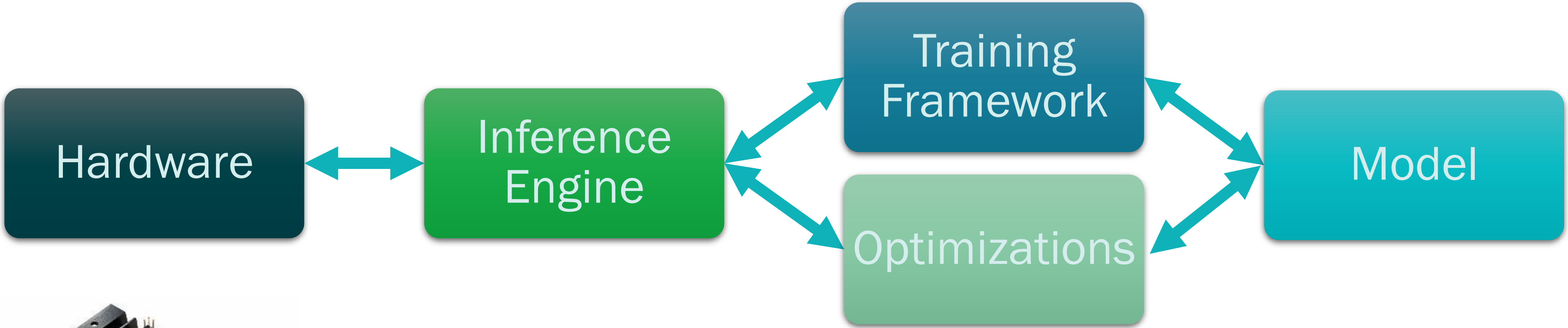
TensorFlow

8-bit Quantized

Limited Layer Choice

Only Some Models

Choices


Hardware

Inference Engine

Training Framework

Optimizations

Model

→ TensorRT

← PyTorch

→ 16-bit FP

← Complex Pruning

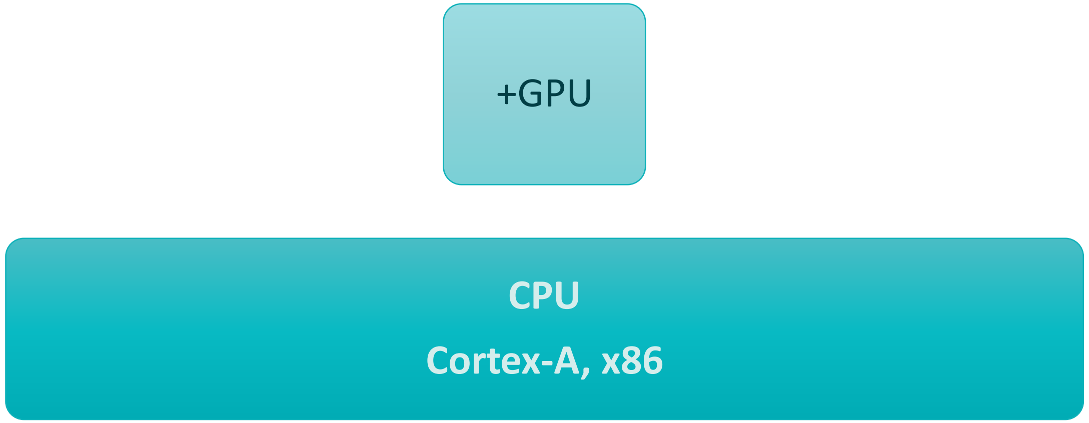
← YOLOv5

2021 embedded VISION summit VIRTUAL | MAY 25-27

Hardware

CHAMBERLAIN GROUP

Hardware Choices



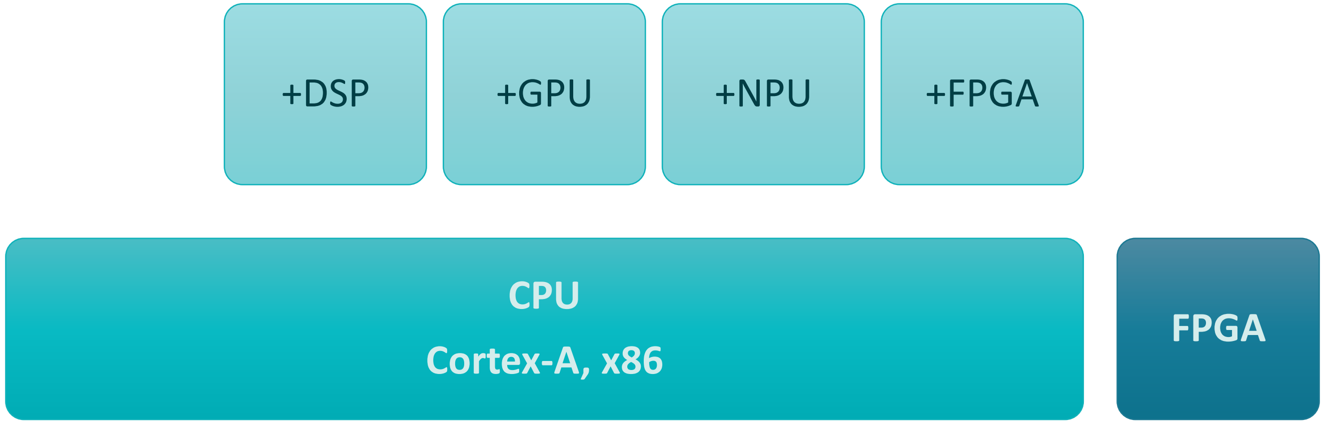
+GPU

CPU

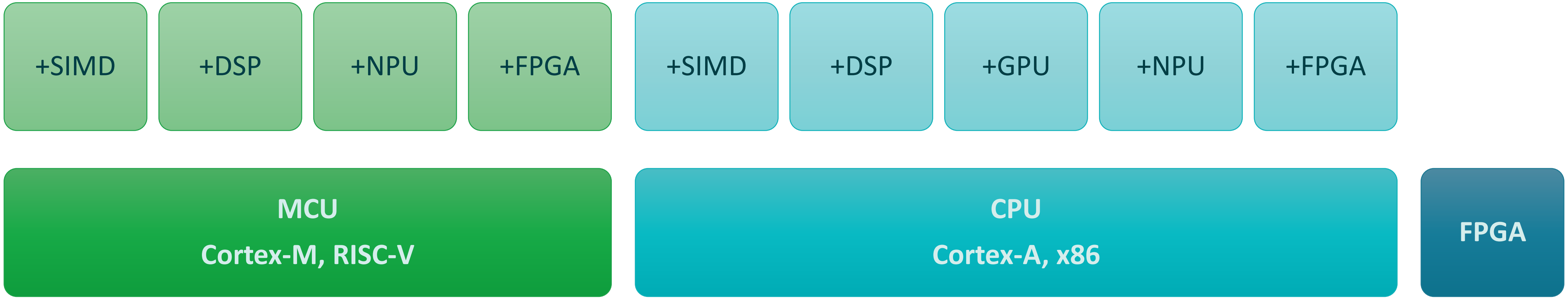
Cortex-A, x86

CHAMBERLAIN GROUP

Hardware Choices



Hardware Choices



+SIMD
+DSP
+NPU
+FPGA

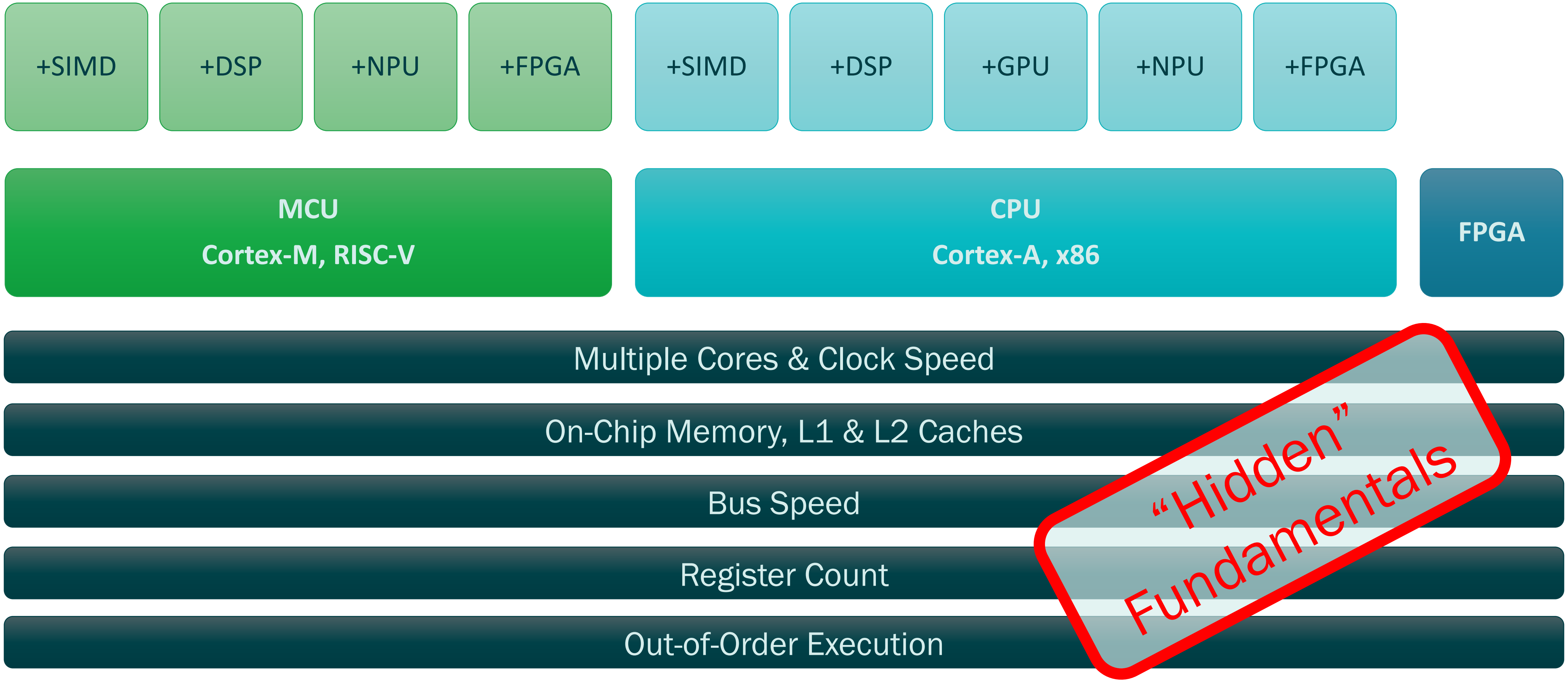
MCU
Cortex-M, RISC-V

+SIMD
+DSP
+GPU
+NPU
+FPGA

CPU
Cortex-A, x86

FPGA

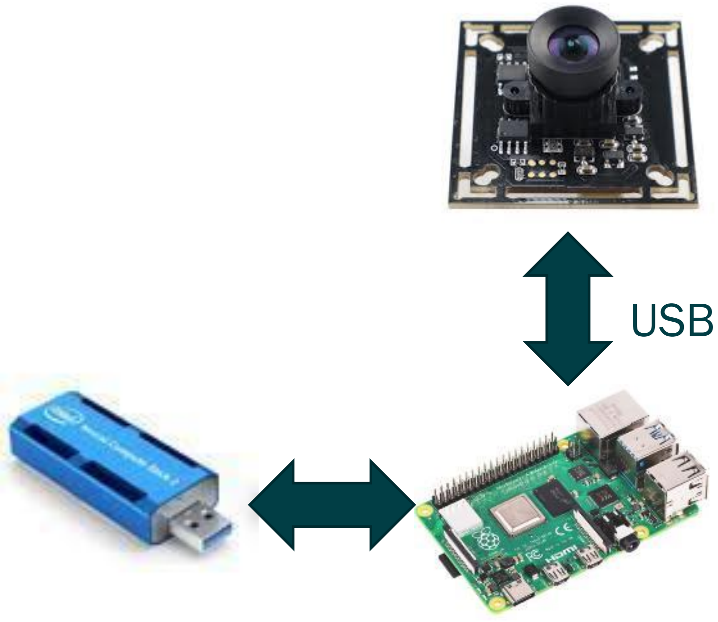
Hardware Choices



"Hidden" Fundamentals

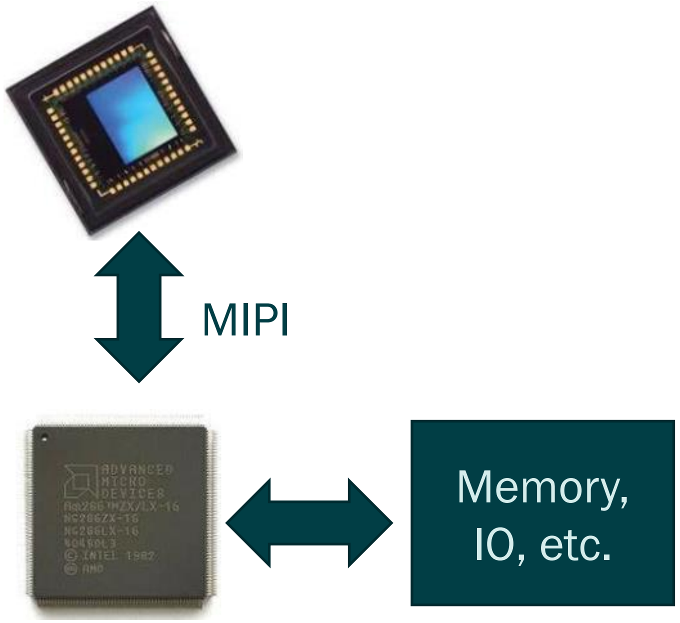
Hardware: Common Configurations

Low Volume



Software → Hardware

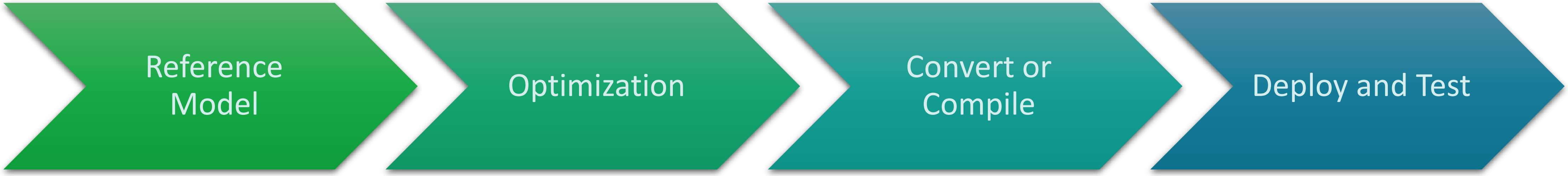
Mass Production



Hardware → Software

Model & Software

Model Optimization Workflow



