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

180+ ppl!
Machine learning is amazing...
Machine learning is amazing…

super human accuracy,
self driving cars, automated scientific discoveries…
Machine learning is amazing…

super human accuracy, self driving cars, automated scientific discoveries…

wow!
Software era:

1. Problem to solve
2. Write code
3. Run on fast machine
Software era:

Problem to solve → Write code → Run on fast machine
Software era:
- Problem to solve
- Write code
- Run on fast machine

Machine learning era:
- Problem to solve
- Data + model templates
- Train on fastest machine
- Inference on fast & cheap machine
Problem to solve → Write code → Run on fast machine

Software era:

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

Machine learning era:

Model size and compute cost growing fast

by Eugenio Culurciello
Software era:

- Problem to solve
- Write code
- Run on fast machine

Machine learning era:

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

Model size and compute cost growing fast

Training costs growing exponentially

by Eugenio Culurciello

by Open AI
42 Years of Microprocessor Trend Data

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

Year


Popularity and computational cost of ML. Oops.
Fundamental trade-off between specialization and performance/efficiency.
Popularity and computational cost of ML. Oops.

Fundamental trade-off between specialization and performance/efficiency.
Fundamental trade-off between specialization and performance/efficiency.

Machine learning algorithms are relatively simple to implement in HW… great!
Machine learning algorithms are relatively simple to implement in HW… great!

Fundamental trade-off between specialization and performance/efficiency.

- Popularity and computational cost of ML. Oops.

Machine Learning Makes Computer Architecture Cool Again!

Models:

- CNN
- GAN
- RNN
- MLP
- DQNN

+~50 startups
Models:

- CNN
- GAN
- RNN
- MLP
- DQNN

Frameworks:

- TensorFlow
- PyTorch
- RoBERTa
- Keras
- MXNet

Companies and Hardware:

- AMD
- Intel
- NVIDIA
- Xilinx
- Microsoft
- Qualcomm
- Amazon
- Google
- Huawei
- +~50 startups
<table>
<thead>
<tr>
<th>Models:</th>
<th>CNN</th>
<th>GAN</th>
<th>RNN</th>
<th>MLP</th>
<th>DQNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frameworks:</td>
<td><img src="https://www.tensorflow.org/" alt="TensorFlow" /></td>
<td><img src="https://pytorch.org/" alt="PyTorch" /></td>
<td><img src="https://www.tensorboard.org/" alt="TensorBoard" /></td>
<td><img src="https://keras.io/" alt="Keras" /></td>
<td><img src="https://www.tensorboard.org/" alt="TensorBoard" /></td>
</tr>
</tbody>
</table>

**Challenge:** Efficiently deploying deep learning everywhere

![Image of various hardware and software logos](image)

+~50 startups
Gaurav Kapoor, Core Machine Learning
HW+SW optimization is key for efficiency
HW+SW optimization is key for efficiency

Lots of hand-tuning, full automation would be a holy grail
Academic group focused on Systems + Computer Architecture + Machine Learning + Programming Languages
Academic group focused on Systems + Computer Architecture + Machine Learning + Programming Languages
Academic group focused on **Systems + Computer Architecture + Machine Learning + Programming Languages**
PL: High-level support for future ML applications

Compilers: Extensible support for future models, optimizations and hardware architectures

Systems: On-device and cloud-based training, distributed systems for ML

Computer Architecture: Extensible, energy efficient hardware designs for inference and training
Computer Architecture: Extensible, energy efficient hardware designs for inference and training

Compilers: Extensible support for future models, optimizations and hardware architectures

PL: High-level support for future ML applications

Systems: On-device and cloud-based training, distributed systems for ML

ML for Systems: Automatic Learning-Based Design and Optimizations

Computer Architecture: Extensible, energy efficient hardware designs for inference and training
ML for Systems: Automatic Learning-Based Design and Optimizations

ML for better ML systems!

PL: High-level support for future ML applications

Compilers: Extensible support for future models, optimizations and hardware architectures

Systems: On-device and cloud-based training, distributed systems for ML

Computer Architecture: Extensible, energy efficient hardware designs for inference and training
ML for Systems: Automatic Learning-Based Design and Optimizations

ML for better ML systems!

