tinyML® Summit

Miniature dreams can come true...

March 28-30, 2022 | San Francisco Bay Area

TINY ML

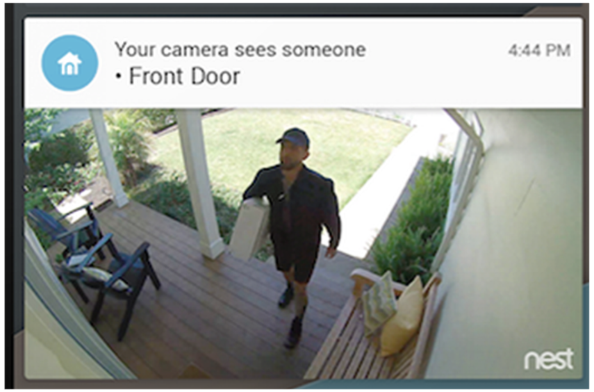
www.tinyML.org

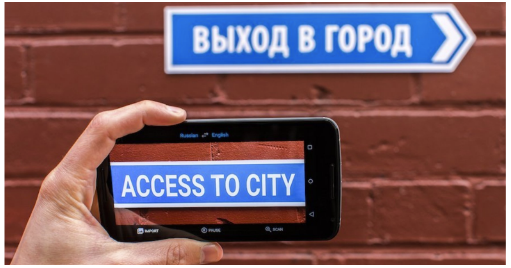
Compiling TinyML Models with microTVM

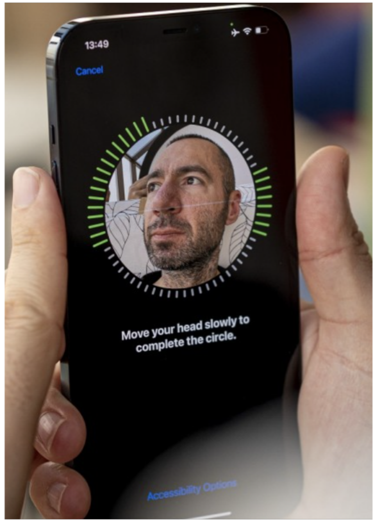
Andrew Reusch
Software Engineer, OctoML
TVM Project Management Committee Member

AI at the Edge

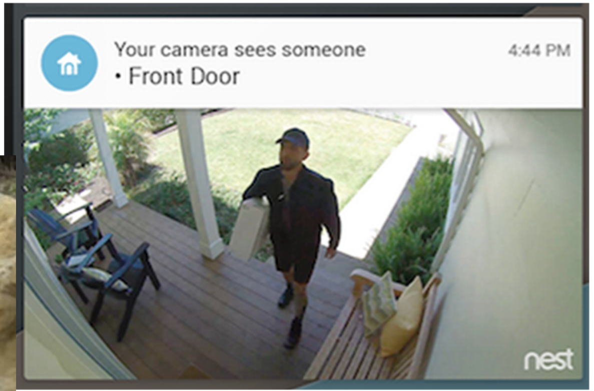
AI at the Edge

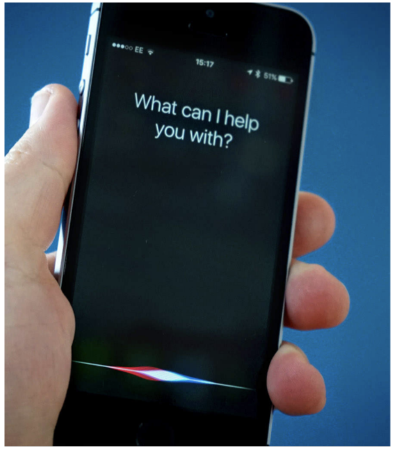


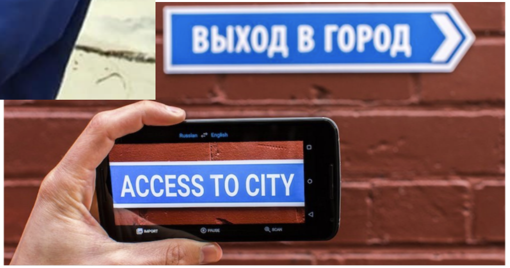


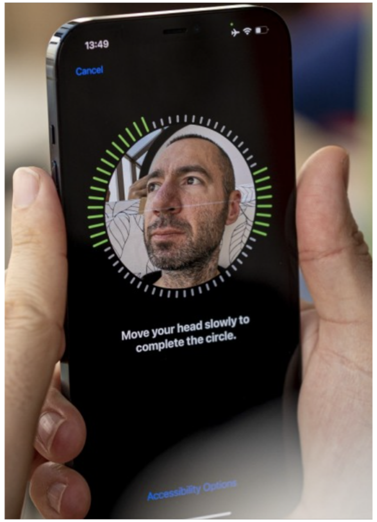


AI at the Edge


Your camera sees someone 4:44 PM
• Front Door


What can I help you with?