Reference Model

- State-of-the-art
- FP32 GPU
- Purpose: validate your training data

Optimization

- Choose backbone
- Choose head
- Quantization
- Pruning
- NAS

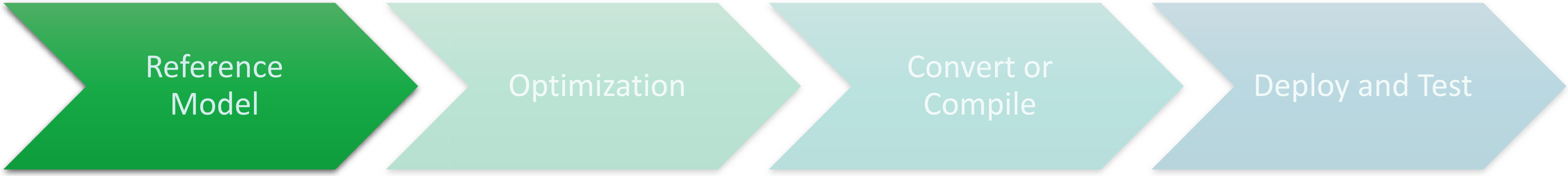
Convert or Compile

- Convert
- Compile

Deploy and Test

- Verify
- Review Choices
- Analyze
- Optimize
- Integrate

Model Optimization Workflow



Reference Model

- State-of-the-art
- FP32 GPU
- Purpose: validate your training data

Optimization

- Choose backbone
- Choose head
- Quantization
- Pruning
- NAS

Convert or Compile

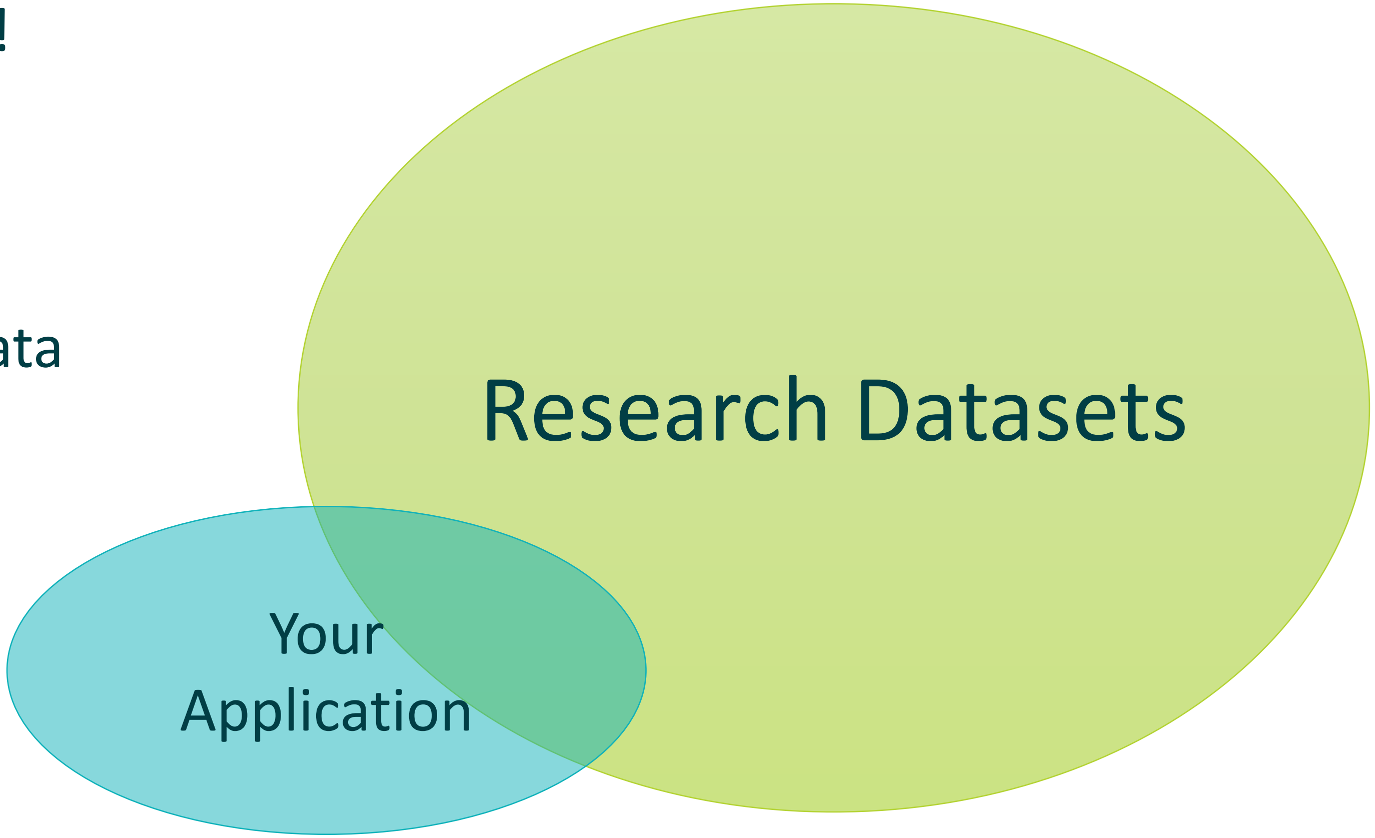
- Convert
- Compile

Deploy and Test

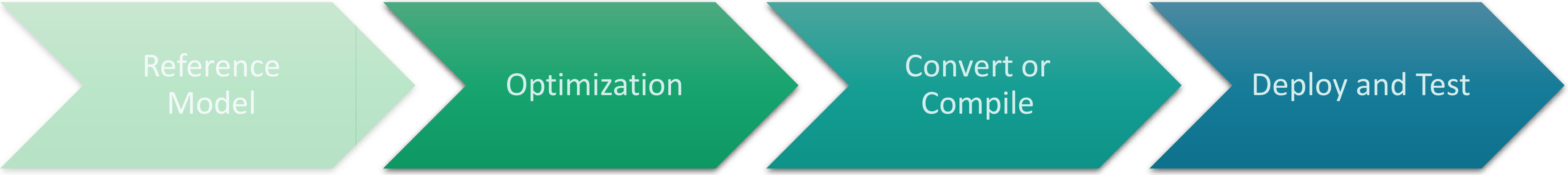
- Verify
- Review Choices
- Analyze
- Optimize
- Integrate

First, you will need data!

- Your problem space is different!
- Smaller datasets are usually OK
  - Smaller models need less data
  - Fine tuning needs less data



Model Optimization Workflow



Reference Model

- State-of-the-art
- FP32 GPU
- Purpose: validate your training data

Optimization

