Welcome to the 1st 2nd TVM and Deep Learning Compilation Conference!
Welcome to the 1st 2nd TVM and Deep Learning Compilation Conference!

200+ ppl!
Welcome to the 1st 2nd TVM and Deep Learning Compilation Conference!

200+ ppl!
Machine learning era:

1. Problem to solve
2. Data + model templates
3. Train on fast $\text{machine}$
4. Inference on fast & cheap enough machine
Machine learning era: 

Problem to solve → Data + model templates → Train on *fa$te$t* machine → Inference on fast & cheap enough machine

Model size and compute cost growing fast
Machine learning era:

- Problem to solve
- Data + model templates
- Train on fastest machine
- Inference on fast & cheap enough machine

Training costs growing exponentially

Model size and compute cost growing fast

![Graph showing the increase in compute costs from AlexNet to AlphaGo Zero](image)

- By Eugenio Culurciello
- Training costs growing exponentially by Open AI
Machine learning era:

1. **Problem to solve**
2. **Data + model templates**
3. **Train on fastest machine**
4. **Inference on fast & cheap enough machine**

**Training costs growing exponentially**

- Model size and compute cost growing fast
- $1.5M in EC2 costs!

![Graph showing the increase in compute requirements from AlexNet to AlphaGo Zero](image)

by Eugenio Culurciello

Training costs growing exponentially

- Model size and compute cost growing fast
- $1.5M in EC2 costs!

![Graph showing the increase in compute requirements from AlexNet to AlphaGo Zero](image)

by Open AI
Machine learning era:

1. **Problem to solve**
2. **Data + model templates**
3. **Train on fast enough machine**
4. **Inference on fast & cheap enough machine**

### Model size and compute cost growing fast

- Inference costs growing exponentially
- $1.5M in EC2 costs!
- $1.2M home in the Bay Area... :)

By Eugenio Culurciello

![Graph showing model size and compute cost growth](image)

**AlexNet to AlphaGo Zero: A 300,000x Increase in Compute**

![Graph comparing AlexNet to AlphaGo Zero](image)

By Open AI
MIT Technology Review

Training a single AI model can emit as much carbon as five cars in their lifetimes

Deep learning has a terrible carbon footprint.

by Karen Hao  Jun 6, 2019

The artificial-intelligence industry is often compared to the oil industry: once mined and refined, data, like oil, can be a highly lucrative commodity. Now it seems the metaphor may extend even further. Like its fossil-fuel counterpart, the process of deep learning...
It gets more serious…
It gets more serious...

Computational cost of ML. Oops. :)

42 Years of Microprocessor Trend Data

Transistors (thousands)
Single-Thread Performance (SpecINT x 10^3)
Frequency (MHz)
Typical Power (Watts)
Number of Logical Cores

Impact of ML will be limited if we don’t squeeze as much efficiency as we can!
Impact of ML will be limited if we don’t squeeze as much efficiency as we can!

Model, SW and HW optimization are key…
A perfect storm
A perfect storm

Cambrian explosion of models, workloads, and use cases.

<table>
<thead>
<tr>
<th>CNN</th>
<th>GAN</th>
<th>RNN</th>
<th>MLP</th>
<th>DQNN</th>
</tr>
</thead>
</table>


A perfect storm

Growing set of requirements: **cost, latency, power, security & privacy**

Cambrian explosion of models, workloads, and use cases.

<table>
<thead>
<tr>
<th></th>
<th>CNN</th>
<th>GAN</th>
<th>RNN</th>
<th>MLP</th>
<th>DQNN</th>
</tr>
</thead>
</table>
A perfect storm

Growing set of requirements: **cost, latency, power, security & privacy**

Cambrian explosion of models, workloads, and use cases.

<table>
<thead>
<tr>
<th>CNN</th>
<th>GAN</th>
<th>RNN</th>
<th>MLP</th>
<th>DQNN</th>
</tr>
</thead>
</table>

Silicon scaling limitations (Dennard and Moore):

Cambrian explosion of HW backends. Heterogeneous HW.
A perfect storm

Growing set of requirements: **cost, latency, power, security & privacy**

Cambrian explosion of models, workloads, and use cases.

Rapidly evolving ML software ecosystem quickly fragmenting

Silicon scaling limitations (Dennard and Moore):

Cambrian explosion of HW backends. Heterogeneous HW.
Deep learning “stack” (r?)evolution

Lots of hand-tuning, full automation would be really nice…
Deep learning “stack” (r?)evolution

Lots of hand-tuning, full automation would be really nice…
## Current Dominant Deep Learning Systems Landscape

### Orchestrators
- Kubeflow
- Seldon
- Algorithmia
- Azure ML
- GCP Datalab

### Frameworks and Inference engines
- TensorFlow
- PyTorch
- ONNX
- TensorRT

### DL Compilers
- nGraph
- GLOW
- XLA
- MLIR

### Kernel Libraries
- cuDNN
- NNPack
- MKL-DNN

*Hand optimized*
# Current Dominant Deep Learning Systems Landscape

## Orchestrators
- Kubeflow
- Seldon
- Algorithmia
- Azure ML
- GCP Datalab

## Frameworks and Inference engines
- TensorFlow
- PyTorch
- ONNX
- TensorRT

## DL Compilers
- nGraph
- Glow
- XLA
- MLIR

## Kernel Libraries
- cuDNN
- NNPack
- MKL-DNN

## Hardware

Open source, **automated** end-to-end optimization framework for deep learning.
Using ML for better ML systems...