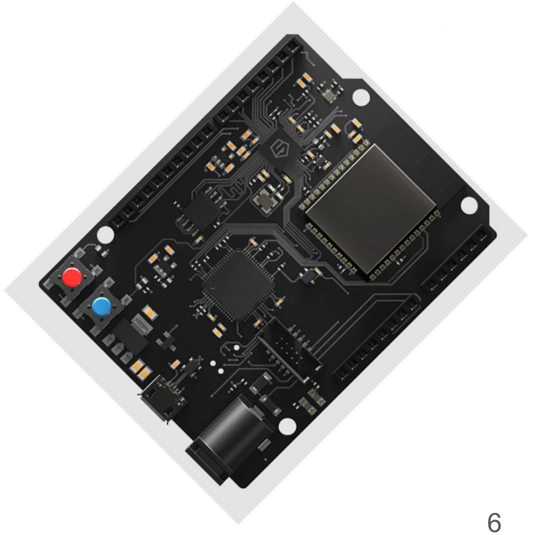

Move your head slowly to complete the circle.

The Deployment Challenge

where the model runs

This route may be easy:

✅ A fast impl may exist for each layer
✅ Tools exist to translate the model graph into C
✅ Robust test and debug tools

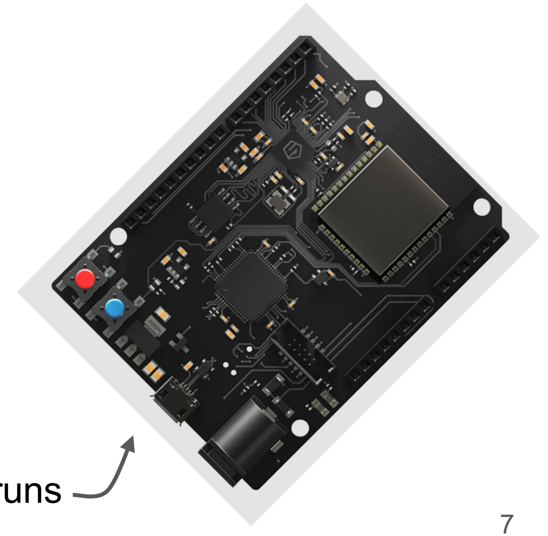


the model

The Deployment Challenge

This route may be difficult:

⛔ Hand-optimize each layer
⛔ Tune memory usage of your impl
⛔ Test and debug on your own



the model

where the model runs

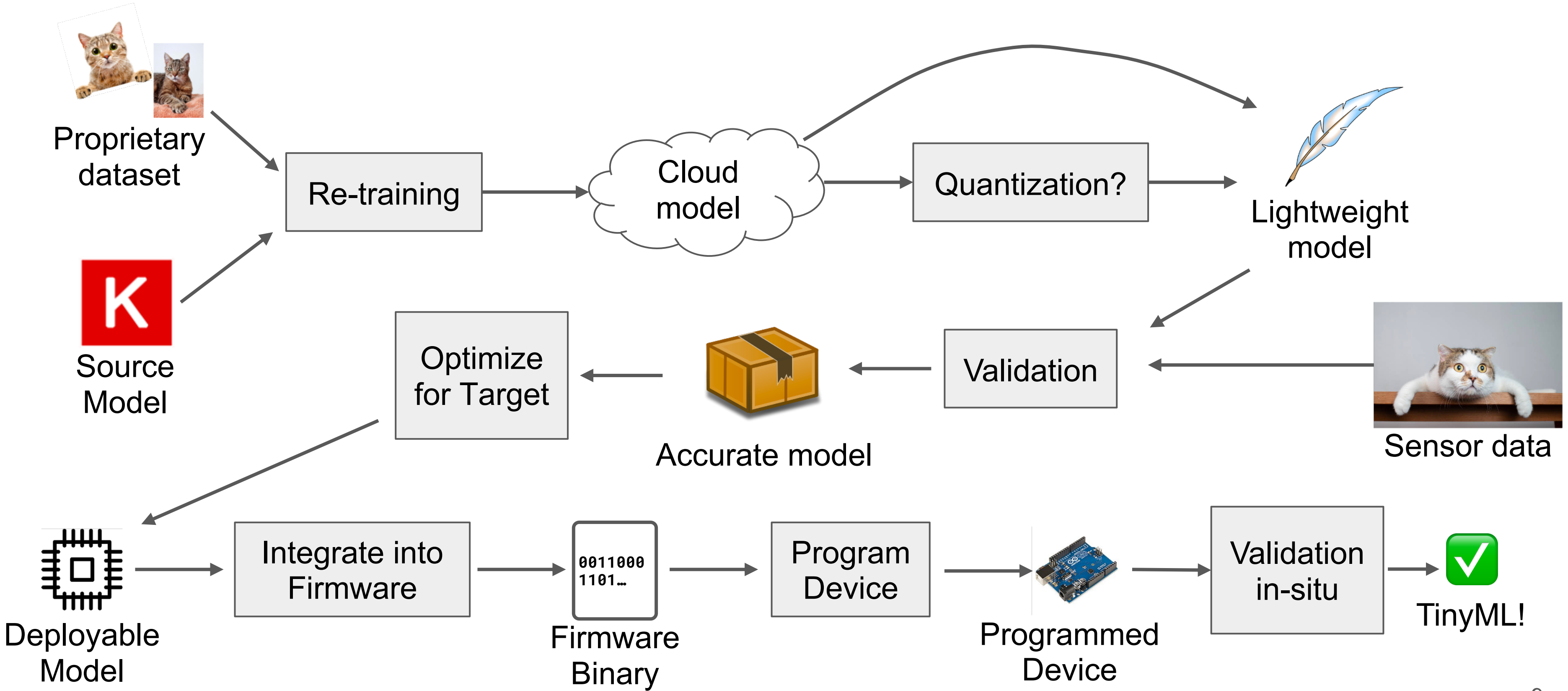
We need to speak a common language

```
x = keras.Conv2d(
  2, 3, activation='relu')
keras.MaxPool2d(x)
```

```
...
```

```
Traceback:
#0 at HardFault_Handler
#1 at ???????
#2 at conv2d_relu_maxpool_layer
#3 at run_model
#4 at main
```

