XILINX

MLIR Open Meeting

End-to-end flow for ML Compiler

Stephen Neuendorffer

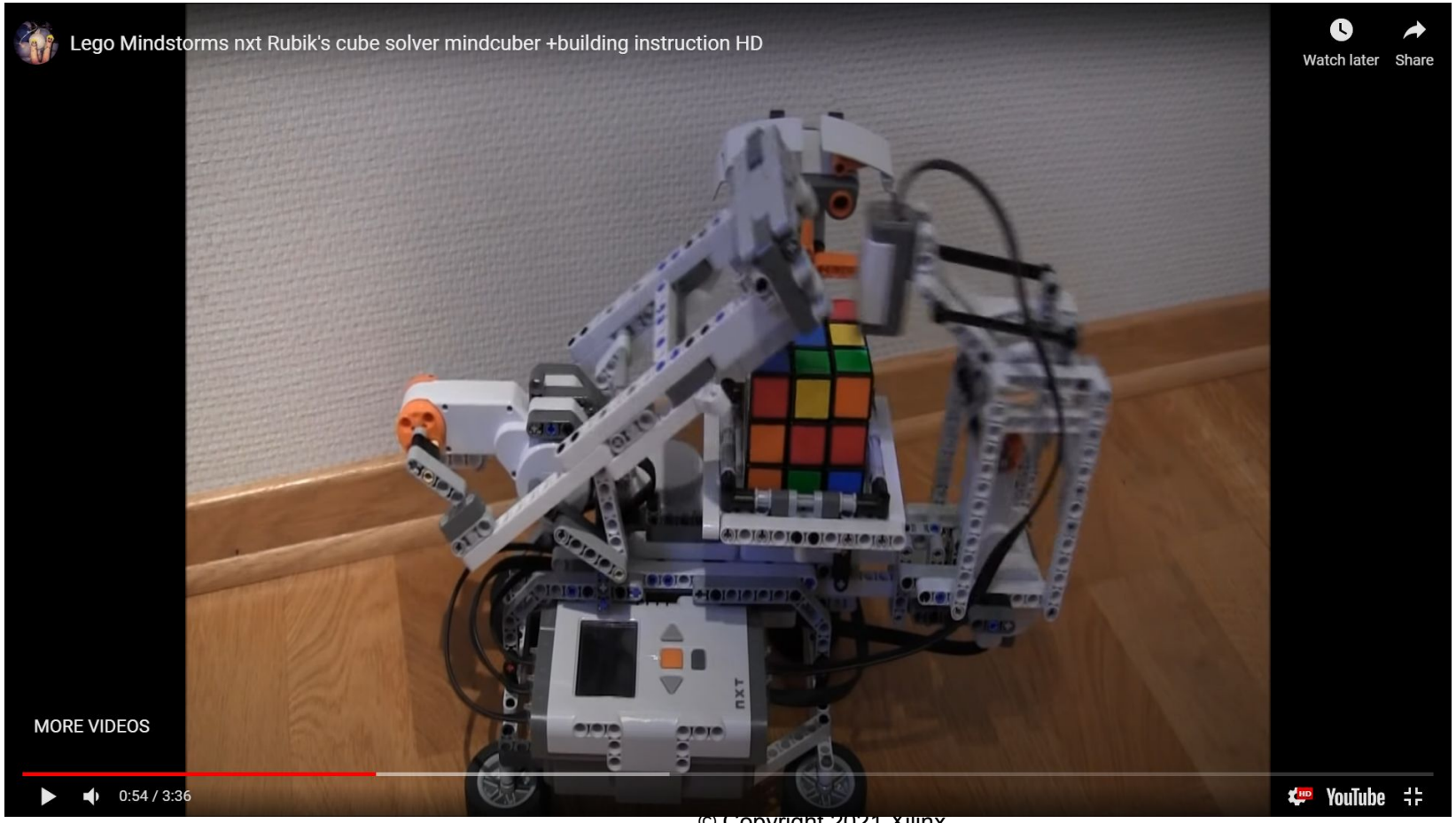
Dec 2, 2021

Today

- MLIR has made lots of progress as a *framework* for Machine Learning

Proposal

- We believe that an important next step in this progress is an end-to-end community machine-learning flow working out of the box.



XILINX.

What This Means to Me

Out-of-the-box

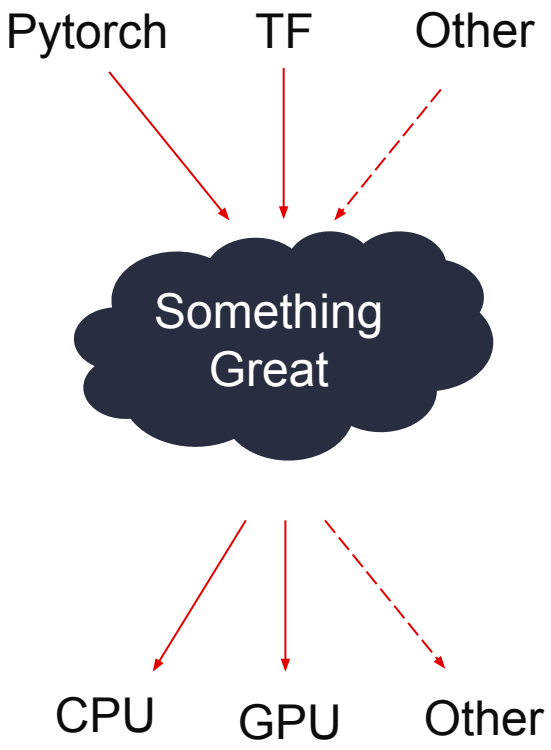
Download, Compile, Run

End-to-End

ML design to running Code

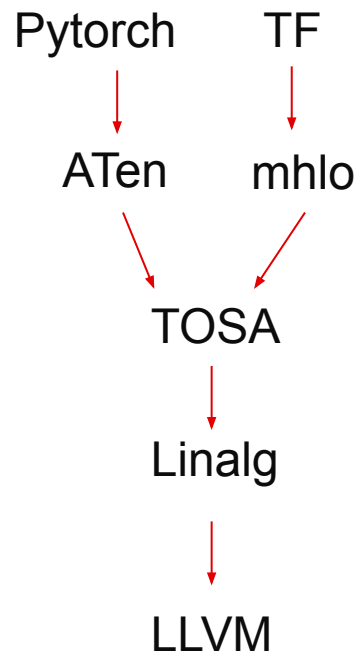
Community-based

Target commodity CPU, GPU



What do we have to work with

- Dialects
  - torch-mlir/ATen
  - onnx-mlir
  - TOSA
  - Linalg
  - Affine
  - GPU
  - LLVM
- Lowering paths between (most of) them



(One possibility... not to scale)

What does success look like?

- MLCommons benchmarks published using MLIR
- ML frameworks build on MLIR, rather than the other way around
- People stop asking me why I'm working on MLIR rather than XYZ

Existing progress

- https://github.com/google/iree-samples/tree/main/iree-torch
  - Currently factor of 2 slower performance than alternatives...
- @marbre: TF -> TOSA -> EmitC -> C++

Open Questions

- What level of performance is sufficient?
- Target Libraries vs. 'full codegen'?
  - potential libraries: Eigen, TOSA reference, Torch, cudnn
- What data layouts do we assume?
- What level of 'runtime support' should be expected/required?
- What 'backend target' interfaces do we need to allow extensibility?
- What existing use models should we replicate/model after?
  - e.g. TVM flow to build an inference .dll

Open Questions (2)

- How do we get the out-of-tree bits in the right spot?
- Training and/or inference?
- Distributed execution?