PL: High-level support for future ML applications

Compilers: Extensible support for future models, optimizations and hardware architectures

Systems: On-device and cloud-based training, distributed systems for ML

Computer Architecture: Extensible, energy efficient hardware designs for inference and training
Open Source Deployment

- **ML for Systems:** Automatic Learning-Based Design and Optimizations
- **PL:** High-level support for future ML applications
- **Compilers:** Extensible support for future models, optimizations and hardware architectures
- **Systems:** On-device and cloud-based training, distributed systems for ML
- **Computer Architecture:** Extensible, energy efficient hardware designs for inference and training

Open source when ready

Provides infrastructure
Open source compilers have transformed our industry
Open source compilers have transformed our industry

First major open source compiler collection

LLVM: Higher-Level IR, new optimizations, easier extensibility
Open source compilers have transformed our industry

In the age of domain-specialized systems...

 GCC
First major open source compiler collection

 LLVM: Higher-Level IR, new optimizations, easier extensibility
Open source compilers have transformed our industry

In the age of domain-specialized systems...

Specialized compiler stack
for Deep Learning

GCC
First major open source compiler collection

LLVM: Higher-Level IR, new optimizations, easier extensibility

tvm
End the tyranny of closed deep learning systems!
High-Level Differentiable IR

Tensor Expression IR

LLVM, CUDA, Metal
import tvm
from tvm import relay

graph, params =
    frontend.from_keras(keras_resnet50)
graph, lib, params =
    relay.build(graph, target)

Compile
```python
import tvm
from tvm import relay

graph, params =
    frontend.from_keras(keras_resnet50)
graph, lib, params =
    relay.build(graph, target)

Compile
```
import tvm
from tvm import relay

graph, params = frontend.from_keras(keras_resnet50)
graph, lib, params = relay.build(graph, target)

module = runtime.create(graph, lib, tvm.gpu(0))
module.set_input(**params)
module.run(data=data_array)
output = tvm.nd.empty(out_shape, ctx=tvm.gpu(0))
module.get_output(0, output)

compile

Deployable Module

input

deploy

tabby, tabby cat

tabby, tabby cat
import tvm
from tvm import relay

graph, params = 
    frontend.from_keras(keras_resnet50)

graph, lib, params = 
    relay.build(graph, target)

Compile

Deploy

module = runtime.create(graph, lib, tvm.gpu(0))
module.set_input(**params)
module.run(data=data_array)
output = tvm.nd.empty(out_shape, ctx=tvm.gpu(0))
module.get_output(0, output)

On languages and platforms you choose
Automated by Machine Learning

High-Level Differentiable IR

Tensor Expression IR

LLVM, CUDA, Metal

VTA

Edge FPGA

Cloud FPGA

ASIC
Automated by Machine Learning

**High-Level Differentiable IR**

**Tensor Expression IR**

**LLVM, CUDA, Metal**

**VTA**

**Optimization**

- AutoTVM
- AutoVTA

**Hardware Fleet**

- Edge FPGA
- Cloud FPGA
- ASIC
Automated by Machine Learning

High-Level Differentiable IR

Tensor Expression IR

LLVM, CUDA, Metal

VTA

Optimization

AutoTVM

AutoVTA

Hardware Fleet

Edge FPGA

Cloud FPGA

ASIC
Diverse Hardware backends

Optimization

High-Level Differentiable IR

Tensor Expression IR

AutoTVM
Diverse Hardware backends

High-Level Differentiable IR

Tensor Expression IR

Optimization

AutoTVM

LLVM

ARM

x86

AMDGPU

NVPTX

Javascript

WASM
Diverse Hardware backends

- ARM
- x86
- AMDGPU
- NVPTX
- Javascript
- WASM
- CUDA
- Vulkan
- Metal
- C

High-Level Differentiable IR
Tensor Expression IR

Optimization
AutoTVM
Diverse Hardware backends

High-Level Differentiable IR

Tensor Expression IR

Optimization

AutoTVM

LLVM

ARM  x86  AMDGPU  CUDA  Vulkan

NVPTX  Javascript  WASM  Metal  C

VTA
TVM Open Source Community

<table>
<thead>
<tr>
<th>Code</th>
<th>Issues</th>
<th>Pull requests</th>
<th>Projects</th>
<th>Wiki</th>
<th>Insights</th>
<th>Settings</th>
</tr>
</thead>
</table>

Open deep learning compiler stack for cpu, gpu and specialized accelerators [https://tvm.ai](https://tvm.ai)

- compiler
- tensor
- deep-learning
- dsl
- gpu
- opencl
- metal
- performance
- javascript
- rocm
- tvm
- vulkan
- spirv

- 1,968 commits
- 1 branch
- 4 releases
- 166 contributors
- Apache-2.0
Apache governance model: grant project ownership by merit.
11 committers, 29 reviewers, 166 contributors.
Contributed by the community, for the community.
Industrial Impact
TVM + AWS

Vin Sharma, Amazon SageMaker Neo

Amazon: vinarm@ | Twitter: ciphr@
How is AWS using TVM?
How is AWS using TVM?