Model Deployment is a Workflow

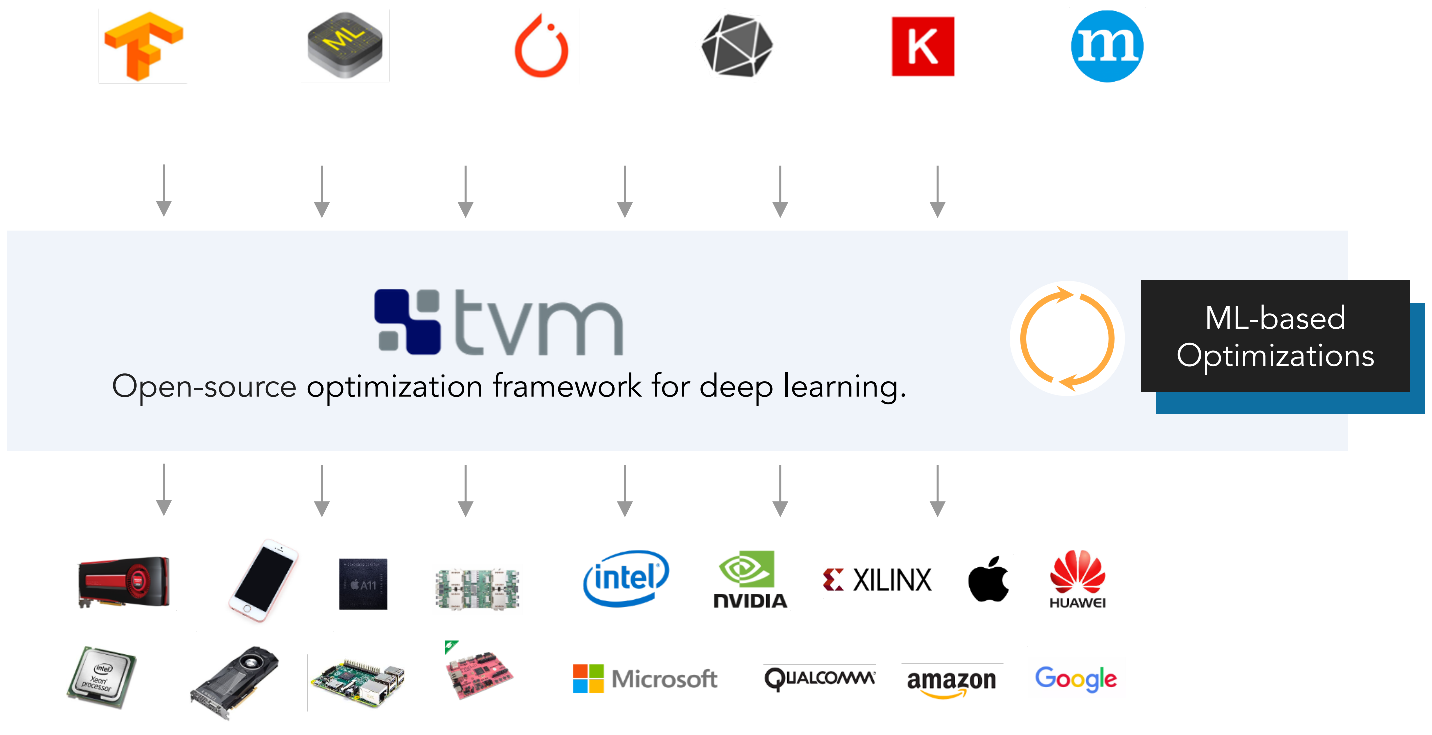


The Many Languages of TinyML Deployment

Lots of different expertise needed from different roles:

- Model developers
- MLSys engineers
- Firmware engineers
- Test engineers
- And more?

Apache TVM: Bridging the gap as a DL compiler and runtime



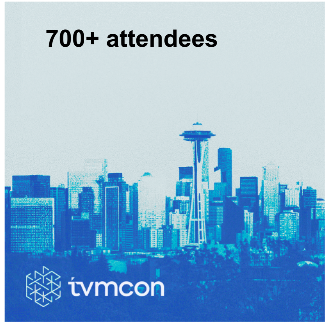
Backends for
x86, nVidia/CUDA, AMD, ARM, MIPS, RISC-V, etc

TVM is an emerging industry standard

Every "Alexa" wake-up today across all devices uses a model optimized with TVM

Open source
~674 contributors from industry and academia.

"[TVM enabled] real-time on mobile CPUs for free...We are excited about the performance TVM achieves." More than 85x speed-up for speech recognition model.

