μTVM

An AI Compiler and Framework for Arm Microcontrollers (and offload)
Gustavo Romero
Tom Gall
What is TVM?

An ML Framework with agnostic compilation stack, capable of targeting CPUs, GPUs, DSPs and NPUs

Key characteristics
- Apache Project
- Open Source - Apache version 2
- https://tvm.apache.org
- Supports a range of hardware, microcontrollers to very large
TVM Architecture

High-Level Differentiable IR

Tensor Expression and Optimization Search Space

LLVM, CUDA, Metal  VTA

Optimization

AutoTVM

Device Fleet

Image credit: TVM Project
Great, wasn’t this about microcontrollers?
Inference on Microcontrollers - Issues

- Model Ingestion
- Cartesian product of models, frameworks and hardware
- Optimizations for microcontrollers must tune for a wider set of characteristics such as power, working set size, offload
- Quantization, pruning and distillation yet retaining accuracy is critical

- Model integration with application logic
- Integration with variety of RTOSes (everyone has a favorite) or RTOSless
- Support a range of hardware targets with different offload capabilities
- Deployment at scale
µTVM

µTVM is part of the TVM project with features tailored to deeply embedded environments

- Leverages TVM framework
- Minimal runtime
- RPC for interactive development
- Zephyr, (Mbed OS) or RTOSless(*)
- C, C++ / Python language bindings

µTVM is a work in progress!
Development Workflow

TVM supports a wide range of models.
**µTVM ARM targets**

- Zephyr RTOS
  - µTVM -> Zephyr application
  - Supported boards:
    - ST NUCLEO-F746ZG board, Cortex-m7
    - ST STM32F746 Discovery board, Cortex-m7
    - MPS2-AN521 (QEMU), Cortex-m33 (WIP)
    - MPS3-AN547 (emulated), Cortex-m55 (planned)
  - Issues:
    - Zephyr support (WIP)
    - No QEMU support
**TVMC**

- **tvmc** is the cli for TVM
- Two main use-cases: **a)** development workflow and **b)** integration with other projects (Makefile, etc)
- Currently only supports TVM targets (no microTVM targets)
- RFC proposed to add µTVM targets [1],
- Relies on two other RFCs:
  - The Model Library Format [2]
  - The Project API [3]
- **tvmc** allows to split TVM/µTVM workflow in independent stages: compile, run, and autotune

- The workflow stages that generate artifacts (like compile) use the Model Library Format to store the artifacts
- The Project API will allow a better integration with various RTOSes
- For **tvmc** supporting the µTVM targets, additional stages specific to µTVM are proposed under new context 'micro':
  - $ tvmc compile --target="c" ... (compile relay)
  - $ tvmc micro create-project --type zephyr ... --output=dir
  - $ tvmc micro build --input=dir --model compiled.tgz
  - $ tvmc micro flash --input=dir
  - $ tvmc micro run

---

[1] [https://discuss.tvm.apache.org/t/rfc-tvmc-add-support-for-tvm/9049/6](https://discuss.tvm.apache.org/t/rfc-tvmc-add-support-for-tvm/9049/6) - [RFC] TVMC: Add support for µTVM (WIP)
[3] To be published soon - Project API RFC
µTVM Binary

Low-level SoC control
- Device Startup
- Peripheral Support
- Time Control Libraries
- Vendor Code
  - Libraries - Example STM32F7Cube
- RTOS
  - Zephyr, MbedOS, FreeRTOS, Azure RTOS, etc..

µTVM RPC and Runtime
- MISRA-C Runtime
- µTVM RPC Server
  - Enables interactive development
  - Driven by developers workstation
- Framing and Session

Compiled TVM operations
- Generated by TVM specific to user workload
- Maybe linked separately depending on RAM requirements
Example: running µTVM on ST discovery board

What: Interactive development example in Python (TVM repo, in tutorials/micro/micro_tflite.py)

1. Load model
   a. TensorFlow Lite format
   b. Prediction of $\sin(x)$
2. Compile
3. Run uses Zephyr 2.4.0

Intel Host PC (Debian/Ubuntu)
- Native ARM hosted works too
- USB -> STM32F746 Discovery Board via STLink (OpenOCD, from Zephyr SDK)
Load model from `.tflite`

```python
137 tflite_model_buf = open(model_path, "rb").read()
142 import tflite
144 tflite_model = tflite.Model.GetRootAsModel(tflite_model_buf, 0)
```

Parse Python model object to convert it into a **relay module** and **weights**

```python
169 mod, params = relay.frontend.from_tflite(
170    tflite_model, shape_dict={input_tensor: input_shape},
    dtype_dict={input_tensor: input_dtype}
)
```

Specify the TARGET. It will be used by TVM to generate C source using LLVM API internally

```python
193 TARGET = tvm.target.target.micro("stm32f746xx")
194 BOARD = "stm32f746g_disco" # or "nucleo_f746zg"
```

Given the TARGET, generate runtime graph (.json), C sources (**ops**), and model parameters per layer

```python
202 graph, c_mod, c_params = relay.build(mod, target=TARGET, params=params)
```

Pick up a compiler using TARGET (and BOARD) to compile C sources into native code for the MCU

```python
227 compiler = zephyr.ZephyrCompiler(
228    project_dir=project_dir,
229    board=BOARD if "stm32f746" in str(TARGET) else "qemu_x86",
230    zephyr_toolchain_variant="zephyr",
240    micro_binary = tvm.micro.build_static_runtime(
241        workspace, compiler,
242        with tvm.micro.Session(binary=micro_binary, flasher=flasher) as session:
243            graph_mod = tvm.micro.create_local_graph_runtime(
258                graph, session.get_system_lib(), session.context
260            )
```

**RUN**

```python
269 graph_mod.set_input()
270 graph_mod.run()
```

Build fw img using the specified compiler

**Set input and run the model**

Flash and open RPC session
Plans / Code / Coordination

Community Project RFCs

- Standalone uTVM Roadmap -
  https://discuss.tvm.apache.org/t/rfc-tvm-standalone-tvm-roadmap/6987
- M2 https://discuss.tvm.apache.org/t/tvm-microtvm-m2-roadmap/8821

Pull Requests

- Community: https://github.com/apache/tvm/pulls
- Linaro: https://collaborate.linaro.org/display/AIML/AI+on+Microcontrollers

Coordination - Contact tom.gall@linaro.org

- Linaro AI Project
- Weekly Sync meetings on Wednesday + Slack channel
- Product Level Spec - Linaro Member companies set goals for μTVM to propel it towards a product ready state
- https://collaborate.linaro.org/display/AIML/AI+on+Microcontrollers
µTVM - How to get involved

Source - https://github.com/apache/tvm
Contributor Guide - https://tvm.apache.org/docs/contribute/
Forum - https://discuss.tvm.ai
Slack - tvmai.slack.com
Bugs / Issues - https://github.com/apache/tvm/issues
Thank you

Accelerating deployment in the Arm Ecosystem