- Choose backbone
- Choose head
- Quantization
- Pruning
- NAS

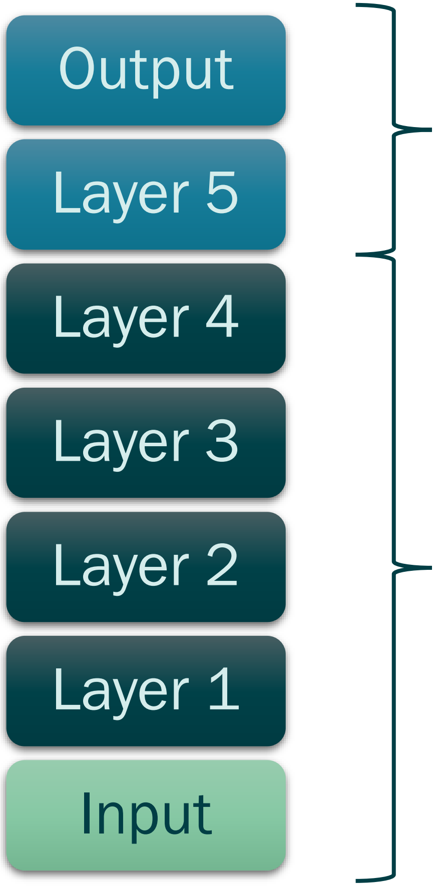
Convert or Compile

- Convert
- Compile

Deploy and Test

- Verify
- Review Choices
- Analyze
- Optimize
- Integrate

Model Mash-Up



Head & Neck:

- Interprets results
- Inexpensive to fine-tune
- Lower data requirements than backbone

Backbone:

- Extracts features
- Costly to train; needs lots of data & time
- **Recommendation: pre-trained weights**

Optimization Options

Novice → Amateur → Professional → Expert

Model Structure

- Model Zoo
- Community-Supported
- Mix & Match Head & BB
- Code it yourself

Model Optimizations

- Pre-Optimized Model
- Quantization, Pruning, Compression
- Decomposition
- NAS / OFA

Deeplite Commercial ← → Open Source

EDGE IMPULSE

Inference Engine

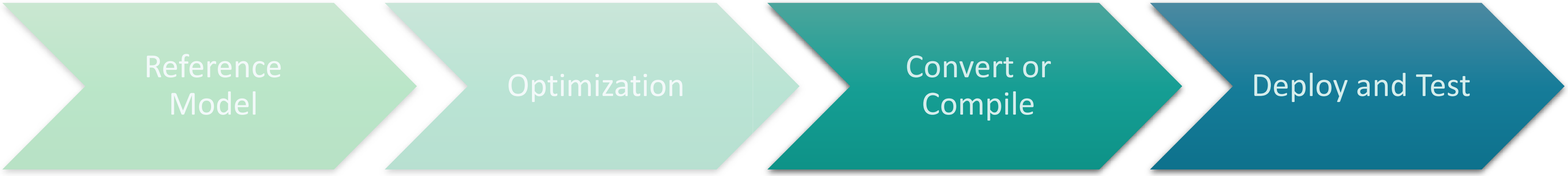
- Pre-Optimized Runtime
- Community-Supported Runtimes
- Hand-Optimized Code

Training Data

- Pretrained
- Fine-tune with your data
- Train from scratch

CHAMBERLAIN GROUP

Model Optimization Workflow



Reference Model

- State-of-the-art
- FP32 GPU
- Purpose: validate your training data

Optimization

- Choose backbone
- Choose head
- Quantization
- Pruning
- NAS

Convert or Compile

- Convert
- Compile

Deploy and Test

- Verify
- Review Choices
- Analyze
- Optimize
- Integrate

Inference Engine

- **Runtime**
  - Model is interpreted
  - Model deployed separately
  - Easier OTA updates
- **Compiler**
  - Model is compiled
  - Model is part of firmware
  - Weights are often constants

- **Code Optimizations**
  - Memory Usage
    - Cache-aware (e.g., tiling)
    - Efficient register usage
  - Vectorization
    - Use SIMD
    - Use DSP, NPU, GPU
  - Parallelization

Final Step: Integrate into your app

- **Consensus over time**
  - No model gets it right all the time
- **High frame rate:**
  - More samples for consensus
  - Lower per-sample accuracy
- **Low frame rate:**
  - Fewer samples for consensus
  - Higher per-sample accuracy

100 ms inference time does NOT mean 10 FPS!