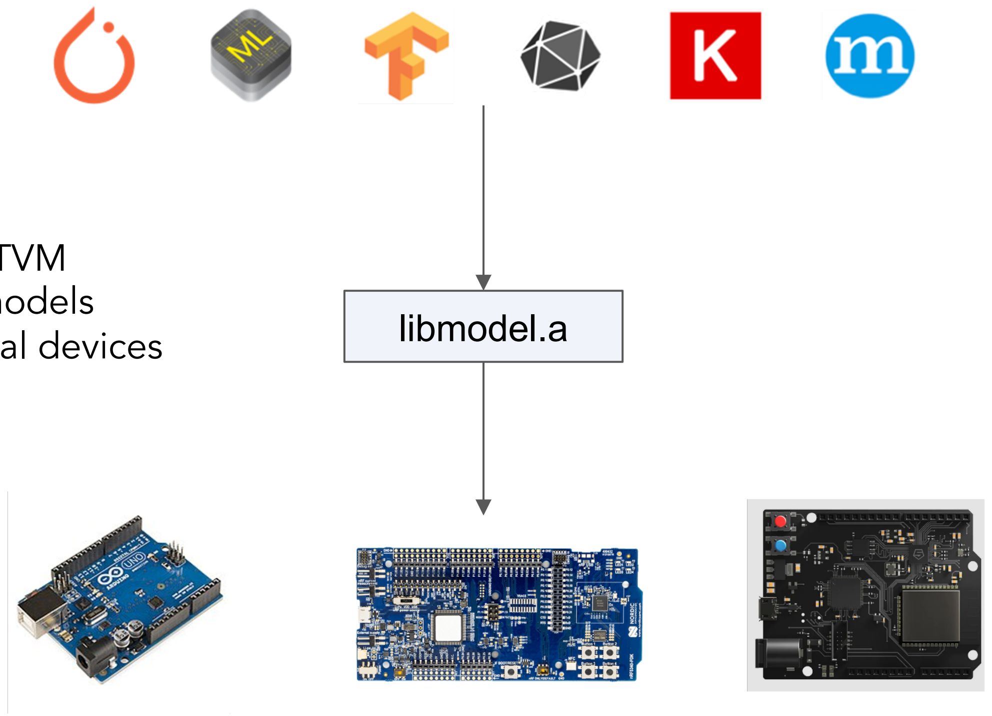


Bing query understanding: 112ms (Tensorflow) -> 34ms (TVM).
QnA bot: 73ms->28ms (CPU), 10.1ms->5.5ms (GPU)

"TVM is key to ML Access on Hexagon" - Jeff Gehlhaar, VP Technology

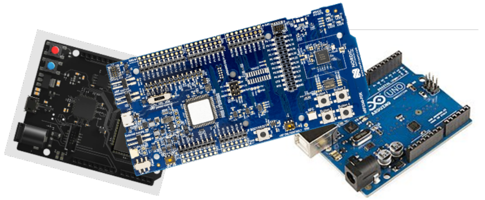
Unified ML compilation stack for CPU, GPU, NPU built with TVM

What is microTVM?



using TVM
to run models
on bare-metal devices

microTVM works in places without...

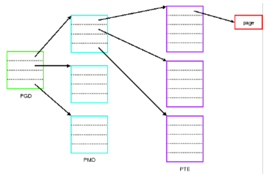


🚫 Operating Systems

- no files, DLLs, .so, memory mapping, kernels

🚫 Virtual Memory

- No malloc, C++ RAII, exceptions, …



🚫 Advanced Programming Languages

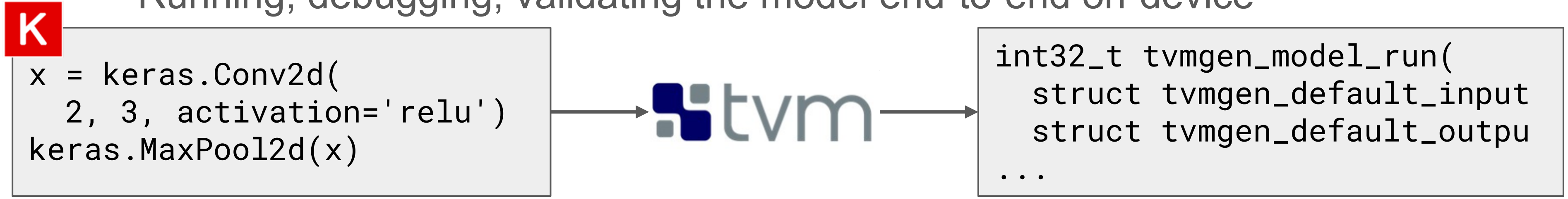
- No C++, Rust, Python, …
  (But we like those and you could use them!)

The microTVM Vision

> Codify and Automate Model Deployment onto any hardware platform

Including:

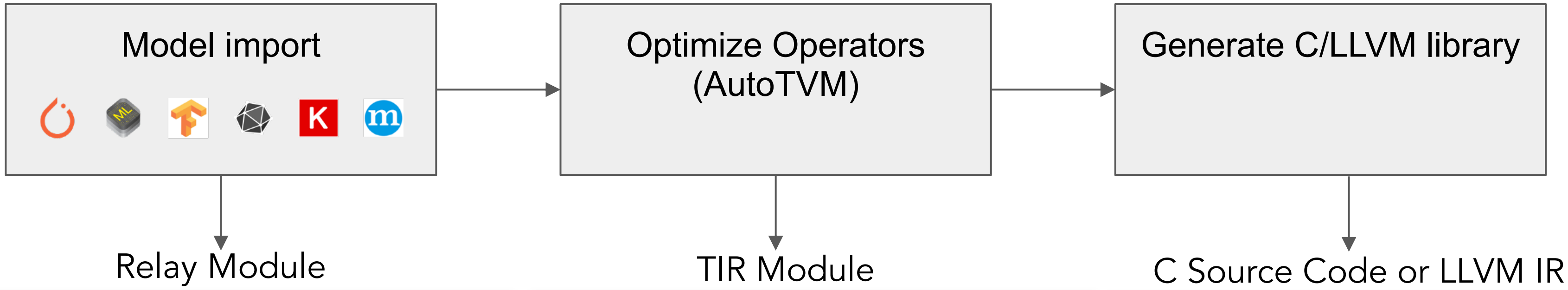
- Tailoring the model to fit
- Deciding how to leverage the hardware available
- Optimizing the model for application and platform
- Running, debugging, validating the model end-to-end on-device



```
x = keras.Conv2d(
  2, 3, activation='relu')
keras.MaxPool2d(x)
```

```
int32_t tvmgen_model_run(
  struct tvmgen_default_input
  struct tvmgen_default_outpu
...
```

