muRISCV-NN: Efficient Deep-Learning Kernels for RISC-V

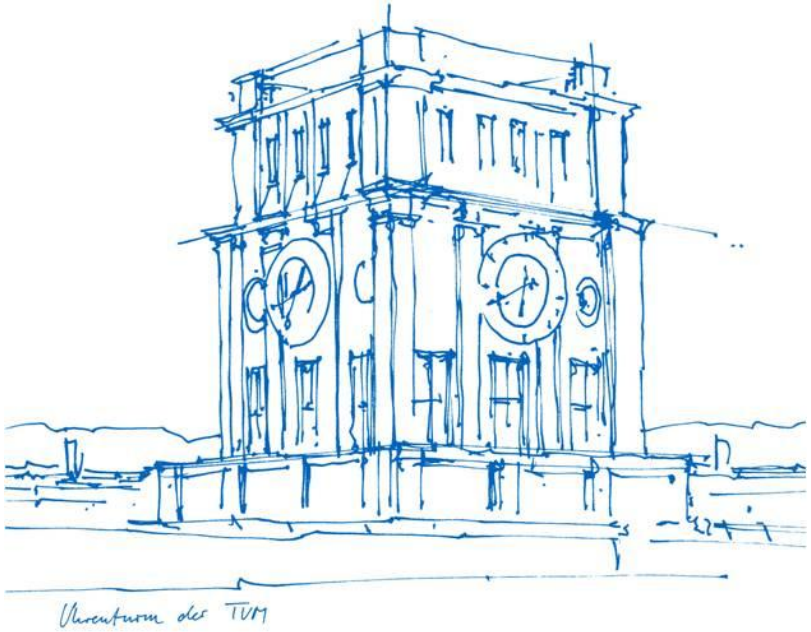
Fabian Peddinghaus, Philipp van Kempen,

Rafael Stahl, Daniel Müller-Gritschneder

Technical University of Munich

Chair for Electronic Design Automation

1st of September 2022



Deep Learning Workloads

Compression,
Anomaly Detection

FC / Deep AutoEncoder

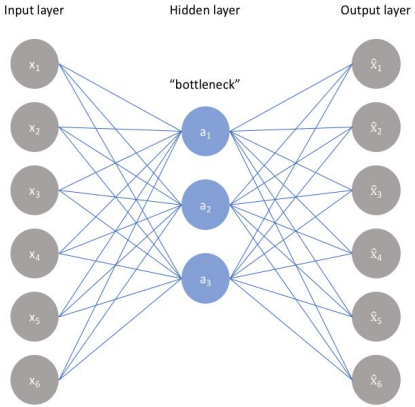
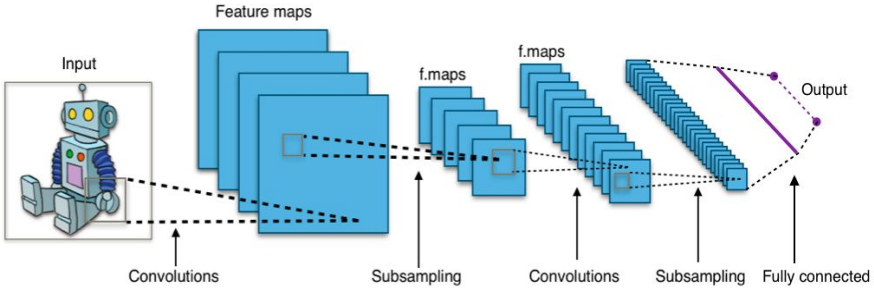


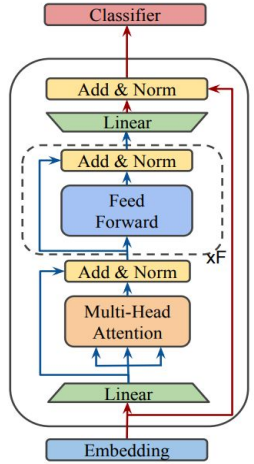
Image Analysis,
Object Detection

Convolutional Neural Network (CNN)



