The Past, Present, and Future of Deep Learning Acceleration Stacks

Thierry Moreau, Post Doctorate Researcher at University of Washington

ARM Research Summit, RAAW Workshop, September 16th 2019

sampl

PAUL G. ALLEN SCHOOL OF COMPUTER SCIENCE & ENGINEERING

tvm.ai

nytimes.com

Trump Calls for Renegotiating Deep Learning Abstractions

September 16, 2019 - Kate Conger and Noam Scheiber

The president of the United States has no particular interest in the economics of artificial intelligence (AI) — as far as anyone can tell, at least. In a conversation with journalists on Friday, President Trump said, "I'm not too much of an expert on that," when he was asked whether he was worried about AI displacing humans in jobs. As for AI, the president has given no indication that he is an avid fan of anything related to it. Not surprisingly, no analysts can name any job Trump actually wants to hire AI-savvy scientists for. On Twitter, Trump quoted the old joke "It is said that when three monkeys with typewriters tried to write the Declaration of Independence, none could. So they hired an Irishman with a pipe to write it, and voila!" He then complained that such an Irishman would not be able to use any of his knowledge of AI to draft his new tax plan. (Is it possible this was a reference to actually having formed the Senate tax bill by reading every single page?)

FAKE NEWS

Deep learning is replacing humans at many tasks…

Grover: A State-of-the-Art Defense against Neural Fake News

R. Zellers et al. Defending Against Neural Fake News, arXiv preprint 2019

…and specialization has enabled it

- It takes 2 weeks to train Grover Mega on 256-core TPU v3 slice
  - 13.4 PFLOPs of peak compute throughput…

HIGHLIGHTS: JUNE 2012

- Sequoia, an IBM BlueGene/Q system installed at the National Nuclear Security Administration and part of the Advanced Simulation and Computing Program (ASC) is the No. 1 system on the TOP500. It was first delivered to the Lawrence Livermore National Laboratory in 2011and now full deployed with an impressive 16.32 Petaflop/s on the Linpack benchmark using 1,572,864 cores. Sequoia is one of the most energy efficient systems on the list consuming a total of 7.89.

…throughput that could "rival" the best supercomputer 7 years ago!

Training throughput accessible to a small academic research lab has grown drastically thanks to specialization

	2012		2019
Model	AlexNet		Grover Max
DL Processor	GTX580 GPU	**specialization** →	TPUv3
Compute Throughput	3.2 TFLOPS	**4200x** →	13.4 PFLOPs
Time to Train	1 week	**2x** →	2 weeks
Task	Human-Level Image Classification		Human-Level Generation / Detection of Fake News

Hardware efficiency dictates our ability to realize complex human-level tasks with deep learning

We are in the middle of a golden age of DL Specialization

(credit: http://basicmi.github.io/AI-Chip/)

Tech Giants/System

Google
Microsoft
facebook
aws
Apple
IBM
Alibaba Group 阿里巴巴集团
HUAWEI
Baidu 百度
Tesla
Hewlett Pack Enterprise
FUJITSU
DELL
Western Digital
NOKIA
LG

IC Vender/Fabless

intel
SAMSUNG
NVIDIA
QUALCOMM
AMD
XILINX
MEDIATEK
UNISOC
MARVELL
Rockchip 瑞芯微电子
Ambarella
NationalChip

Automated Driving

NXP
Texas Instruments
RENESAS
TOSHIBA
ST

Startup in China

Cambricon 寒武纪科技
地平线 Horizon Robotics
BITMAIN
intelliFusion 云天励飞
Think Force
NextVPU 肇观电子
Canaan
Enflame
INTENGINE 探境科技
清微智能 TSING MICRO
亿智科技
WITHMEM 忆芯科技

Smart Voice

ChipIntelli
Rokid
AISPEECH 思必驰 专注人性化的智能语音
Unisound 云知声

Startup Worldwide

Cerebras
WAVE COMPUTING
Graphcore
SambaNova SYSTEMS
habana
HAILO
thinci
KALRAY
groq
Tachyum
Esperanto TECHNOLOGIES
PEZY Computing
Eta Compute
GREENWAVES TECHNOLOGIES