Languages of the TVM Compiler

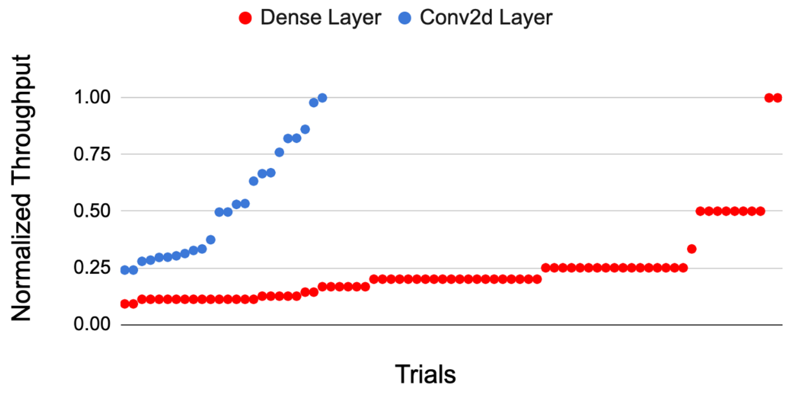


```
#[version = "0.0.5"]
def @main(%data : Tensor[(1, 3, 64, 64), int8],
          %weight : Tensor[(8, 3, 5, 5), int8]) {
    %1 = nn.conv2d(
         %data,
         %weight,
         padding=[2, 2],
         channels=8,
         kernel_size=[5, 5],
         data_layout="NCHW",
         kernel_layout="OIHW",
         out_dtype="int32");
  %3 = right_shift(%1, 9);
  %4 = cast(%3, dtype="int8");
  %4
}
```

```
 primfn(placeholder_2: handle,
        placeholder_3: handle,
        T_cast_1: handle) -> ()
  allocate(kernel_vec, int8, [600]) {
    for (bs.c.fused.h.fused: int32, 0, 64)
"parallel" {
      for (w: int32, 0, 64) {
        for (vc: int32, 0, 3) {
          data_vec[(((bs.c.fused.h.fused*192) +
(w*3)) + vc)] =
(uint8*)placeholder_5[(((vc*4096) +
(bs.c.fused.h.fused*64)) + w)]
        }
      }
    }
    // ...
```

```
int32_t
fused_nn_contrib_conv2d_NCHWc_right_shift_cast(
void* args, void* arg_type_ids,
int32_t num_args, void* out_ret_value,
void* out_ret_tcode, void* resource_handle) {
  void* data_pad = TVMBackendAllocWorkspace(1,
dev_id, (uint64_t)13872, 1, 8);
  for (int32_t i0_i1_fused_i2_fused = 0;
i0_i1_fused_i2_fused < 68;
++i0_i1_fused_i2_fused) {
    for (int32_t i3 = 0; i3 < 68; ++i3) {
      for (int32_t i4 = 0; i4 < 3; ++i4) {

((uint8_t*)data_pad)[((((i0_i1_fused_i2_fused *
204) + (i3 * 3)) + i4))] = (((((2 <=
i0_i1_fused_i2_fused) && (i0_i1_fused_i2_fused
< 66)) && (2 <= i3)) && (i3 < 66)) ?
((uint8_t*)placeholder)[(((((i0_i1_fused_i2_fus
ed * 192) + (i3 * 3)) + i4) - 390))] :
(uint8_t)0);
```

Why TIR?



- Separates implementation from definition
  - Relay is functional, TIR is procedural

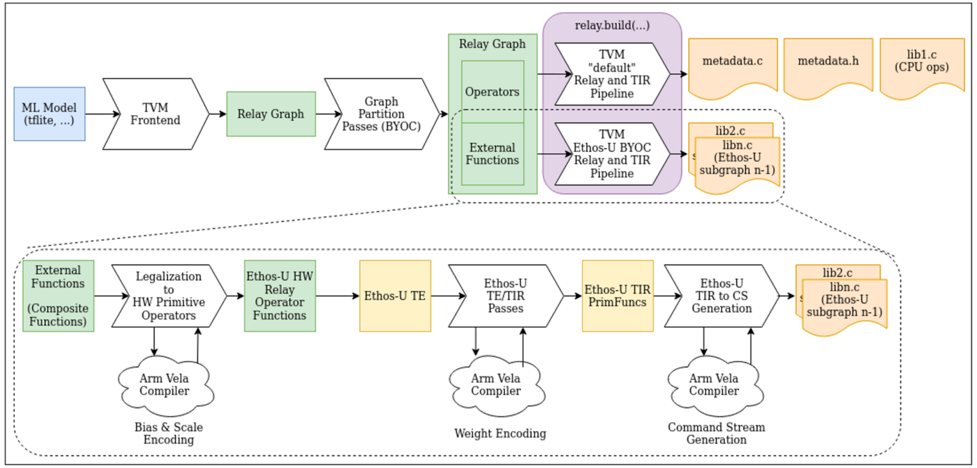
- TVM can automate implementation of layers
  - AutoTVM and MetaScheduler allows TVM to automatically tile, reorder, loop-unroll, etc
  - Future: Auto-tensorization allows TVM to generate an implementation given knowledge of vector intrinsics