Deal with design complexity and large parameter spaces...
Using ML for better ML systems...

Deal with design complexity and large parameter spaces...

Model optimization strategies and parameters

Efficient operator implementations

Data communication patterns

Model-HW co-tuning

Searching for efficient HW designs
This past year...
This past year...

Broader model coverage (e.g., PyTorch integration, RelayVM, BERT, SSD)
This past year…

Broader model coverage (e.g., PyTorch integration, RelayVM, BERT, SSD)

More hardware backends (e.g., CortexM, RISC-V, DSPs)
This past year…

Broader model coverage (e.g., PyTorch integration, RelayVM, BERT, SSD)

More optimizations (e.g., quantization, data layout)

More hardware backends (e.g., CortexM, RISC-V, DSPs)
This past year...

**Broader model coverage (e.g., PyTorch integration, RelayVM, BERT, SSD)**

More optimizations (e.g., quantization, data layout)

**More hardware backends (e.g., CortexM, RISC-V, DSPs)**

Usability (tutorials, docs, automation), community development
70% growth from Dec 2018 to 295 contributors from UW, Berkeley, Cornell, UCLA, Amazon, Huawei, NTT, Facebook, Microsoft, Qualcomm, Alibaba, Intel, …
Open Source Community Growth and Impact

70% growth from Dec 2018 to 295 contributors from UW, Berkeley, Cornell, UCLA, Amazon, Huawei, NTT, Facebook, Microsoft, Qualcomm, Alibaba, Intel, …

Used in production at leading vendors:

- **aws**
  - Deep Learning Compiler Service

- **HUAWEI**
  - Tensor Engine for mobile ASIC

- **facebook**
  - Mobile and Server Optimizations

- **Microsoft**
  - Cloud-side model optimization
Open Source Community Growth and Impact

70% growth from Dec 2018 to 295 contributors from UW, Berkeley, Cornell, UCLA, Amazon, Huawei, NTT, Facebook, Microsoft, Qualcomm, Alibaba, Intel, …

Used in production at leading vendors:

- AWS: Deep Learning Compiler Service
- Huawei: Tensor Engine for mobile ASIC
- Facebook: Mobile and Server Optimizations
- Microsoft: Cloud-side model optimization

Incubated as Apache TVM recently. Independent governance, allowing competitors to collaborate.
Open Source Community Growth and Impact

70% growth from Dec 2018 to 295 contributors from UW, Berkeley, Cornell, UCLA, Amazon, Huawei, NTT, Facebook, Microsoft, Qualcomm, Alibaba, Intel, ...

Used in production at leading vendors:
- Deep Learning Compiler Service
- Tensor Engine for mobile
- Mobile and Server Optimization
- Cloud-side model optimization

Incubated as Apache TVM recently. Independent governance, allowing competitors to collaborate.
Jeff Gehlhaar
Qualcomm Technologies, Inc.

AI Overview

Jeff Gehlhaar, VP Technology
Qualcomm Technologies, Inc.
We’re creating a future of distributed intelligence

Our platforms are enabling a world of decentralized computing to realize the true potential of AI at scale. On-device inference processes data closest to the source for maximum speed and security, and low-latency 5G connectivity augments experiences with edge cloud processing for training updates and connected services.
Our process

We design and develop holistic AI systems

Our process provides a comprehensive approach to AI research and development. We take on hard problems and tackle complexity head on to meticulously design and build systems that deliver complete end-to-end AI solutions, from fundamental research to product execution.
Our AI software products

Qualcomm Neural Processing SDK, Qualcomm Hexagon and Qualcomm AI Engine are products of Qualcomm Technologies, Inc.
Qualcomm Neural Network Core

- **CPU ops**: QML, NEON
- **GPU ops**: Kernels, OpenCL
- **DSP ops**: Hexagon NN, HVX
- **Tensor ops**: HTA

- Android NN
- Qualcomm Neural Processing Model Loader

- CPU
- GPU
- DSP
Qualcomm Neural Processing Model Loader

Qualcomm Neural Network Core

CPU ops
- QML
- NEON

GPU ops
- Kernels
- OpenCL

DSP ops
- Hexagon NN
- HVX
- HTA

Tensor ops

CPU

GPU

DSP

HTA
Qualcomm Neon Processing Model Loader

Qualcomm Neural Network Core

CPU ops
- QML
- NEON

GPU ops
- Kernels
- OpenCL

DSP ops
- HVX

Tensor ops
- HTA

Hexagon NN

CPU ops
- Hexagon NN

GPU ops
- Hexagon NN

DSP ops
- Hexagon NN
Hexagon NN

- Currently supports ~100 ops
Hexagon NN

- Currently supports ~100 ops
- Handwritten and optimized across 3 different Hexagon architecture variations
Hexagon NN