FPGA/eFPGA

Achronix
flexlogix

Processing in Memory

MYTHIC
SYNTIANT
AREANNA
gyrfalcon technology

Optical Computing

LIGHTELLIGENCE
LIGHTMATTER
Optalysys

Neuromorphic

aiCTX
brainchip
Preferred Networks
Kneron
INNOGRIT
NeuroBlade
NOVUMIND

More at https://basicmi.github.io/AI-Chip/

IP/Design Sevice

arm
SYNOPSYS
Imagination
CEVA
cadence
SiFive
ARTERIS IP
Moortec

Design service with In-house IP

VeriSilicon
BROADCOM
GUC
alchip
FARADAY
eSilicon

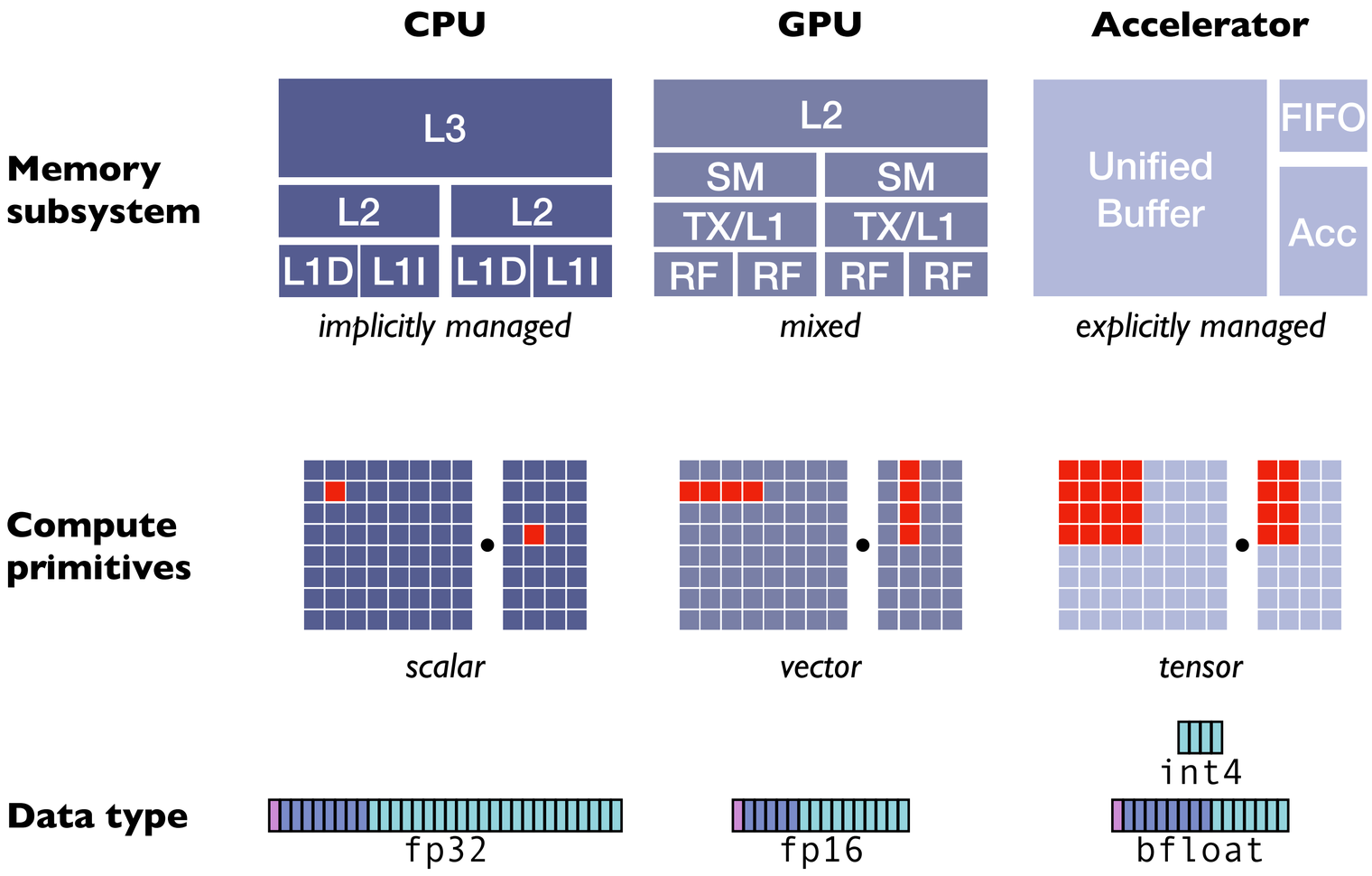
Compilers

TensorFlow XLA
GLOW
tvm
NVIDIA TensorRT
plaidML
nGraph
Tiramisu Compiler
ONNC
The Tensor Algebra Compiler (taco)