- Define procedural optimizations one level above codegen
  - Example: common subexpr elimination could be used with CPU, GPU, or even with expression that span both

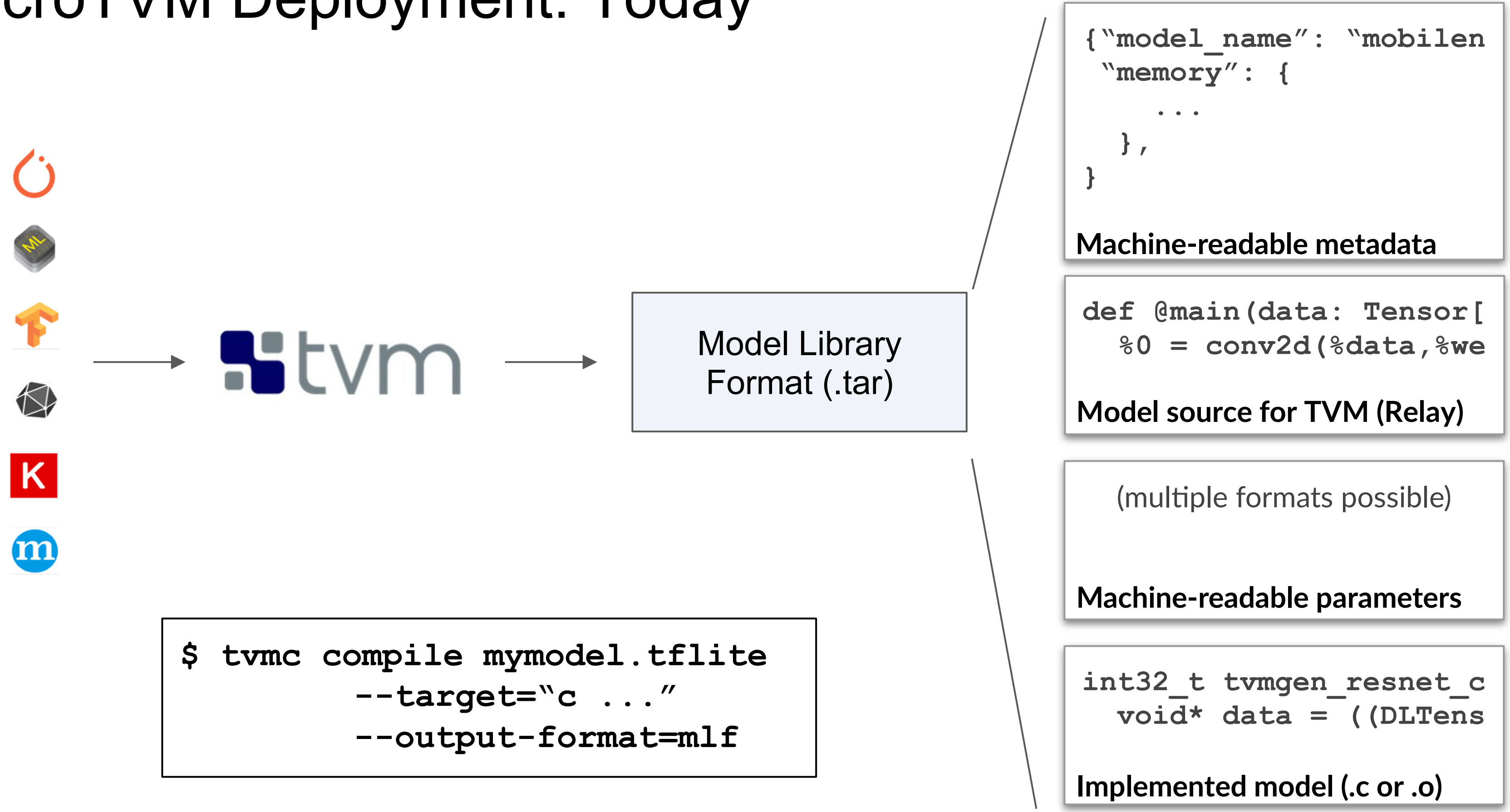
- Graph-level modeling in TIR
  - Memory allocations, buffer copies, and operator order can all be determined ahead-of-time

Accelerating microTVM by hand

- Hand-tuned implementations can complement TVM's optimization strategies
- Software libraries
  - CMSIS-NN for microTVM
  - Other examples: TensorRT, CuBLAS, CUTLASS, etc.
- Hardware accelerators
  - Ethos-U for ARM platforms


ML Model (tflite, ...)
TVM Frontend
Relay Graph
Graph Partition Passes (BYOC)
Relay Graph
Operators
External Functions
relay.build(...)
TVM "default" Relay and TIR Pipeline
TVM Ethos-U BYOC Relay and TIR Pipeline
metadata.c
metadata.h
lib1.c (CPU ops)
lib2.c
libn.c (Ethos-U subgraph n-1)
External Functions (Composite Functions)
Legalization to HW Primitive Operators
Ethos-U HW Relay Operator Functions
Ethos-U TE
Ethos-U TE/TIR Passes
Ethos-U TIR PrimFuncs
Ethos-U TIR to CS Generation
lib2.c
libn.c (Ethos-U subgraph n-1)
Arm Vela Compiler
Bias & Scale Encoding
Arm Vela Compiler
Weight Encoding
Arm Vela Compiler
Command Stream Generation

microTVM Deployment: Today



```
{“model_name”: “mobilen
 “memory”: {
      ...
   },
}
```