- Currently supports ~100 ops
- Handwritten and optimized across 3 different Hexagon architecture variations
- Ops have to be written for both Hexagon Vector Extensions (HVX) and Hexagon Tensor Accelerator (HTA) units
Hexagon NN

- Currently supports ~100 ops
- Handwritten and optimized across 3 different Hexagon architecture variations
- Ops have to be written for both Hexagon Vector Extensions (HVX) and Hexagon Tensor Accelerator (HTA) units
- Incredible demand from customers to add new operators and operator variants
Hexagon NN

- Currently supports ~100 ops
- Handwritten and optimized across 3 different Hexagon architecture variations
- Ops have to be written for both Hexagon Vector Extensions (HVX) and Hexagon Tensor Accelerator (HTA) units
- Incredible demand from customers to add new operators and operator variants

Hexagon is a flexible and power efficient but complex IP block to program efficiently. Like Halide for CV applications, TVM gives us internal development advantage and gives customers a tool to develop custom operators.
Hexagon NN

- Currently supports ~100 ops
- Handwritten and optimized across 3 different Hexagon architecture variations
- Ops have to be written for both Hexagon Vector Extensions (HVX) and Hexagon Tensor Accelerator (HTA) units
- Incredible demand from customers to add new operators and operator variants

Hexagon is a flexible and power efficient but complex IP block to program efficiently. Like Halide for CV applications, TVM gives us internal development advantage and gives customers a tool to develop custom operators.

TVM is key to ML Access on Hexagon
Key Ideas and Innovations

Qualcomm Technologies, Inc. is a leader in silicon for on-device and cloud solutions.

Hexagon hardware provides a key power/performance advantage but is complicated to optimize.

TVM and domain specific languages are key for per-kernel and whole graph optimization strategies.

Our Qualcomm AI Research is advancing hardware aware optimization strategies.
Thank you

Nothing in these materials is an offer to sell any of the components or devices referenced herein.

©2018-2019 Qualcomm Technologies, Inc. and/or its affiliated companies. All Rights Reserved.

References in this presentation to “Qualcomm” may mean Qualcomm Incorporated, Qualcomm Technologies, Inc., and/or other subsidiaries or business units within the Qualcomm corporate structure, as applicable. Qualcomm Incorporated includes Qualcomm’s licensing business, QTL, and the vast majority of its patent portfolio. Qualcomm Technologies, Inc., a wholly-owned subsidiary of Qualcomm Incorporated, operates, along with its subsidiaries, substantially all of Qualcomm’s engineering, research and development functions, and substantially all of its product and services businesses, including its semiconductor business, QCT.

Follow us on:  
For more information, visit us at:  
www.qualcomm.com & www.qualcomm.com/blog
Yida Wang
AWS AI
AWS AI

• The broadest and most complete set of machine learning capabilities
  • AI Services
  • Amazon SageMaker
  • ML Frameworks & Infrastructure
AWS AI

• The broadest and most complete set of machine learning capabilities
  • AI Services
  • Amazon SageMaker
  • ML Frameworks & Infrastructure

• More machine learning happens on AWS than anywhere else
  • 81% of deep learning in cloud runs on AWS
TVM@AWS
TVM@AWS

• As a cloud service: Amazon SageMaker Neo
  • Train models once, run anywhere with up to 2x performance improvement
TVM@AWS

• As a cloud service: Amazon SageMaker Neo
  • Train models once, run anywhere with up to 2x performance improvement

• As a solution
  • Fastest model inference on a number of Amazon EC2 instances
  • Alexa Wakeword model on Amazon Echo
  • Collaborating with a number of external device makers
TVM@AWS

• As a cloud service: Amazon SageMaker Neo
  • Train models once, run anywhere with up to 2x performance improvement
• As a solution
  • Fastest model inference on a number of Amazon EC2 instances
  • Alexa Wakeword model on Amazon Echo
  • Collaborating with a number of external device makers
• As a research project
  • Three accepted peer-reviewed papers
  • More under review and in preparation
TVM@AWS

- As a cloud service: Amazon SageMaker Neo
  - Train models once, run anywhere with up to 2x performance improvement

- As a solution
  - Fastest model inference on a number of Amazon EC2 instances
  - Alexa Wakeword model on Amazon Echo
  - Collaborating with a number of external device makers

- As a research project
  - Three accepted peer-reviewed papers
  - More under review and in preparation

- As a compiler
  - AWS Inferentia
AWS@TVM
AWS@TVM

• Join the effort from the very beginning, one of the major contributors
AWS@TVM

• Join the effort from the very beginning, one of the major contributors

• Major features in the past year
  • Frontend: TF object detection model
  • Relay: pass manager, VM, QNN dialect, graph partitioning
  • Optimization: vision-specific ops, conv2d_transpose, sparsity, BERT
  • Runtime: bring your own codegen
AWS@TVM