Natural Language
Processing

**Transformer
(mobileBert)**

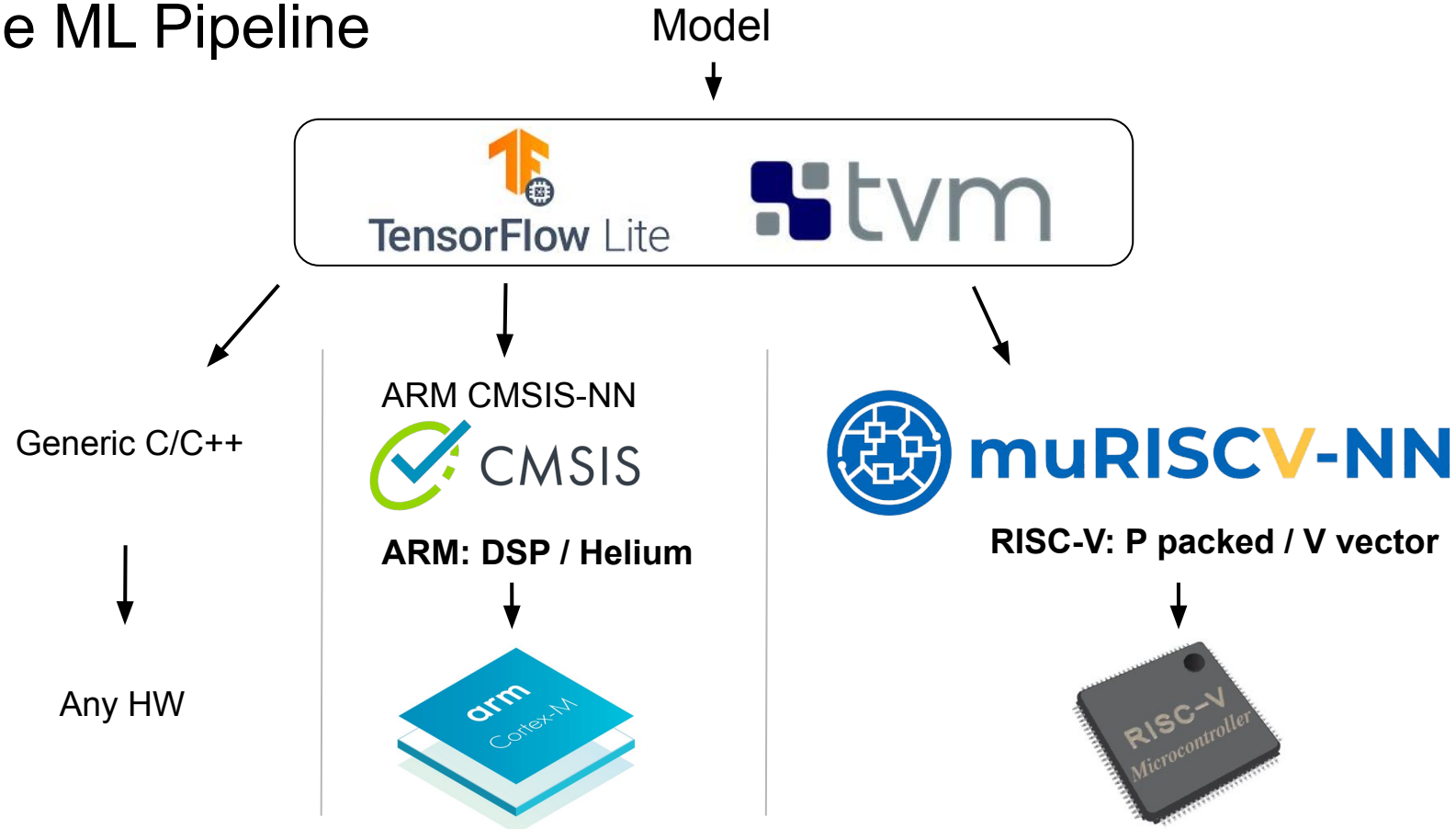


Why Deep Learning at the Edge?