Machine-readable metadata

```
def @main(data: Tensor[
   %0 = conv2d(%data,%we
```

Model source for TVM (Relay)

(multiple formats possible)

Machine-readable parameters

```
int32_t tvmgen_resnet_c
   void* data = ((DLTens
```

Implemented model (.c or .o)

```
$ tvmc compile mymodel.tflite
           --target=“c ...”
           --output-format=mlf
```

microTVM Deployment: Today

```
{"model_name": "mobilene
 "memory": {
     ...
   },
}
```

Machine-readable metadata

```
def @main(data: Tensor[
  %0 = conv2d(%data,%we
```

Model source for TVM (Relay)

(multiple formats possible)

Machine-readable parameters

```
int32_t tvmgen_resnet_c
  void* data = ((DLTens
```

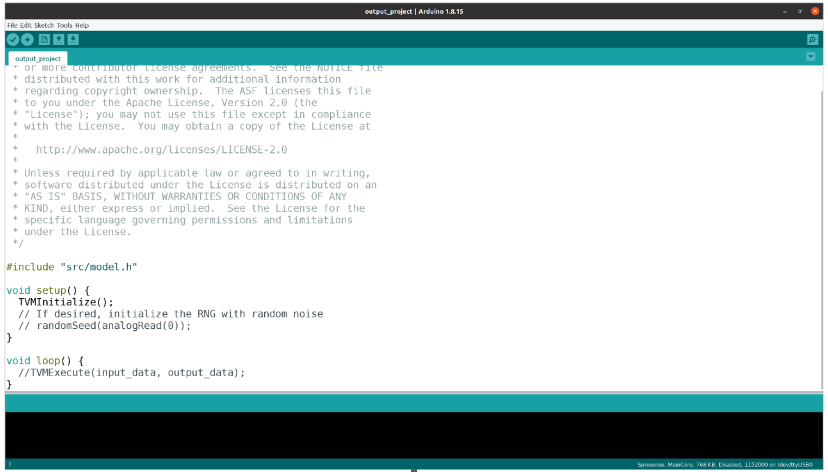
Implemented model (.c or .o)

```
$ tvmc micro generate-project
    -t arduino
    mymodel.tflite
```

TVM C Runtime

Firmware main()

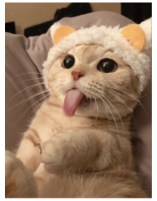
Platform libraries

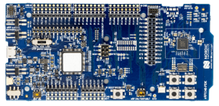


```
 * or more contributor license agreements.  See the NOTICE file
 * distributed with this work for additional information
 * regarding copyright ownership.  The ASF licenses this file
 * to you under the Apache License, Version 2.0 (the
 * "License"); you may not use this file except in compliance
 * with the License.  You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing,
 * software distributed under the License is distributed on an
 * "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
 * KIND, either express or implied.  See the License for the
 * specific language governing permissions and limitations
 * under the License.
 */

#include "src/model.h"

void setup() {
  TVMInitialize();
  // If desired, initialize the RNG with random noise
  // randomSeed(analogRead(0));
}

void loop() {
  //TVMExecute(input_data, output_data);
}
```