• Join the effort from the very beginning, one of the major contributors

• Major features in the past year
  • Frontend: TF object detection model
  • Relay: pass manager, VM, QNN dialect, graph partitioning
  • Optimization: vision-specific ops, conv2d_transpose, sparsity, BERT
  • Runtime: bring your own codegen

• Service in the community
  • 2 PMC members, 8 committers, 14 reviewers, and growing
  • Active participation and leadership
Jason Knight

OctoML
Secure and efficient deep learning everywhere
Prediction:
Prediction:

N = number of people building machine learning models
Prediction:

N = number of people building machine learning models

M = number of software developers
Prediction:

N = number of people building machine learning models

M = number of software developers

N >> M
Prediction:

N = number of people building machine learning models

M = number of software developers

$N \gg M$

as $t \to \infty$
Deep learning deployment should be easy. For **everyone**.
Deployment Pain/Complexity

- Model ingestion
- Performance estimation and comparison
- Cartesian product of models, frameworks, and hardware
- Optimization
  - O0, O1, O2
  - Target settings: march, mtune, mcpu
  - Size reductions
  - Quantization, pruning, distillation
- Custom operators (scheduling, cross hardware support)
- Lack of portability / varying coverage across frameworks
- Model integration
  - Output portability
  - Packaging (Android APK, iOS ipa, Python wheel, Maven artifact, etc)
Deep learning deployment should be easy. For *everyone*. 
Deep learning deployment should be easy. For *everyone*.

TVM is core to making that happen.
Deep learning deployment should be easy. For everyone.

TVM is core to making that happen.

... but it’s only the first (important!) step
What are we doing about it?

To make DL deployment easy for everyone:

1. Strengthen the core:
   - Invest in open source TVM for robustness, accessibility, community, and coverage
   - (See next slide)
OctoML investments into TVM

OctoML invests in TVM

Talks today:

Unified IR – Tianqi Chen
Dynamic Execution and Virtual Machine – Jared Roesch and Haichen Shen
uTVM: TVM on bare-metal devices – Logan Weber
TVM at OctoML – Jason Knight

Not presented today:

TVM Transformer Improvements – Josh Fromm
Automatic Quantization – Ziheng Jiang
What are we doing about it?

To make DL deployment easy for everyone:

1. Strengthen the core:
   - Invest in open source TVM for robustness, accessibility, community, and coverage
   - (See next slide)
What are we doing about it?

To make DL deployment easy for everyone:

1. Strengthen the core:
   ○ Invest in open source TVM for robustness, accessibility, community, and coverage
   ○ (See next slide)

2. Build additional stepping stones
   ○ By forming a company! (come see our OctoML talk in the afternoon)
Simple, secure, and efficient deployment of ML models in the edge and the cloud

Drive TVM adoption
Core infrastructure and improvements

Expand the set of users who can deploy ML models:
Services, automation, and integrations

Apache TVM ecosystem

OctoML
Team - The Octonauts

Luis Ceze
Co-founder, CEO
PhD in Computer Architecture and Compilers
Professor at UW-CSE
Venture Partner, Madrona Ventures

Jason Knight
Co-founder, CPO
PhD in Computational Biology and Machine Learning

Tianqi Chen
Co-founder, CTO
PhD in Machine Learning
Professor at CMU-CS

Thierry Moreau
Co-founder, Architect
PhD in Computer Architecture

Jared Roesch
Co-founder, Architect
(soon) PhD in Programming Languages

Andrew McHarg
Ziheng Jiang
Amanda Robles

Login Weber
An Wang
Josh Fromm
Zachary Taback

Jay Bartot
Carlos Guzman
Arvind Krishnamurthy

OctoML
Find out more!

Come to our presentation about the Octomizer this afternoon

- Our first SaaS product for making DL deployment easy
  - Push button AutoTVM optimization
  - Perf comparisons/analysis across models, frameworks, and hardware
  - And more!

https://octoml.ai (mailing list signup)
@octoml on Twitter
Email us! (jknigh@octoml.ai)
Zach Tatlock
Let’s Get in the Wayback Machine
Let’s Get in the Wayback Machine
Challenges for Deep Learning IRs

- State-of-the-art models increasingly depend on:
  - Datatypes - lists, trees, graphs
  - Control flow - branches, loops, recursion
  - Whole-program analyses and optimizations
  - Any one feature “easy to bolt on”
  - Folklore suggests full, expressive IR will be slow

```python
let encode = λ st.
  if(...):
    encode(step(st))
  else:
    ...
```
Challenges for Deep Learning IRs

• State-of-the-art models increasingly depend on:
  • Datatypes - lists, trees, graphs
  • Control flow - branches, loops, recursion
  • Whole-program analyses and optimizations
  • Any one feature “easy to bolt on”
  • Folklore suggests full, expressive IR will be slow

```plaintext
let encode = λ st.
  if(...):
    encode(step(st))
  else:
    ...
```

• Relay generalizes NNVM
• Retains graph-level optimizations
• Provides more expressive features
  • Datatypes, control flow, code re-use
• Functional semantics to simplify analysis
• Automatic differentiation + optimizations