• As a back-end for Apache MXNet
  • To deploy easily onto edge devices
  • To improve performance on target hardware
How is AWS using TVM?

• As a back-end for Apache MXNet
  • To deploy easily onto edge devices
  • To improve performance on target hardware

• As an optimizer for Amazon AI services
  • Amazon Rekognition: To improve end-to-end latency
  • Amazon Alexa: To increase resource efficiency on Echo/Dot
How is AWS using TVM?

• As a back-end for Apache MXNet
  • To deploy easily onto edge devices
  • To improve performance on target hardware

• As an optimizer for Amazon AI services
  • Amazon Rekognition: To improve end-to-end latency
  • Amazon Alexa: To increase resource efficiency on Echo/Dot

• In a tool chain for Amazon Inferentia
How is AWS using TVM?

• As a back-end for Apache MXNet
  • To deploy easily onto edge devices
  • To improve performance on target hardware

• As an optimizer for Amazon AI services
  • Amazon Rekognition: To improve end-to-end latency
  • Amazon Alexa: To increase resource efficiency on Echo/Dot

• In a tool chain for Amazon Inferentia
How is AWS enabling adoption of TVM?

In a new service called Amazon SageMaker Neo.
How is AWS enabling adoption of TVM?

In a new service called **Amazon SageMaker Neo**, you can now train and serve TVM models:

- **Model input files**: MXNet: .json & .params
- **Name and shape of input node**: "data":[1,3,227,277]
- **Framework**:
- **Output Location**: Cloud Instance Type | Edge Device
How is AWS enabling adoption of TVM?

In a new service called **Amazon SageMaker Neo**

- **Model input files:** MXNet: .json & .params
- **Name and shape of input node:** "data":[1,3,227,277]
- **Framework:**
- **Output Location:**
- **Target Platform:**
  - Cloud Instance Type | Edge Device

We're Hiring!
How is AWS contributing to TVM?

Releasing all TVM modifications and enhancements in Neo to open source

- **Frameworks:** TensorFlow, MXNet, PyTorch, ONNX
- **Models:** ResNet, VGG, Inception, MobileNet, DenseNet, SqueezeNet
- **Operators:** Several new ops in NNVM/TVM
- **Optimizations:** Node Annotation, Graph Partitioning, Ring Buffer, NHWC, Graph Tuning
- **Acceleration Library:** Nvidia TensorRT
- **Hardware:** Cross-Compilation to ARM, Intel, Nvidia; More Coming Soon
How is AWS contributing to TVM?

Releasing all TVM modifications and enhancements in Neo to open source

- **Frameworks**: TensorFlow, MXNet, PyTorch, ONNX
- **Models**: ResNet, VGG, Inception, MobileNet, DenseNet, SqueezeNet
- **Operators**: Several new ops in NNVM/TVM
- **Optimizations**: Node Annotation, Graph Partitioning, Ring Buffer, NHWC, Graph Tuning
- **Acceleration Library**: Nvidia TensorRT
- **Hardware**: Cross-Compilation to ARM, Intel, Nvidia; More Coming Soon
### TVM on Huawei’s AI portfolio

<table>
<thead>
<tr>
<th>AI Applications</th>
<th>Application Enablement</th>
</tr>
</thead>
<tbody>
<tr>
<td>HiAI Service</td>
<td>General APIs1</td>
</tr>
<tr>
<td>HiAI Engine</td>
<td>Advanced APIs</td>
</tr>
<tr>
<td>MindSpore</td>
<td>Pre-integrated Solutions</td>
</tr>
<tr>
<td>ModelArts</td>
<td></td>
</tr>
</tbody>
</table>