7
Spresense, MainCore, 768 KB, Disabled, 1152000 on /dev/ttyUSB0

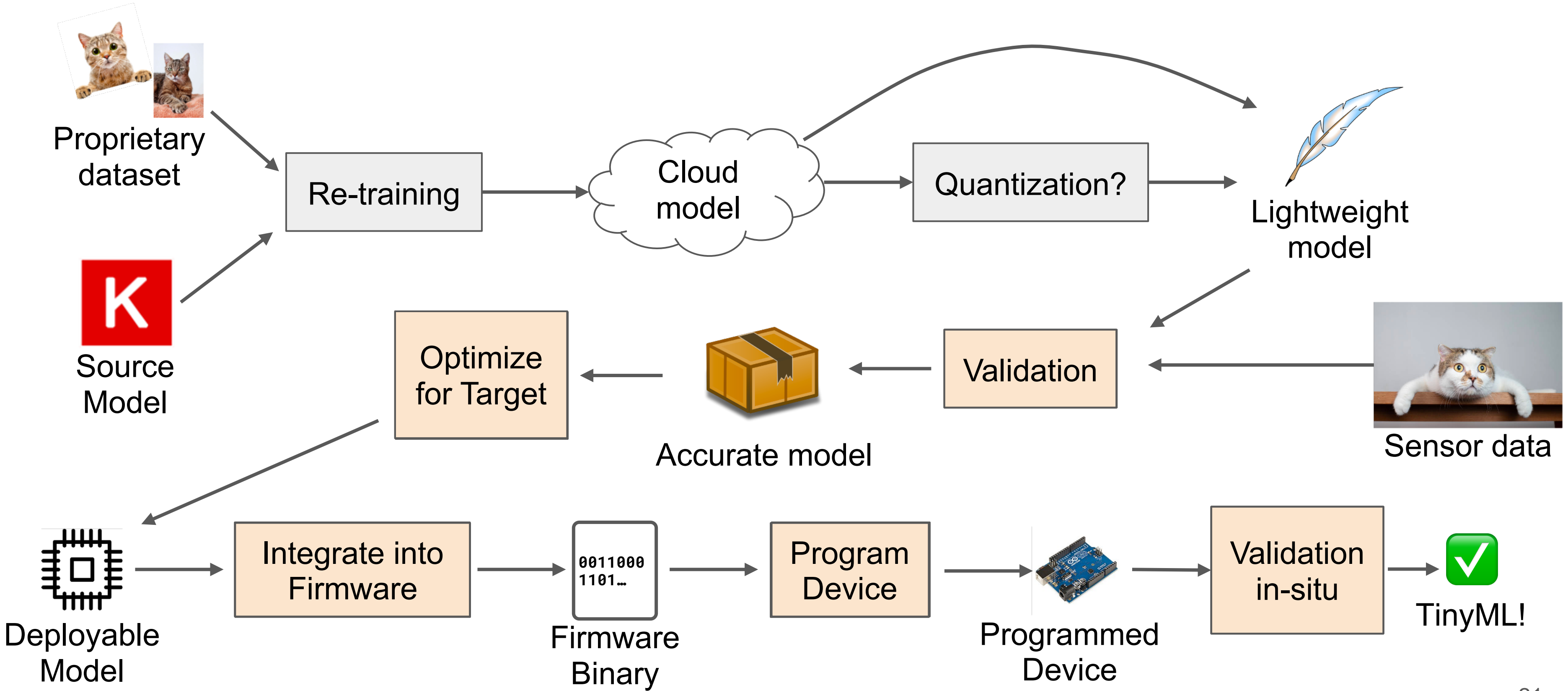
Compile / flash





"cat"

microTVM as a Workflow



Recently landed work in microTVM

- Support for automatic optimization via AutoTVM
- Whole-program memory planning: Unified Static Memory Planner
- Automatically create Arduino projects from an arbitrary model

Upcoming work in microTVM

- UMA - Universal Modular Accelerator infrastructure
  - API specifically designed to guide the addition of accelerator-specific compilation flows
- C Device API – first-class support for accelerators in the microTVM runtime
- Improved CLI support for AutoTVM using `tvmc`
- `tvm` (and microTVM) coming to PyPI
  - `pip install apache-tvm`

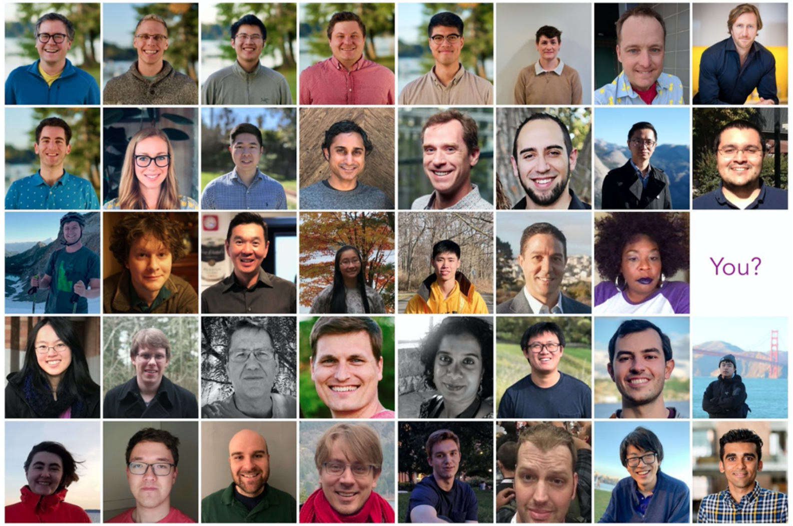
Getting Involved

- microTVM is a community-driven effort, and we welcome new contributions!
- See the microTVM Roadmap for a sense of planned work you can get involved with
- We welcome new ideas and questions on our Discuss forum and Discord, and new contributors to join our weekly TVM Community Meeting

https://tvm.apache.org/community

By the way, we're hiring!

OctoML



OctoML started in July 2019 - $130M in venture capital, now 100+ people and **growing**!

We're hiring Software Engineers, Product Managers, Researchers, and more!

https://octoml.ai/careers

Or email me: **areusch@octoml.ai**

Thank you!

- Much of the work summarized today was authored and contributed by members of the TVM Community.

- microTVM would not be where it is today without their help

TINY ML SUMMIT

tinyML Summit 2022 Sponsors

ANALOG DEVICES AHEAD OF WHAT'S POSSIBLE™ AONdevices arm ASPINITY brainchip The Neuromorphic Computing Company CEVA®

Deeplite EDGE IMPULSE emza visual sense FotaHub GREENWAVES TECHNOLOGIES Grovety Inc.

Himax HOTG imagimob infineon itemis KLIKA·TECH GLOBAL IOT SOLUTIONS

LatentAI LATTICE SEMICONDUCTOR Micro.ai OmniML NXP PXI Plumerai

PROPHESEE Qeexo Qualcomm Rackner RealityAI® Engineering Solutions for the Edge REEXEN technology

RENESAS SAP seeed studio The IoT Hardware Enabler SensiML™ Sony Semiconductor Solutions Corporation ST life.augmented SA STREAM ANALYZE

synaptics® SynSense SYNTIANT Tensil.ai TensorFlow XMOS

Copyright Notice

www.tinyml.org