Reserve CPU cycles for:

- Ingesting from the sensor/buffer
- Interpreting the output
- Network
- Other app functions
- Temperature management

Example

Example: Object Detection on ArmV7

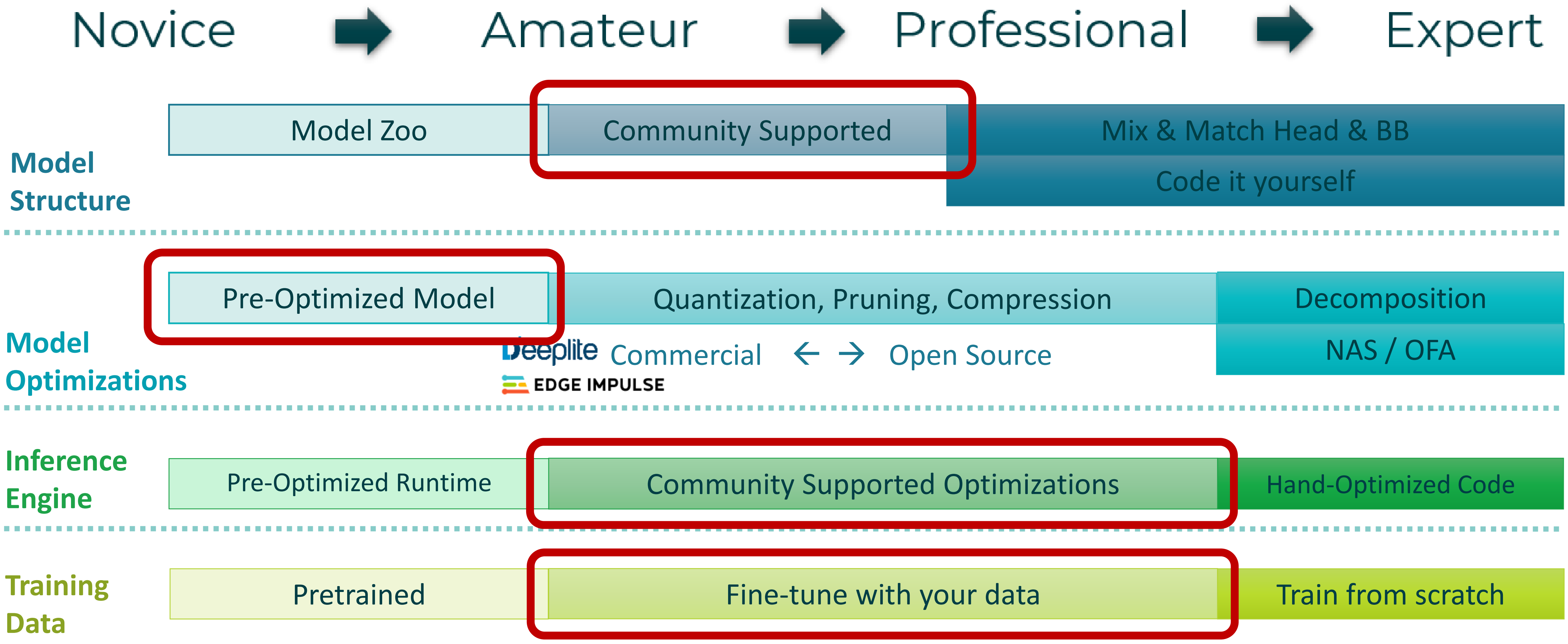
Task	Vehicle Detection
Reference Model	YOLOv5-s, FP32, PyTorch
Compute Constraints	ARMv7 w/NEON, no accelerators

Seeed NPI i.MX6ULL
Cortex A7 @ 800 MHz

Raspberry Pi 2 B v1.1
Cortex A7 @ 900 MHz

ASUS Tinkerboard (v1)
Cortex A17 @ 1.8 GHz

Optimization Options



Novice → Amateur → Professional → Expert

Model Structure

- Model Zoo
- Community Supported
- Mix & Match Head & BB
- Code it yourself

Model Optimizations

- Pre-Optimized Model
- Quantization, Pruning, Compression
- Deeplite, EDGE IMPULSE Commercial ← → Open Source
- Decomposition
- NAS / OFA

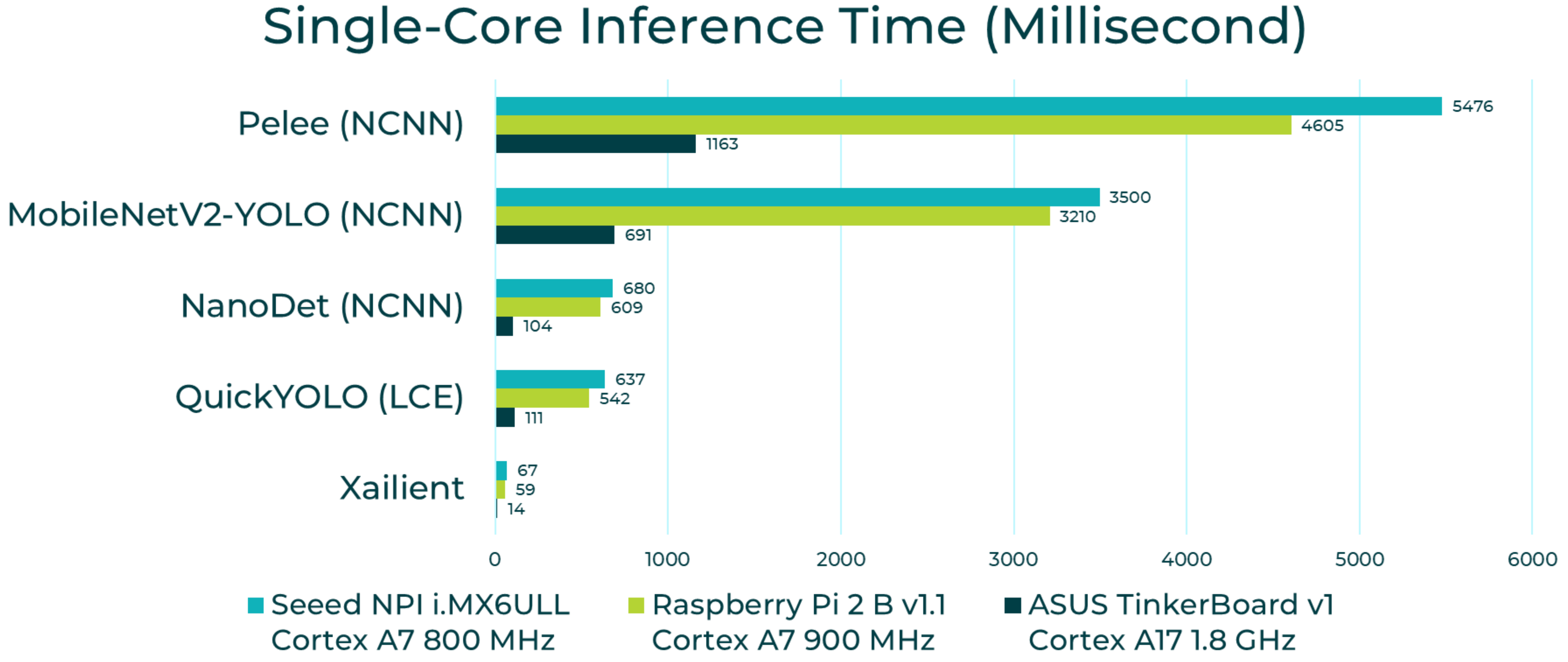
Inference Engine

- Pre-Optimized Runtime
- Community Supported Optimizations
- Hand-Optimized Code