#### Appication enabling:
- Full-pipeline services (ModelArts), hierarchical APIs, and pre-integrated solutions

#### MindSpore:
- Unified training and inference framework for device, edge, and cloud (both standalone and cooperative)

#### CANN:
- Chip operators library and highly automated operators development toolkit

#### Ascend:
- AI chip series based on unified scalable architecture

<table>
<thead>
<tr>
<th>CCE lib/extensions</th>
<th>Tensor Engine / TVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ascend-Nano</td>
<td>Ascend-Tiny</td>
</tr>
<tr>
<td>Ascend-Lite</td>
<td>Ascend-Mini</td>
</tr>
<tr>
<td>Ascend</td>
<td>Ascend-Max</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Framework</th>
<th>Chip Enabler</th>
</tr>
</thead>
<tbody>
<tr>
<td>TensorFlow</td>
<td>MindSpore</td>
</tr>
<tr>
<td>PyTorch</td>
<td></td>
</tr>
<tr>
<td>PaddlePaddle</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Consumer Device</th>
<th>Public Cloud</th>
<th>Private Cloud</th>
<th>Edge Computing</th>
<th>Industrial IoT Device</th>
</tr>
</thead>
</table>
During model conversion we use TE/TVM to customize operators for completeness and performance.

70+ operators are written by TVM, bring us ~3x development efficiency improvement.
Successful Practice with Audi in Level 4 Autonomous Driving

~ A Complete City Commute Record ~

Driving in the evening

High-speed cruise

Traffic Jam Pilot (TJP)

Traffic light identification

Pedestrian identification

Automatic parking

Joint developed autonomous driving algorithm gains leading scores in industry authoritative KITTI 2D/3D/BEV tests!
TVM is working on Atlas series product

Atlas 200 Developer Kit
- 16 TOPS INT8@24 W
- 1 USB type-C, 2 CCM interfaces, 1 GE network port, 1 SD card slot
- 8 GB memory

Atlas 300 AI Accelerator Card
- 64 TOPS INT8@75 W
- 64-channel HD video real-time analysis and JPEG decoding
- 32 GB memory, 204.8 GB/s memory bandwidth
- PCIe 3.0 x16, half-height half-length card

Atlas 500 AI Edge Station
- Capable of processing 16-channel HD videos in the size of a set-top-box (STB)
- Delivers 4x higher performance over counterparts

Atlas 800 AI Appliance
- Provides optimized AI environment based on the standard framework and programming environment
- Leverages high-performance GPU scheduling algorithms, improving resource utilization by over 15%

Smart Manufacturing
(intelligent quality inspection and flexible manufacturing)

Intelligent Care
(kindergarten and elderly care)

Smart Transportation
(traffic light tuning, intelligent traffic guiding)
Huawei’s Contributions on TVM

8 Contributors:
kun-zh, sgrechanik-h, libing4752, derisavi-huawei, solin319, ehsanmok, gaoxiong-1, jiacunjiang1215

4 Reviewers:
Srkreddy1238, PariksheetPinjari909, siju-Samuel, Xqdan

We are working on:
1. Huawei Ascend ASIC support.
2. Front end to support Darknet, ONNX.
3. Optimization on Auto-TVM, IR extensions.
4. Tensorize, cache read/write, access_ptr API.

In the future we will try to:
1. Codegen for fused operators.
2. NLP support.
4. Training Operators.
VGG11 on Raspberry Pi 3B

TensorflowLite
32bit fp
66% top-1 ImageNet accuracy
1.42 fps
VGG11 on Raspberry Pi 3B

- TensorflowLite
  - 32bit fp
  - 66% top-1 ImageNet accuracy
  - 1.42 fps

- Trained binarized model
- Operators implemented with TVM
VGG11 on Raspberry Pi 3B

TensorflowLite
- 32bit fp
- 66% top-1 ImageNet accuracy
- 1.42 fps

TVM
- 2-bit activation 1-bit weight
- 62% top-1 ImageNet accuracy
- 4.67 fps
Further down the stack...
Thierry Moreau
Open Source Stack Overview

High-Level Differentiable IR

Tensor Expression IR

VTA Runtime & JIT Compiler

VTA Hardware/Software Interface (ISA)

VTA MicroArchitecture  VTA Simulator

Versatile Tensor Accelerator Stack (VTA)
Open Source Stack Overview

VTA Backends

- **Simulator**: out-of-the-box testing to write compiler passes

High-Level Differentiable IR

Tensor Expression IR

VTA Runtime & JIT Compiler

VTA Hardware/Software Interface (ISA)

VTA MicroArchitecture

VTA Simulator

Versatile Tensor Accelerator Stack (VTA)
Open Source Stack Overview

VTA Backends

- **Simulator**: out-of-the-box testing to write compiler passes
- **FPGA**: fast design iteration, quick deployment, flexibility