Benchmarks

MLPerf
AI - Benchmark
AI Matrix.
AIIA 中国人工智能产业发展联盟 Artificial Intelligence Industry Alliance
DAWNBench
MLMark An EEMBC Benchmark

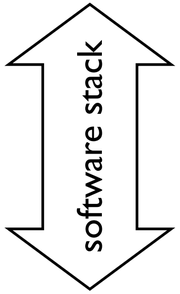
What Makes Accelerators Different from CPUs/GPUs?



Peak FLOPs are only useful if a programmer can access them

ML Engineer

describe and train new models to achieve new capabilities

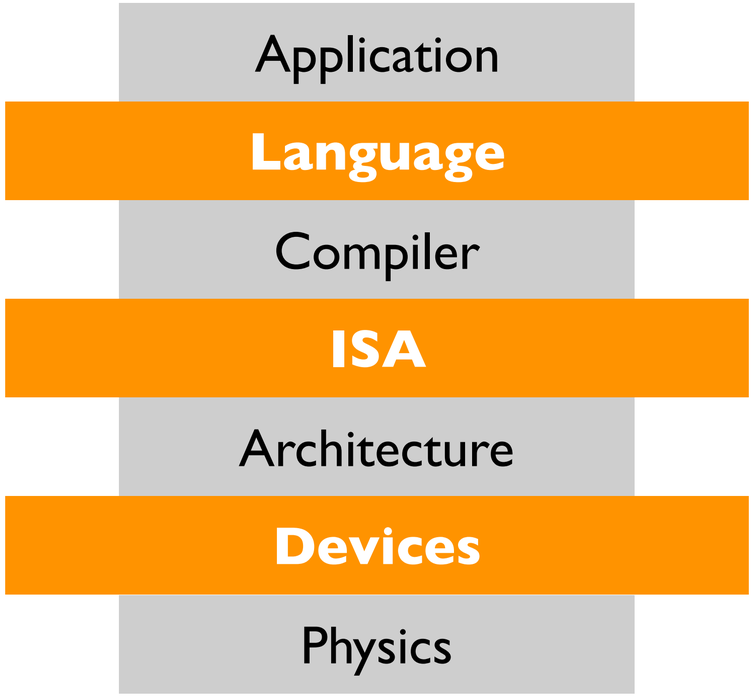


reconcile productivity and efficiency

DL Accelerator Chip

push boundaries on efficiency

Compute stack overview



abstraction layers let us innovate in a mostly siloed fashion

DL compute stack yesterday (2012)

Application

Language

Compiler

ISA

Architecture

Devices

Physics

Alex Krizhevsky

(Mar 2013-Sep 2017) At Google in Mountain View, California.
My email: akrizhevsky@gmail.com.

If you are looking for the CIFAR-10 and CIFAR-100 datasets, click here.

Code (very outdated stuff)

Here's some CUDA/C++ code that I wrote. The descriptions here are rather skimpy, so email me if you need help getting any of it to run.

For GTX 580-class GPUs (compute capability ≥ 2.0):

- Abstract convolutional neural network for CUDA 4.0 (Google code project link) -- A C++/CUDA (with a python front-end) implementation of neural networks using the back-propagation algorithm. Capable of modeling any directed acyclic graph of layers. Supports convolutional layers, pooling layers, fully-connected layers, locally-connected layers, and others.

  Includes a new implementation of convolution in CUDA which is several times faster than the 2D convolution routines posted below. Visit the project page for more information.

- 2D local filter routines for CUDA 3.2 **(Updated Apr 2, 2011)**

  Note: This code has been subsumed by the convolutional neural network code above, which includes a faster version of this code with more features (such as sparse filter-channel connectivity).