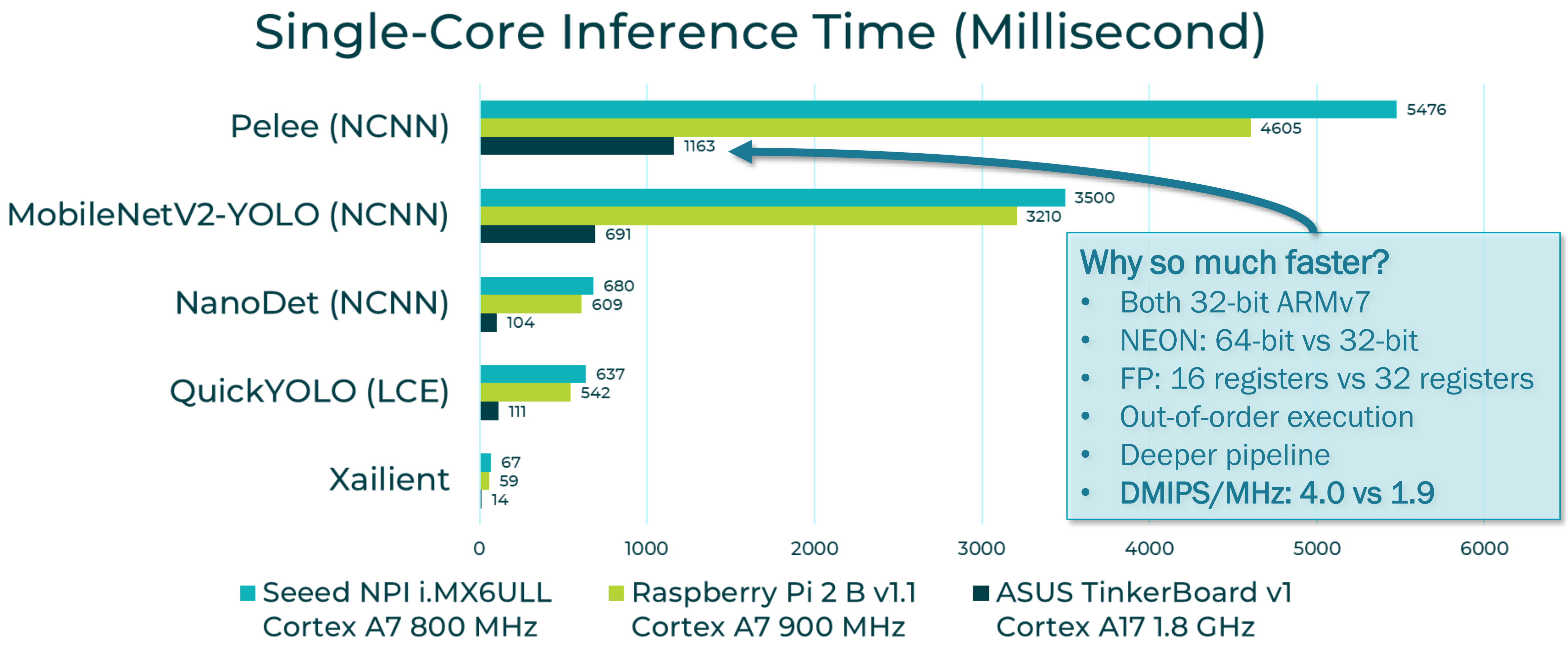
Training Data

- Pretrained
- Fine-tune with your data
- Train from scratch

Models & Tools Tested



Models & Tools Tested

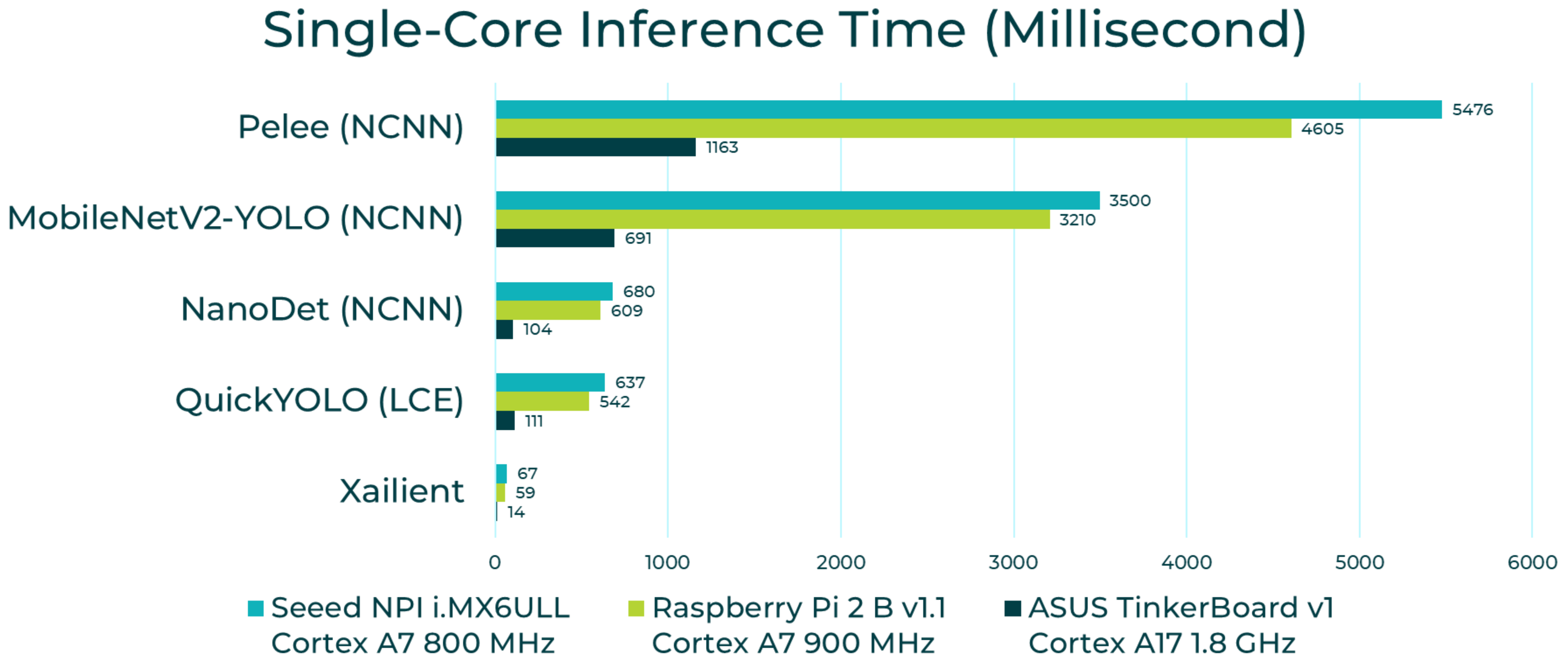


Single-Core Inference Time (Millisecond)

