MLIR

Accelerating AI
Data drives the continuous improvement cycle for ML models.

Researchers provide new algorithmic innovations unlocking new techniques and models.

Compute allows it all to scale as datasets get larger and algorithms need to scale on that accordingly.
Models are growing and getting more complex

**Model Size**: larger models require more multiply accumulate operations.

**Model Complexity**: as model complexity increases it becomes harder to fully utilize hardware.

Source: OpenAI - AI & Compute
Moore’s law ending: new hardware needed

Source: 40 years microprocessor trend data
Explosion in HW startups & custom chips

**Billions of dollars** of funding going towards new ML ASICs

**Rapid expansion** of heterogeneous hardware solutions

### Significant growth in deals and funding

**ML semiconductors global funding history**

($M, # of deals)
Ongoing explosion in Edge Hardware

~5.5B Mobile Phones

250B+ Microcontrollers

Edge TPUs
Heterogeneous hardware is now the norm

Scaling from phones down to microcontrollers

Memory, energy, performance and latency constraints become paramount
Datacenters

GPUs
CPUs
TPUs

and other hardware accelerators
More hardware features: complexity

- Many different hardware accelerators focused on ML
- Many different types and architectures: 4-bit, 16-bit, 32-bit...
- Inability to quickly scale up and down hardware consistently and varying levels of abstractions
Vibrant and expanding SW ecosystem

- Many frameworks & standards proposed
- Many different graph implementations
- Each framework is trying to gain a usability and performance edge over each other
None of this is scaling!
What’s wrong?

- Systems don't interop
- Can’t handle all operators and types consistently on all hardware
- There is poor developer usability and debuggability across hardware
- No generalizable standard for ensuring software and hardware scales together
- Everyone is trying to build the same thing at great cost
- Fragmentation exists everywhere in the market today
So what would we want?
Common infrastructure: building blocks

- Best in class graph and compiler technology
- Designed for both training and inference, mobile and server
- The ability to scale ML from the edge all the way to the server
- A standard representation for types and operators, custom ops
- Framework independent
- Neutral governance
The industry agrees

- Largest HW partners in the world
- 95% of the world’s data-center accelerator hardware
- 4 billion mobile phones countless IoT devices
- Governance moved to LLVM
What is MLIR?

A new **compiler infrastructure** that enables machine learning models to be consistently represented and executed **on any type of hardware**.
How is MLIR different?

**State of Art Compiler Technology**
MLIR is NOT just a common graph serialization format nor is there anything like it.

**Modular & Extensible**
From graph representation through optimization to code generation.

**Not opinionated**
Choose the level of representation that is right for your device.
MLIR Compiler Infrastructure

**TLDR:** A common graph representation and legalization framework, a common set of optimization and conversion passes and a full code generation pipeline.
Enables many different approaches

You are free to utilize different components of the system as you need. MLIR can also be modularized as a graph rewriting tool like we do for TensorFlow Lite.
What does this mean for TensorFlow?
Building a better TensorFlow

The Same Public Interfaces

We are maintaining all the public interfaces:
GraphDef, SavedModel, Python APIs, HLO, XLA, XRT, xprof, ...

Hardware Representation

Better flexibility and representation across all hardware
Higher performance and more reliable

Stack Convergence

Increased stack convergence
A better and more consistent user experience
Easier to try new hardware
What does this mean for you as a Python Developer?
A better TensorFlow developer experience

MLIR will make your TensorFlow development experience so much better.

How?

- Enable consistent model across different hardware
- Enable better out-of-the-box performance
- Pinpoint mistakes back to the line of python code
/** Current TensorFlow error experience */

F0122 11:20:14.691357  27738 import_tensorflow.cc:2549] Check failed: status.ok() Unexpected value for attribute 'data_format'. Expected 'NHWC'

*** Check failure stack trace: ***

...

*** SIGABRT received by PID 27738 (TID 27738) from PID 27738; ***

F0122 11:20:14.691357  27738 import_tensorflow.cc:2549] Check failed: status.ok() Unexpected value for attribute 'data_format'. Expected 'NHWC'


Aborted
node "MobilenetV1/MobilenetV1/Conv2d_0/Conv2D" defined at 'convolution2d' tensorflow/contrib/layers/python/layers/layers.py:1156:
  conv_dims=2)
^

...
MLIR: propelling the industry
Neutral governance through LLVM

MLIR has been accepted as part of the nonprofit LLVM Foundation

This will enable even faster adoption of MLIR by the industry as a whole
MLIR is building a global compiler community to make ML better for everyone.
Together, let's advance the state of ML

MLIR is accelerating data, algorithmic and HW innovation through open infrastructure that ensures models will be represented, executed and scaled correctly.
MLIR: Accelerating AI for the world.
Thank you

github.com/tensorflow/mlir

Questions?
mlir@tensorflow.org