High-Level Differentiable IR

Tensor Expression IR

VTA Runtime & JIT Compiler

VTA Hardware/Software Interface (ISA)

VTA MicroArchitecture

VTA Simulator

Versatile Tensor Accelerator Stack (VTA)
Open Source Stack Overview

VTA Backends

- **Simulator**: out-of-the-box testing to write compiler passes
- **FPGA**: fast design iteration, quick deployment, flexibility
- **ASIC**: industrial-strength efficiency

High-Level Differentiable IR

Tensor Expression IR

VTA Runtime & JIT Compiler

VTA Hardware/Software Interface (ISA)

VTA MicroArchitecture  VTA Simulator

Versatile Tensor Accelerator Stack (VTA)
Hardware Exploration with VTA

HW / SW Constraints

FPGA
- # BRAMs
- DRAM channels
- logic resources

Model
- batch size
- data types
- channel width

Architecture Knobs
- GEMM Intrinsic: e.g. (1,32) x (32,32) vs. (4,16) x (16,16)
- # of units in tensor ALU: e.g. 32 vs. 16
- BRAM allocation between buffers, register file, micro-op cache

Circuit Knobs
- Circuit Pipelining: e.g. for GEMM core between [11, 20] stages
- PLL Frequency Sweeps: e.g. 250 vs. 300 vs. 333MHz

VTA Design Space

VTA Candidate Designs
- #1 Design AAA @ 307GOPs
- #2 Design BBB @ 307GOPs
- #3 Design CCC @ 307GOPs
- #4 Design DDD @ 256GOPs

Needs to pass place & route and pass timing closure

Deliverable
- 307 GOPs
- 256 GOPs

Operator Performance

AutoTuning

Tuned Operator Lib

VTA Design BBB

FPGA

Graph Optimizer

Model

custom
Hardware Exploration with VTA

**HW / SW Constraints**
- FPGA
  - # BRAMs
  - DRAM channels
  - logic resources
- Model
  - batch size
  - data types
  - channel width

**VTA Design Space**

**Architecture Knobs**
- GEMM Intrinsic: e.g. (1,32) x (32,32) vs. (4,16) x (16,16)
- # of units in tensor ALU: e.g. 32 vs. 16
- BRAM allocation between buffers, register file, micro-op cache

**Circuit Knobs**
- Circuit Pipelining: e.g. for GEMM core between [11, 20] stages
- PLL Frequency Sweeps: e.g. 250 vs. 300 vs. 333MHz

**VTA Candidate Designs**
- #1 Design AAA @ 307 GOPs
- #2 Design BBB @ 307 GOPs
- #3 Design CCC @ 307 GOPs
- #4 Design DDD @ 256 GOPs

**Deliverable**
- 307 GOPs
- 256 GOPs

**Operator Performance**
- AutoTuning
  - Tuned Operator Lib
  - VTA Design BBB
- FPGA
  - Graph Optimizer
  - Model
    - custom

1000s
Hardware Exploration with VTA

VTA Design Space

Architecture Knobs
- GEMM Intrinsic: e.g. (1,32) x (32,32) vs. (4,16) x (16,16)
- # of units in tensor ALU: e.g. 32 vs. 16
- BRAM allocation between buffers, register file, micro-op cache

Circuit Knobs
- Circuit Pipelining: e.g. for GEMM core between [11, 20] stages
- PLL Frequency Sweeps: e.g. 250 vs. 300 vs. 333MHz

VTA Candidate Designs

#1 Design AAA @ 307GOPs
#2 Design BBB @ 307GOPs
#3 Design CCC @ 307GOPs
#4 Design DDD @ 256GOPs

Needs to pass place & route and pass timing closure

1000s ~ 10

HW / SW Constraints
- FPGA
  - # BRAMs
  - DRAM channels
  - logic resources
- Model
  - batch size
  - data types
  - channel width

Deliverable
- 307 GOPs
- 256 GOPs

Operator Performance
- AutoTuning
  - Tuned Operator Lib
  - VTA Design BBB

Graph Optimizer
- Model
  - custom
Schedule Exploration with VTA

VTA Candidate Designs

- #1 Design AAA @ 307 GOPs
- #2 Design BBB @ 307 GOPs
- #3 Design CCC @ 307 GOPs
- #4 Design DDD @ 256 GOPs

Needs to pass place & route and pass timing closure
Schedule Exploration with VTA