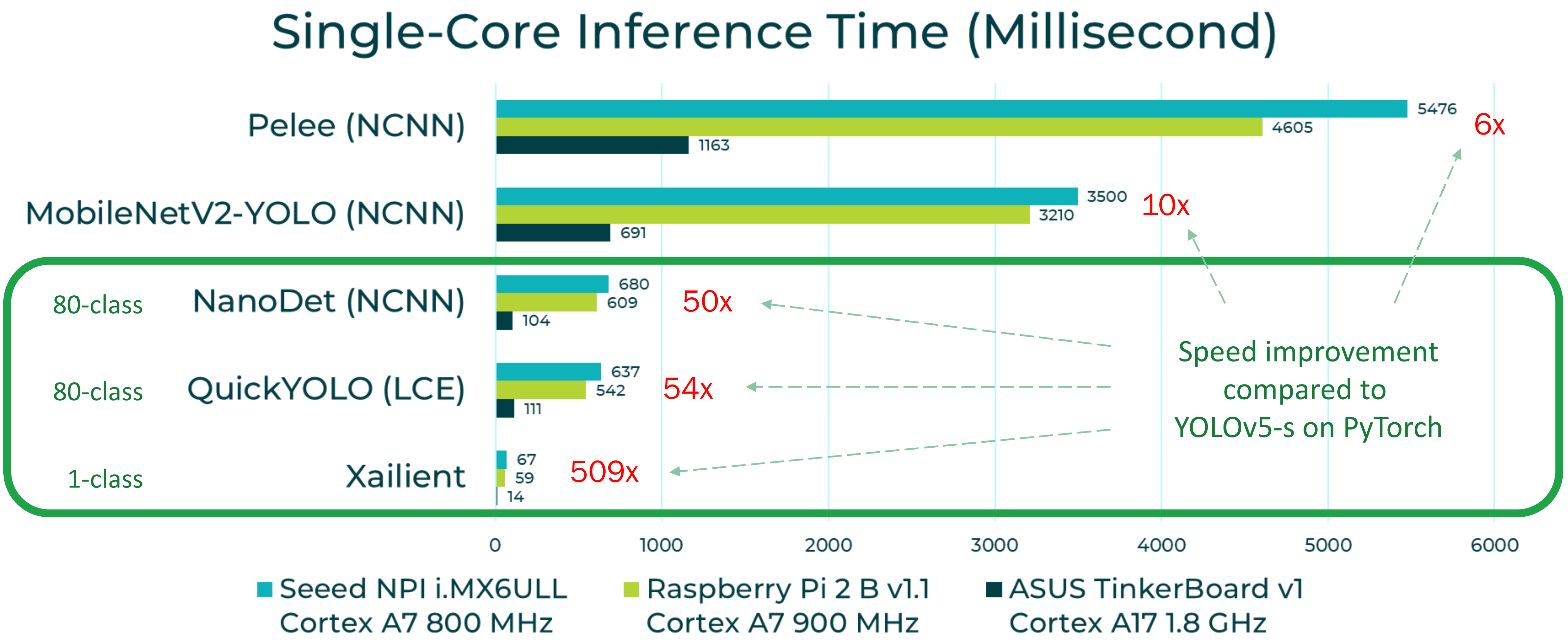
Why so much faster?

- Both 32-bit ARMv7
- NEON: 64-bit vs 32-bit
- FP: 16 registers vs 32 registers
- Out-of-order execution
- Deeper pipeline
- **DMIPS/MHz: 4.0 vs 1.9**

Models & Tools Tested



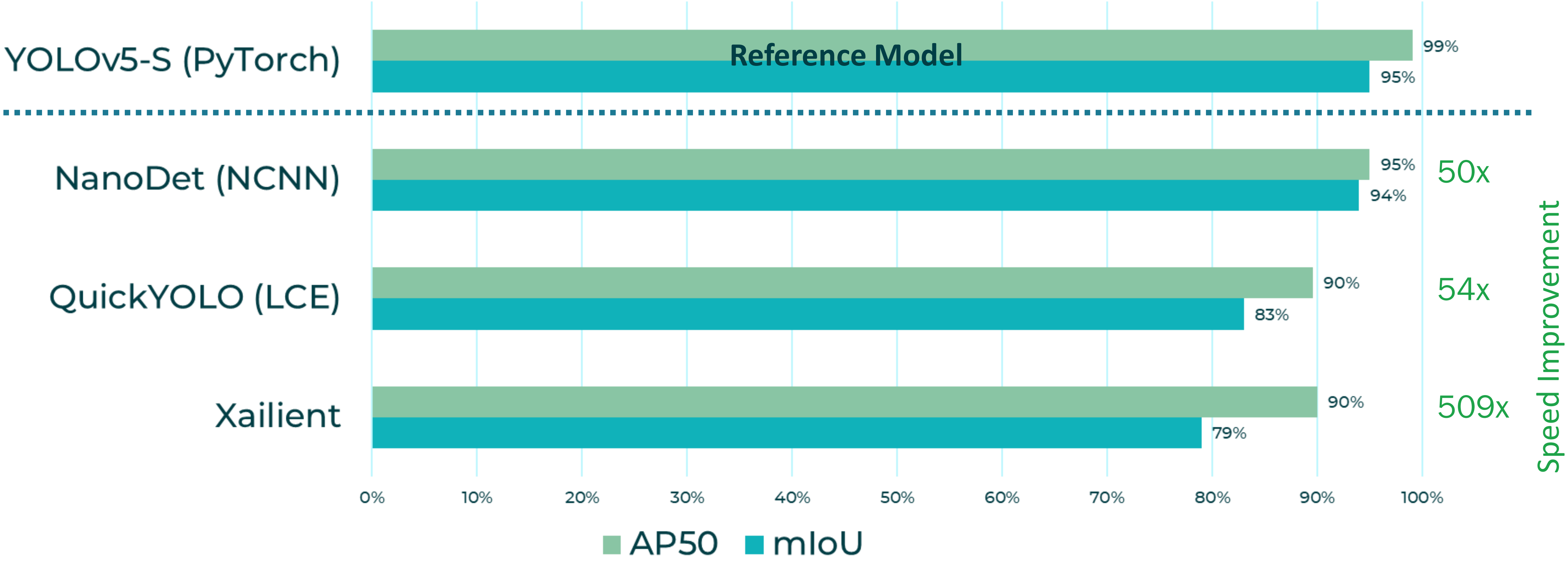
Models & Tools Tested



CHAMBERLAIN GROUP

Models & Tools Tested

Model Accuracy (Custom Vehicle Dataset)



Conclusions

- Inference on edge devices has become both possible and practical
- Small hardware features can make a big difference in speed
- Selecting the right model and the right inference engine for your hardware can expand the scope of what is possible

Example of Resource Slide

Detectors used in the Example:

YOLOv5

https://github.com/ultralytics/yolov5

NanoDet

https://github.com/RangiLyu/nanodet

QuickYOLO

https://github.com/tehtea/QuickYOLO

Xailient

https://www.xailient.com/

Chamberlain Group:

https://chamberlaingroup.com

Note:

Many more links and resources are available at the end of the slide deck.