- **BLERP** acronym
  - Mitigates **B**andwidth constraints
  - Reduces **L**atency, which is important for real-time applications
  - Improves **E**conomics, by reducing costs of transmitting data and processing it in the cloud
  - Increases **R**eliability, as on-device models are inherently more reliable than connection to the cloud
  - Improves **P**rivacy, when data is processed locally and never transmitted to the cloud

- **TinyML**
  - ML at the edge in the mW range
  - Only loosely defined term

Edge ML Pipeline



muRISCV-NN

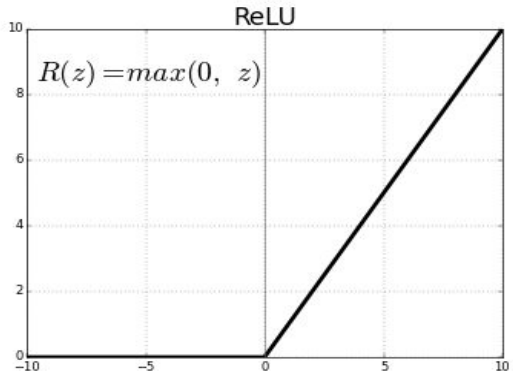
Efficient deep learning kernels for embedded platforms and microcontrollers.

- CMSIS-NN fork
  - Same unit tests
  - → **functionally equivalent!**
- RISC-V extension
  - **P packed** 0.9.6
  - **V vector** 1.0 (Zve32x)
- Toolchain
  - **RISC-V GCC**
  - **LLVM 14**
- Integration with
  - **TensorFlow Lite Micro**
  - **TVM**
- Simulator support
  - **Spike**
  - **riscvOVPsim**
  - **ETISS**
  - **(QEMU)**

RISC-V V Extension: ReLU Example

Reference implementation in plain C

```
muriscv_nn_status muriscv_nn_relu_q7(q7_t *data, const uint16_t size) {
    for (uint16_t i = 0; i < size; i++) {
        if (data[i] < 0) data[i] = 0;
    }
    return MURISCV_NN_MATH_SUCCESS;
}
```



Assembly implementation

```
# a0=data, a1=size
for:
   vsetvli t0, a1, e8, m8, ta, ma  # Vectors of 8bit
   vle8.v v0, (a0)                 # Load bytes
   vmax.vx v0, v0, zero            # Apply activation
   vse8.v v0, (a0)                 # Store bytes
   add a0, a0, t0                  # Decrement size
   sub a1, a1, t0                  # Bump pointer
   bnez a1, for                    # Any more
```

Intrinsics implementation in C

```
for (size_t cnt = size, vl = 0; cnt > 0; cnt -= vl, data += vl) {
    vl = vsetvl_e8m8(cnt);
    vint8m8_t val = vle8_v_i8m8(data, vl);
    val = vmax_vx_i8m8(val, 0, vl);
    vse8_v_i8m8(data, val, vl);
}
```

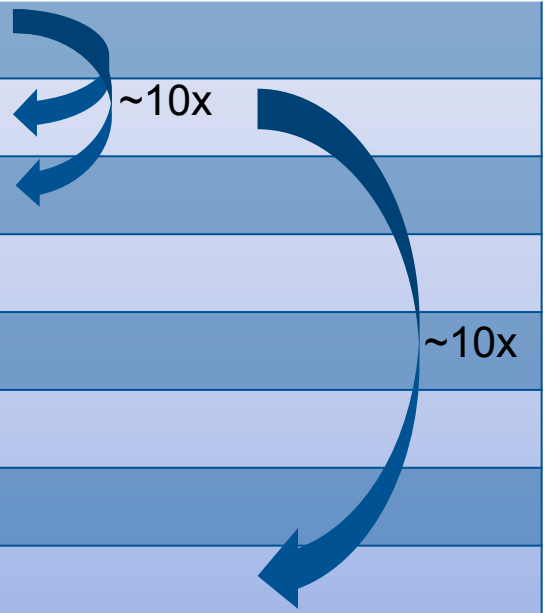
muRISCV-NN Results

- Models: TinyMLPerf full int8 quantized (https://github.com/mlcommons/tiny)
  - “toycar”: Audio anomaly detection (Deep AutoEncoder)
  - “resnet”: ImageNet detector (ResNet)
  - “aww”: Audio keyword spotting (DS-CNN)
  - “vww”: Visual person detection (MobileNet)