Expr e ::= %l
  | @g
  | const((r | b), s, bt)
  | e((r, ..., r))?((e, ..., e)
  | let %l (: r)? = e; e
  | e; e
  | %graph = e; e
  | fn ((tyParam, ..., tyParam))?
  | (param, ..., param) (→ r)? {e}
  | (e, ..., e)
  | e . n
  | if (e) {e} else {e}
  | match (e) {
  |    | p → e
  |  ... 
  |    | p → e
  | }
  | op
  | ref(e)
  | !e
  | e := e

~ “OCaml for ML”
Relay: Expressiveness + Performance

- High-level Relay models match NNVM in traditional vision inference
Relay: Expressiveness + Performance

- High-level Relay models match NNVM in traditional vision inference
Relay: Expressiveness + Performance

- Low-cost abstraction enabled by:
  - Tensor shape inference and specialization
  - High-level operator fusion
  - Whole-program partial evaluation
Relay: Expressiveness + Performance

- Low-cost abstraction enabled by:
  - Tensor shape inference and specialization
  - High-level operator fusion
  - Whole-program partial evaluation
  - But most of all by extensible, composable optimization framework!
Relay Win: Support for New Models

- High-level Relay models for RNNs and LSTMs can outperform the rest.
Relay Win: Support for New Models

- High-level Relay models for RNNs and LSTMs can outperform the rest

Plus support for new/improved targets via high-level transformations:
Relay Win: Support for New Models

- High-level Relay models for RNNs and LSTMs can outperform the rest

Plus support for new/improved targets via high-level transformations:
New Features

Relay in Production

Relay is an imperative, type-safe programming language designed to be an expressive intermediate representation for machine learning systems. Relay supports algebraic data types, closures, control flow, and recursion, allowing it to directly represent more complex models than computation graph-based IRs (e.g., NNVM) can. In TVM v0.8, Relay is in stable phase and is ready for production.

- Algebraic Data Types (ADT) support (#2564, #3233). ADT provides an expressive, efficient, and safe way to realize recursive computation (e.g., RNN). Refer to https://docs.tvm.ai/langref/relay.adt.html for more information.
- Pass manager for Relay (#2328, #3234, #2191)
- Most frameworks have been supported in Relay, including ONNX, Keras, Tensorflow, Caffe2, CoreML, NNVMv1, MXNet (#2246), Explicitly manifested memory and tensor allocations in Relay (#3360)

Relay Virtual Machine

The Relay Virtual Machine (Relay VM) is the new generation of runtime to strike a balance between performance and flexibility when deploying and executing Relay programs. Previously, the graph runtime is able to utilize the fully static nature of the input graphs to perform aggressive optimization such as fully static allocation, and optimal memory reuse. When we introduce models which make use of control-flow, recursion, dynamic shapes, dynamic allocation we must change how execution works.
Relay + You!

- Relay merged into TVM mainline
- Documentation, tutorials, examples
- Add your own analyses and optimizations
- Target new accelerators
- Support new models
- Tons of community support!

+ many more amazing folks!
Relay + You!

- Relay merged into TVM mainline
- Documentation, tutorials, examples
- Add your own analyses and optimizations
- Target new accelerators
- Support new models
- Tons of community support!

+ many more amazing folks!

[Images of people]
Current Deep Learning Landscape

Frameworks and Inference engines

- TensorFlow
- Keras
- ONNX
- TensorRT

DL Compilers

- nGraph
- GLOW
- MLIR

Kernel Libraries

- CuDNN
- NNPack
- MKL-DNN
  - Hand optimized

Hardware
Current Deep Learning Landscape

Frameworks and Inference engines
- TensorFlow
- Keras
- MxNet
- ONNX Runtime
- TensorRT

DL Compilers
- nGraph
- GLOW
- XNNPACK
- MLIR

Kernel Libraries
- CuDNN
- NNPack
- MKL-DNN

Hand optimized

Hardware

Open source, automated end-to-end optimization framework for deep learning.
Existing Deep Learning Frameworks

Frameworks

Hardware
Existing Deep Learning Frameworks

High-level data flow graph

Frameworks

Hardware

NVIDIA
Existing Deep Learning Frameworks

- FRAMEWORKS:
  - TensorFlow
  - ML
  - PyTorch
  - Keras
  - MxNet

- HIGH-LEVEL DATA FLOW GRAPH

- PRIMITIVE TENSOR OPERATORS SUCH AS Conv2D

- HARDWARE:
  - NVIDIA GPU
Existing Deep Learning Frameworks

High-level data flow graph

Primitive Tensor operators such as Conv2D

eg. cuDNN

Offload to heavily optimized DNN operator library

Frameworks

Hardware
Limitations of Existing Approach

Frameworks

cuDNN

nVIDIA
Limitations of Existing Approach

Frameworks

cuDNN

NVIDIA
Limitations of Existing Approach

cuDNN
Limitations of Existing Approach

cuDNN

Frameworks
Limitations of Existing Approach

cuDNN

Frameworks
Limitations of Existing Approach

New operator introduced by operator fusion optimization. Potential benefit: 1.5x speedup.
Limitations of Existing Approach