Backup Material

Half of implementing deep learning is fighting Python & C++ errors and resolving library incompatibilities.

Pay close attention to documented versions!

Use “virtualenv”

Become a CMAKE expert!

Papers

Paper Title	URL
Larq Compute Engine: Design, Benchmark, and Deploy State-of-the-Art Binarized Neural Networks	https://arxiv.org/abs/2011.09398
Latent Weights Do Not Exist: Rethinking Binarized Neural Network Optimization	https://arxiv.org/abs/1906.02107
FCOS: Fully Convolutional One-Stage Object Detection	https://arxiv.org/abs/1904.01355
Bridging the Gap Between Anchor-based and Anchor-free Detection via Adaptive Training Sample Selection	https://arxiv.org/abs/1912.02424
Generalized Focal Loss: Learning Qualified and Distributed Bounding Boxes for Dense Object Detection	https://arxiv.org/abs/2006.04388
ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design	https://arxiv.org/abs/1807.11164

Edge Inference Engines

Generic, Open-Source

Inference Engine	Type	Notes	
TFLite	Runtime	Runtime for TF/Keras	https://www.tensorflow.org/lite
TFLite Micro	Runtime	TFLite for MCUs	https://www.tensorflow.org/lite/microcontrollers
Larq Compute Engine	Runtime	Binarized TFLite	https://github.com/larq/compute-engine
NCNN	Runtime	Tencent runtime	https://github.com/Tencent/ncnn
MNN	Runtime	Alibaba runtime	https://github.com/alibaba/MNN
Apache TVM	Compiler	Compiler and optimizer	https://tvm.apache.org/
Apache MicroTVM	Compiler	TVM for MCUs	https://tvm.apache.org/docs/microtvm/index.html
Glow	Compiler	Compiler for ONNX	https://ai.facebook.com/tools/glow/
Microsoft ELL	Compiler	Compiler	https://github.com/Microsoft/ELL
deepC	Compiler	ONNX -> LLVM	https://github.com/ai-techsystems/deepC
NNoM	Library	Keras -> C	https://github.com/majianjia/nnom

Note: This list is not comprehensive.

Edge Inference Engines

Hardware-Specific and Commercial

Vendor	Inference Engine	Type	Notes
Nvidia	TensorRT	Runtime	Support for Nvidia GPUs, such as Jetson Nano
Arm	Arm® NN	Runtime	Optimized for Arm Cortex-A CPU, Mali GPU, Ethos NPU
Arm	CMSIS-NN	Library	Library used by various runtimes and compilers
NXP	NXP eIQ™	Both	Optimized for NXP; Tflite and Glow with Arm-NN & CMSIS-NN
Qualcomm	SNPE	Runtime	For Qualcomm Snapdragon processors
Intel	OpenVINO™	Runtime	Runtime for Intel products, including Movidius
Morpho	SoftNeuro	Runtime	Commercial platform; limited details publicly available.
Edge Impulse	EON	Compiler	Commercial platform; Targeted at Microcontrollers
STMicro	STM32Cube.AI	Compiler	Optimized for STM32
Kendryte	nncase	Compiler	Kendryte K210; https://github.com/kendryte/nncase

Note: This list is not comprehensive.

Model Optimization Tools

Generic, Open-Source

Tool	Framework(s)	URL
TensorFlow MOT	TensorFlow	https://www.tensorflow.org/model_optimization
Microsoft NNI	PyTorch	https://github.com/microsoft/nni
IntelLabs Distiller	PyTorch	https://github.com/IntelLabs/distiller
Riptide	TensorFlow + TVM	https://github.com/jwfromm/Riptide
Qualcomm AIMET	PyTorch, TensorFlow	https://github.com/quic/aimet

Note: This list is not comprehensive.

Model Optimization Tools

Hardware-Specific and Commercial

Tool	Framework(s)	URL
OpenVINO NNCF	PyTorch	https://github.com/openvinotoolkit/nncf
NXP eIQ	TensorFlow, TFLite, ONNX	https://www.nxp.com/design/software/development-software/eiq-ml-development-environment:EIQ
Deeplite	PyTorch, TensorFlow, ONNX	https://www.deeplite.ai/
Edge Impulse	Keras	

Note: This list is not comprehensive.

Peripheral Accelerators

Product	Off-the-Thelf SBC	USB
Nvidia GPU	Jetson Nano, TX1, TX2	-
Movidius Myriad X	-	Intel Neural Compute Stick 2
Google Edge TPU	Coral Dev Board, Dev Board Mini	Coral USB Accelerator
Gryfalcon Lightspeeur®	-	Orange Pi AI Stick Lite
Rockchip RK1808	-	Toybrick RK1808

Note: This list is not comprehensive.

SoCs w/Embedded Accelerators

Product	Acceleration	Single Board Computer
Qualcomm Snapdragon (various)	DSP + GPU (+NPU)	*(by request only)*
Ambarella CV2, CV5, CV22S, CV25S, CV28M	DSP + NPU	*(by request only)*
NXP i.MX 8	DSP + GPU	SolidRun $160+
NXP i.MX 8M Plus	DSP + GPU + NPU	SolidRun, Wandboard $180+
Rockchip RK3399Pro	NPU	Rock Pi N10 $99+
Allwinner V831	NPU	Sipeed MAIX-II Dock $29
Sophon BM1880	NPU	Sophon Edge $129

Note: This list is not comprehensive.

MCUs for Inference

Vendor	Product	Features that support inference
Various	Cortex-M4/7/33/35P	SIMD instructions, FPU; Future Ethos-U55 microNPU
Raspberry Pi	RP2040	Memory, bus fabric
Maxim Integrated	MAX78000	Cortex-M4, CNN accelerator
Kendryte	K210	DNN accelerator
Espressif	ESP32-S3	SIMD instructions, FPU

Note: This list is not comprehensive.