VTA Candidate Designs

#1 Design AAA @ 307GOPs
#2 Design BBB @ 307GOPs
#3 Design CCC @ 307GOPs
#4 Design DDD @ 256GOPs

Operator Performance AutoTuning

Needs to pass place & route and pass timing closure
Schedule Exploration with VTA

VTA Candidate Designs

- #1 Design AAA @ 307 GOPs
- #2 Design BBB @ 307 GOPs
- #3 Design CCC @ 307 GOPs
- #4 Design DDD @ 256 GOPs

Needs to pass place & route and pass timing closure

Operator Performance AutoTuning

Deliverable

- Model
- Graph Optimizer
- Tuned Operator Lib
- VTA Design BBB

FPGA

Tuned Operator Lib

VTA Design BBB

FPGA
TVM+VTA Stack Goals
TVM+VTA Stack Goals

- Blue-print for a complete deep learning acceleration stack
TVM+VTA Stack Goals

• Blue-print for a complete deep learning acceleration stack

• Experimentation framework for cross-stack deep learning optimizations
TVM+VTA Stack Goals

• Blue-print for a complete deep learning acceleration stack

• Experimentation framework for cross-stack deep learning optimizations

• Open-source community for industrial-strength deep learning acceleration
Training Deep Learning Models with TVM
Jared Roesch
Model

High-Level Differentiable IR
Tensor Expression IR

LLVM, CUDA, Metal | VTA

Edge FPGA | Cloud FPGA | ASIC

Standalone inference deployment
Model

High-Level Differentiable IR

Tensor Expression IR

LLVM, CUDA, Metal

VTA

Edge FPGA

Cloud FPGA

ASIC

Standalone inference deployment
Model

High-Level Differentiable IR

Automatic Differentiation

Gradient Program for Training

Tensor Expression IR

LLVM, CUDA, Metal

VTA

Edge FPGA

Cloud FPGA

ASIC
Model

High-Level Differentiable IR

Automatic Differentiation

Gradient Program for Training

Tensor Expression IR

LLVM, CUDA, Metal

VTA

Edge FPGA
Cloud FPGA
ASIC

Standalone training deployment
Automatic Differentiation

- Automatic generation of gradient programs
- Support for customized data types and FPGA training
- Support for distributed execution, and integration with technology such as PHub (see Liang’s talk).

More details on the Relay talk later today!
Road ahead...
On the horizon…

Training

Automation

Hardware
On the horizon…

**Training**
- AutoDiff with Relay
- Training on-device
- Tradeoff accuracy/throughput/Joules

**Automation**

**Hardware**
On the horizon…

**Training**
- AutoDiff with Relay
- Training on-device
- Tradeoff accuracy/throughput/Joules

**Automation**
- Auto quantization
- Full-program optimization
- Automated HW design

**Hardware**
On the horizon…

**Training**
- AutoDiff with Relay
- Training on-device
- Tradeoff accuracy/throughput/Joules

**Automation**
- Auto quantization
- Full-program optimization
- Automated HW design

**Hardware**
- VTA Chisel design
- ASIC flow
- Training on VTA
Big THANKS to our sponsors!
<table>
<thead>
<tr>
<th>Time</th>
<th>Session</th>
</tr>
</thead>
<tbody>
<tr>
<td>9:00</td>
<td><strong>Keynote, TVM Overview, TVM @ Amazon</strong></td>
</tr>
<tr>
<td>11:05</td>
<td><strong>Break</strong></td>
</tr>
<tr>
<td>11:25</td>
<td><strong>Automation, HW Specialization, Security</strong></td>
</tr>
<tr>
<td>12:30</td>
<td><strong>Boxed lunches</strong></td>
</tr>
<tr>
<td>13:30</td>
<td><strong>Training, Programming Systems, Hardware</strong></td>
</tr>
<tr>
<td>15:20</td>
<td><strong>Break, contributors meetup</strong></td>
</tr>
<tr>
<td>15:50</td>
<td><strong>Compilers, FPGAs</strong></td>
</tr>
<tr>
<td>16:30</td>
<td><strong>Lightning talks</strong></td>
</tr>
<tr>
<td>17:35</td>
<td><strong>Community formation</strong></td>
</tr>
<tr>
<td>18:10</td>
<td><strong>Social (food, drinks)</strong></td>
</tr>
<tr>
<td>20:00</td>
<td><strong>adjourn</strong></td>
</tr>
</tbody>
</table>