New operator introduced by operator fusion optimization potential benefit: 1.5x speedup

cuDNN
Limitations of Existing Approach

New operator introduced by operator fusion optimization potential benefit: 1.5x speedup

cuDNN

NVIDIA
Limitations of Existing Approach

New operator introduced by operator fusion optimization potential benefit: 1.5x speedup

Engineering intensive

cuDNN

Frameworks

NVIDIA
TVM: Learning-based Learning System

High-level data flow graph and optimizations

Hardware
TVM: Learning-based Learning System

Frameworks

High-level data flow graph and optimizations

Hardware
TVM: Learning-based Learning System

High-level data flow graph and optimizations

Machine Learning based Program Optimizer
TVM: Learning-based Learning System

Frameworks

High-level data flow graph and optimizations

Machine Learning based Program Optimizer

Directly generate optimized program for new operator workloads and hardware

Hardware
Why Automation is the Future

Clear winner on emerging models in product

Competitive on benchmarking type model

Quickly enables other optimizations: fusion, layout, parallelization

Portable performance across devices
TVM Stack

High-Level Differentiable IR

Tensor Expression and Optimization Search Space

LLVM, CUDA, Metal

VTA

Optimization

AutoTVM

Device Fleet
Community Highlights

More **Dynamism**

**Tiny** machine learning

Better core **Infra**

More Specialized **Accelerator Support**
Community Highlights

More **Dynamism**

Tiny machine learning

Better core **Infra**

More Specialized **Accelerator Support**
Need for More Dynamism

Model

Data
Need for More Dynamism

Model

Data

static computational graph
Need for More Dynamism

Model

Data

static computational graph

program with loops and recursions
Need for More Dynamism

Model

Data

static computational graph

single tensor with known shape

program with loops and recursions
Need for More Dynamism

Model
- static computational graph
- single tensor with known shape

Data
- program with loops and recursions
- sequence, trees, nested data structure
Relay Virtual Machine

Dynamic shape workloads

More runtime objects: Arrays, Tuples, Trees, ADTs

Minimum runtime for dynamic models

Credit: Jared Roesch, Haichen Shen et.al
Community Highlights

More **Dynamism**

**Tiny** machine learning

Better core **Infra**

More Specialized **Accelerator Support**
Machine Learning is Getting into Tiny Devices

Challenges: limited resources, OS support
uTVM: TVM on bare-metal Devices

Support bare-metal J-TAG devices, **no OS is needed**

ARM Cortex-M
RISC-V

Credit: Logan Weber et al
Community Highlights

More Dynamism

Tiny machine learning

Better core Infra

More Specialized Accelerator Support
Core Infrastructure

New integer simplification and analysis

Unified runtime object protocol
Core Infrastructure

New integer simplification and analysis

Unified runtime object protocol
Core Infrastructure

New integer simplification and analysis

Unified runtime object protocol
Core Infrastructure

New integer simplification and analysis

Unified runtime object protocol

Easy to add new objects (trees, graphs)

Cross language support
Community Highlights

More Dynamism

Tiny machine learning

Better core infra

More Specialized Accelerator Support
Tensorization Challenge for Specialized Accelerators

TPUs

Tensor Compute Primitives

Explicitly Managed Memory Subsystem

Unified Buffer

FIFO

Acc
Tensorization Challenge for Specialized Accelerators

TPUs

Tensor Compute Primitives

Explicitly Managed Memory Subsystem

Unified Buffer

FIFO

Acc
Tensorization Challenge

Compute primitives
Tensorization Challenge

Compute primitives

scalar
Tensorization Challenge

Compute primitives

Scalar

Vector
Tensorization Challenge

Compute primitives

Scalar

Vector

Tensor
Tensorization Challenge

Challenge: Build systems to support emerging tensor instructions
Tensorization Challenge

Computation Specification (Tensor Expression)

\[ C = \text{tvm.compute}((m, n), \lambda y, x: \text{tvm.sum}(A[k, y] \times B[k, x], \text{axis}=k)) \]
Tensorization Challenge

Computation Specification (Tensor Expression)

\[
C = \text{tvm.compute}((m, n), \\
\text{lambda } y, x: \text{tvm.sum}(A[k, y] \times B[k, x], \text{axis}=k))
\]

HW Interface Specification by Tensor Expression

\[
\begin{align*}
A &= \text{tvm.placeholder}((8, 8)) \\
B &= \text{tvm.placeholder}((8,)) \\
k &= \text{tvm.reduce_axis}((0, 8)) \\
C &= \text{tvm.compute}((8, 8), \\
&\quad \text{lambda } y, x: \text{tvm.sum}(A[k, y] \times B[k], \text{axis}=k))
\end{align*}
\]
Tensorization Challenge

Computation Specification (Tensor Expression)

\[ C = \text{tvm.compute}((m, n), \lambda y, x: \text{tvm.sum}(A[k, y] \times B[k, x], \text{axis}=k)) \]