- Using TensorFlow Lite Micro
- Compiled with GCC (-Os) for RV32GC(P/V)
- Running on Spike ISS

muRISCV-NN Results: ResNet, TFLM, GCC -Os, Spike

Kernels	Extension	VLEN	Run time [x10^6 Instructions]
Default	-	-	688
muRISCV-NN	-	-	64.8
muRISCV-NN	P-Ext.	-	62.5
muRISCV-NN	V-Ext.	64	37.1
muRISCV-NN	V-Ext.	128	20.3
muRISCV-NN	V-Ext.	256	11.8
muRISCV-NN	V-Ext.	512	7.81
muRISCV-NN	V-Ext.	1024	5.98



muRISCV-NN Results: Other TinyMLPerf Models

In Million Instructions

Model	TFLM	muRISCV-NN	+P-Ext.	+V-Ext. 64	+V-Ext. 1024
toycar	3.43	1.92	1.12	0.728	0.536
aww	154	16.6	13.9	8.86	2.11
ResNet	688	64.8	62.5	37.1	5.98
vww	434	49.7	41.5	26.8	8.73

muRISCV-NN Results: Compared to ARM CMSIS-NN

In Million Instructions

Model	TFLM	CMSIS-NN	+DSP	+Helium
toycar	0.574	0.241	0.086	0.045
aww	8.44	3.64	1.04	0.333
ResNet	223	125	40.6	14.8
vww	207	109	32.8	10.2
			Comparable to P-Ext.	Comparable to V-Ext.

on Corstone-300 simulator

RISC-V Vector Cores

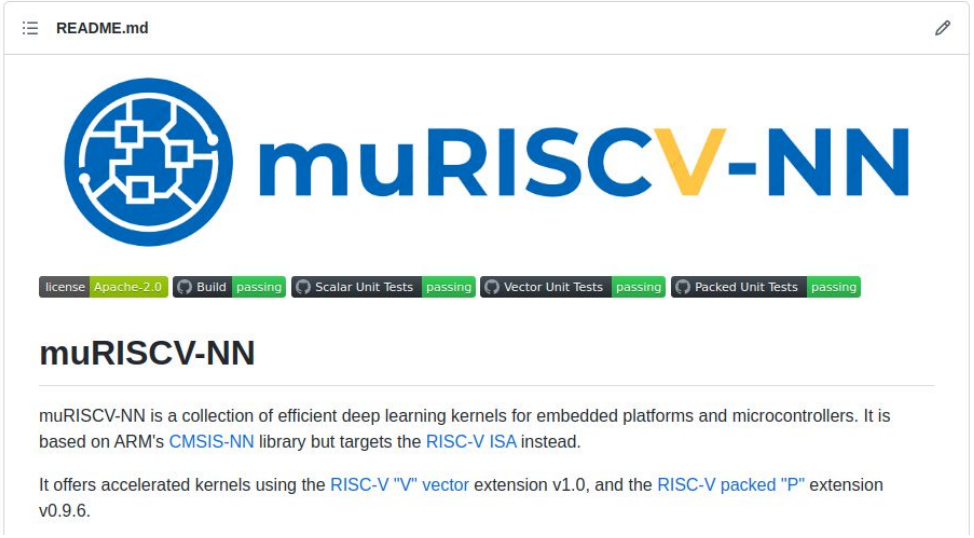
Academic
- **Hwacha** - UC Berkeley
- **Ara** - ETH Zurich
- **Vitruvius+** - BSC
- **RISC-V²** - Univ. of Thrace
- …

Academic Embedded **Zve32x**
- 2021 - **Vicuna** - TU Wien
- 2022 - **Spatz** - ETH Zurich
- …

Commercial
- **OpenC906 / OpenC910** - Alibaba
- **P270 / X280** - SiFive
- **NX27V** - Andes
- **NS-72 / DR1000C** - NSITEXE
- …

Future Work

- Lots of **optimization potential** inside kernels
- Move kernels from **application V** extension to **embedded Zve32x** extension
- Support **RTL simulation** on academic cores
- Support **hardware targets** (as soon as they come to market)



https://github.com/tum-ei-eda/muriscv-nn