HW Interface Specification by Tensor Expression

A = tvm.placeholder((8, 8))
B = tvm.placeholder((8,))
k = tvm.reduce_axis((0, 8))
C = tvm.compute((8, 8),
                 lambda y, x: tvm.sum(A[k, y] \times B[k], \text{axis}=k))

Tensorization
TVM for TensorCore

Credit: Siyuan Feng
TVM for TensorCore

1.4x better on emerging workloads
Transformer related workloads

Credit: Siyuan Feng
VTA: Open & Flexible Deep Learning Accelerator

Current TVM Stack
VTA Runtime & JIT Compiler
VTA Hardware/Software Interface (ISA)
VTA MicroArchitecture
VTA Simulator

compiler, driver, hardware design, full stack open source

VTA: Open & Flexible Deep Learning Accelerator

- Runtime JIT compile accelerator micro code
- Support heterogenous devices, 10x better than CPU on the same board.
- Move hardware complexity to software
- VTA 2.0 release - Chisel compiler, driver, hardware design full stack open source

TSIM: Support for Future Hardware

Current TVM Stack

New NPU Runtime

TSIM Driver

Credit: Luis Vega, Thierry Moureau
TSIM: Support for Future Hardware

Current TVM Stack

New NPU Runtime

TSIM Driver

TSIM Binary

New Hardware Design in Verilog

Verilator

Credit: Luis Vega, Thierry Moureau
TSIM: Support for Future Hardware

Current TVM Stack

New NPU Runtime

TSIM Driver

TSIM Binary

New Hardware Design in Verilog

Verilator

Credit: Luis Vega, Thierry Moureau
Where are we going: Selected Topics

Unified Runtime

Unified IR

Full-stack Automation
Where are we going: Selected Topics

Unified Runtime

Unified IR

Full-stack Automation
Unified Runtime For Heterogeneous Devices

Device Drivers

NPU Driver

CUDA Driver

External Runtimes

TensorRT

PyTorch
Unified Runtime For Heterogeneous Devices

Device Drivers
- NPU Driver
- CUDA Driver

External Runtimes
- TensorRT
- T

Runtime Module Interface
- `tvm::runtime::Module`
- `GetFunction(string) -> tvm::runtime::PackedFunc`
- `SaveToBinary/LoadFromBinary`
Unified Runtime For Heterogeneous Devices

Device Drivers
- NPU Module (NPU Driver)
- CUDA Module (CUDA Driver)

Runtime Module Interface
- GetFunction(string) -> tvm::runtime::PackedFunc
- SaveToBinary/LoadFromBinary

External Runtimes
- TensorRT
- TensorFlow
Unified Runtime For Heterogeneous Devices

- **Device Drivers**
  - NPU Driver
  - CUDA Driver

- **Subclasses**
  - NPUModule
  - CUDAModule
  - TFModule

- **Runtime Module Interface**
  - GetFunction(string) -> tvm::runtime::PackedFunc
  - SaveToBinary/LoadFromBinary

- **External Runtimes**
  - TensorRT
  - TensorFlow
Unified Runtime Benefit

Unified library packaging
mod.export_library("mylib.so")

Free API (Py/Java/Go)
lib = tvm.module.load("mylib.so")
func = lib["npufunction0"]
func(a, b)

Automatic RPC Support
remote = tvm.rpc.connect(board_url, port)
remote.upload("mylib.so")
remote_mod = remote.load_module(“mylib.so”)
func = remote_mod[“npufunction0”]
func(remote_a, remote_b)
Where are we going: Selected Topics

Unified Runtime

Unified IR

Full-stack Automation
Overview of New IR Infra

Unified module/pas, type system, with function variants support
Compilation Flow under the New Infra

**Import**
- IRModule (relay::Function)

**Lower**
- IRModule (te::Function, ExternFunc, …)

**Codegen**
- runtime::Module

**Optimizations**
- High-level optimizations
- (Auto) Schedules
- Low-level optimizations
Mixed Function Variants in the Same Module

```python
def @relay_add_one(%x : Tensor((10,), f32)) {
    call_destination_passing @te_add_one(%x, out=%b)
}

def @te_add_one(%a: NDArray, %b: NDArray) {
    var %n
    %A = decl_buffer(shape=[%n], src=%a)
    %B = decl_buffer(shape=[%n], src=%b)
    for %i = 0 to 10 [data_par] {
        %B[%i] = %A[%i] + 1.0
    }
}
```
@tvm.hybrid
def te_add_one(a, b):
    n = var("n")
    A = bind_buffer(shape=[n], a)
    B = bind_buffer(shape=[n], b)
    for i in iter_range(n, iter_type="data_par"):
        A[i] = B[i] + 1

mod = tvm.IRModule([te_add_one])
print(mod["te_add_one"].args)
@tvm.hybrid
def te_add_one(a, b):
    n = var("n")
    A = bind_buffer(shape=[n], a)
    B = bind_buffer(shape=[n], b)
    for i in iter_range(n, iter_type="data_par"):
        A[i] = B[i] + 1

mod = tvm.IRModule([te_add_one])
print(mod["te_add_one"].args)
@tvm.hybrid
def te_add_one(a, b):
    n = var("n")
    A = bind_buffer(shape=[n], a)
    B = bind_buffer(shape=[n], b)
    for i in iterable_range(n, iter_type="data_par"):
        A[i] = B[i] + 1

mod = tvm.IRModule([te_add_one])
print(mod["te_add_one"].args)
First-class Python Support

```python
@tvm.hybrid
def te_add_one(a, b):
    n = var("n")
    A = bind_buffer(shape=[n], a)
    B = bind_buffer(shape=[n], b)
    for i in iter_range(n, iter_type="data_par"):  
        A[i] = B[i] + 1

mod = tvm.IRModule([te_add_one])
print(mod["te_add_one"].args)
```

Use hybrid script as an alternative text format

Directly write pass, manipulate IR structures

Accelerate innovation, e.g. use (GA/RL/BayesOpt/your favorite ML method) for AutoSchedule

Easy shift to C++ when product ready
Rethink Low-level Tensor IR

IRModule (relay::Function)

IRModule (te::Function, ExternFunc, …)

runtime::Module
Rethink Low-level Tensor IR

IRModule (relay::Function)

IRModule (te::Function, ExternFunc, …)

runtime::Module
Rethink Low-level Tensor IR

- IRModule (relay::Function)
- IRModule (te::Function, ExternFunc, ...)
- runtime::Module

Function as unit of transformation
Schedule transformation as pass
Better tensorization support
Interpolate with Other ML Compiler Infra

TorchScript
MLIR-TF Function

relay::Function

IR Translation

IRModule
ExternFunc
Function in Other IR
te::Function

Custom Packaging

runtime::Module
ExternModule
DSOModule

Custom codegen
Where are we going: Selected Topics

Unified Runtime

Unified IR

Full-stack Automation
Full Stack Automation

High-Level Differentiable IR

Tensor Expression and Optimization Search Space

LLVM, CUDA, Metal

VTA

Edge FPGA

Cloud FPGA

ASIC

AutoTVM
Full Stack Automation

High-Level Differentiable IR

Tensor Expression and Optimization Search Space

LLVM, CUDA, Metal

VTA

Edge FPGA

Cloud FPGA

ASIC
Full Stack Automation

High-Level Differentiable IR

Tensor Expression and Optimization Search Space

LLVM, CUDA, Metal

VTA

AutoTVM across all layers of the stack
2020 Projected Timeline: Selected Topics

- **Jan**: Unified IR Runtime RFC
- **April**: Unified IR Refactoring
- **July**: First Release with New IR Infra
- **Oct**: Full Stack Automation
- **Jan**: Unified Runtime First Version
- **April**: Documentation Benchmarking
2020 Projected Timeline: Selected Topics

Non comprehensive list of on-going topics

Unified IR Refactoring
Unified Runtime First Version
Documentation Benchmarking
Full Stack Automation

Unified IR Runtime RFC
First Release with New IR Infra
Jan April July Oct
2020 Projected Timeline: Selected Topics

Non comprehensive list of on-going topics

- Ultra Low bits
- Gradient/Training
- BERT
- TSIM
- AutoSchedule
- uTVM Standalone
- Dynamic Shape
- NPU coverage
- Unified IR Refactoring
- First Release with New IR Infra
- Unified IR
- Unified Runtime First Version
- Documentation Benchmarking
- Full Stack Automation

Jan | April | July | Oct
Community
Open Source Community

Incubated as Apache TVM. Independent governance, allowing competitors to collaborate.
Open Source Community

Incubated as Apache TVM. Independent governance, allowing competitors to collaborate.

Open Source Code
Open Development
Open Governance
Open Source Community

Incubated as Apache TVM. Independent governance, allowing competitors to collaborate.
Open Source Community

Incubated as Apache TVM. Independent governance, allowing competitors to collaborate.

Growing Developer Community
22 committers, 47 reviewers, 295 contributors
Open Source Community

Incubated as Apache TVM. Independent governance, allowing competitors to collaborate.

Growing Developer Community
22 committers, 47 reviewers, 295 contributors

~70% growth since TVM Conf 2018
Open Source Community

Incubated as Apache TVM. Independent governance, allowing competitors to collaborate.

Growing Developer Community
22 committers, 47 reviewers, 295 contributors

~70% growth since TVM Conf 2018

Monthly Statistics
~50 authors, ~140 PRs, ~1000 discuss forum posts
Big THANKS to our sponsors!
9:00  Keynote & Community Update
      TVM @ AWS, FB

11:10  Break

11:30  Compilers and VMs

12:20  Boxed lunches - Contributors Meetup

13:10  Lightning talks

13:40  Hardware
      TVM @ Microsoft, ARM, Xilinx

15:10  Break

15:30  Automation, new Hardware

16:50  Break

17:00  Lightning talks

18:10  Social (food, drinks)

20:00  adjourn