  A fast implementation of local, non-convolutional filters. These can be used to build an autoencoder, RBM, etc., with locally-connected, non-shared filters. The stride between filters can be arbitrary, but the catch is that the routines are only efficient if there are multiple filters per image location. That is to say, if a filter is to be applied at location (x,y) on the image, it's better to apply 16 or 32 filters there rather than just one. So instead of applying one filter at (x,y), another at (x+1,y), etc., consider applying 32 at (x,y), 32 at (x+stride,y), for some stride of your choosing.

  The routines support color images with as many channels as you like. This means they'll also work for NORB, if you interpret stereo as two channels. You can also use them to build deep, locally-connected nets, in which the set of filter outputs at a particular image location at layer L can be interpreted as the set of channels in the input image to layer L+1.

  Using this code, here's what an RBM with roughly 10000 11x11x2 local hidden units learned in just 10 minutes on NORB. And here's what another with 9216 11x11 local hidden units learned on the tiny images in *under* 10 minutes. No data preprocessing in either case.

  Now used by AccelerEyes (makers of Jacket, a Matlab GPU framework).

For GTX 280-class GPUs (compute capability 1.3):

Note: this code was written for GTX 280-class GPUs. Parts of it may not work on current-generation GPUs.

Much of the programming burden fell on the ML Engineer

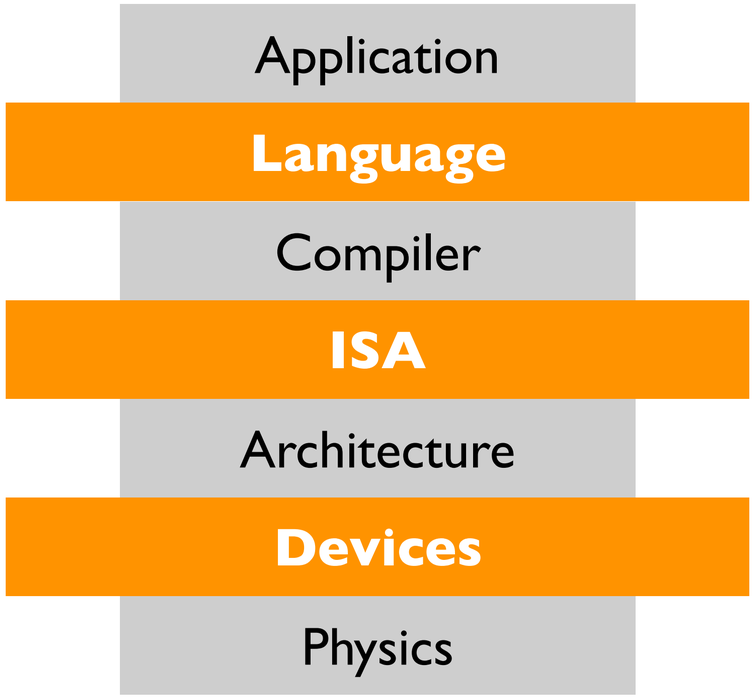
Application	Compiler	Architecture
		
ML Engineer	**Compiler Engineer**	**HW Engineer**
designs model architecture *implements differentiation for training* *manages GPU compute graph* *generates high-performance GPU kernels*	*code generation*	*architecture design*

Compute stack has become more domain specialized and layered

Application	DL Framework	Graph Compiler	Tensor Compiler	Compiler	Architecture
ML Engineer	**Compiler Engineer**	**Compiler Engineer**	**Compiler Engineer**	**Compiler Engineer**	**HW Engineer**
designs model architecture		*manages GPU compute graph*		*code generation*	
	implements differentiation for training		*generates high-performance GPU kernels*		*architecture design*

now the ML Engineer can be more productive and just focus on innovating at the application layer

DL compute stack today



DL compute stack today

Application

DL DSL/Format

DL Framework

Graph IR

Graph Compiler

Tensor IR

Tensor Compiler

Language

Compiler

ISA

Architecture

ONNX
TensorFlowLite
theano
Keras
Caffe
torch
PyTorch
TensorFlow
mxnet
Caffe2
Deep Graph Library

nGraph
NNVM
Relay
GLOW
ONNC
NVIDIA TensorRT

Halide
tvm
Tiramisu Compiler
MLIR
Tensor Comprehensions
DLVM
taco
plaidML

GCC
NVIDIA NVCC
AMD HCC

DL compute stack today

Application

DL DSL/Format

DL Framework

Graph IR

Graph Compiler

Tensor IR

Tensor Compiler

Language

Compiler

ISA

Architecture

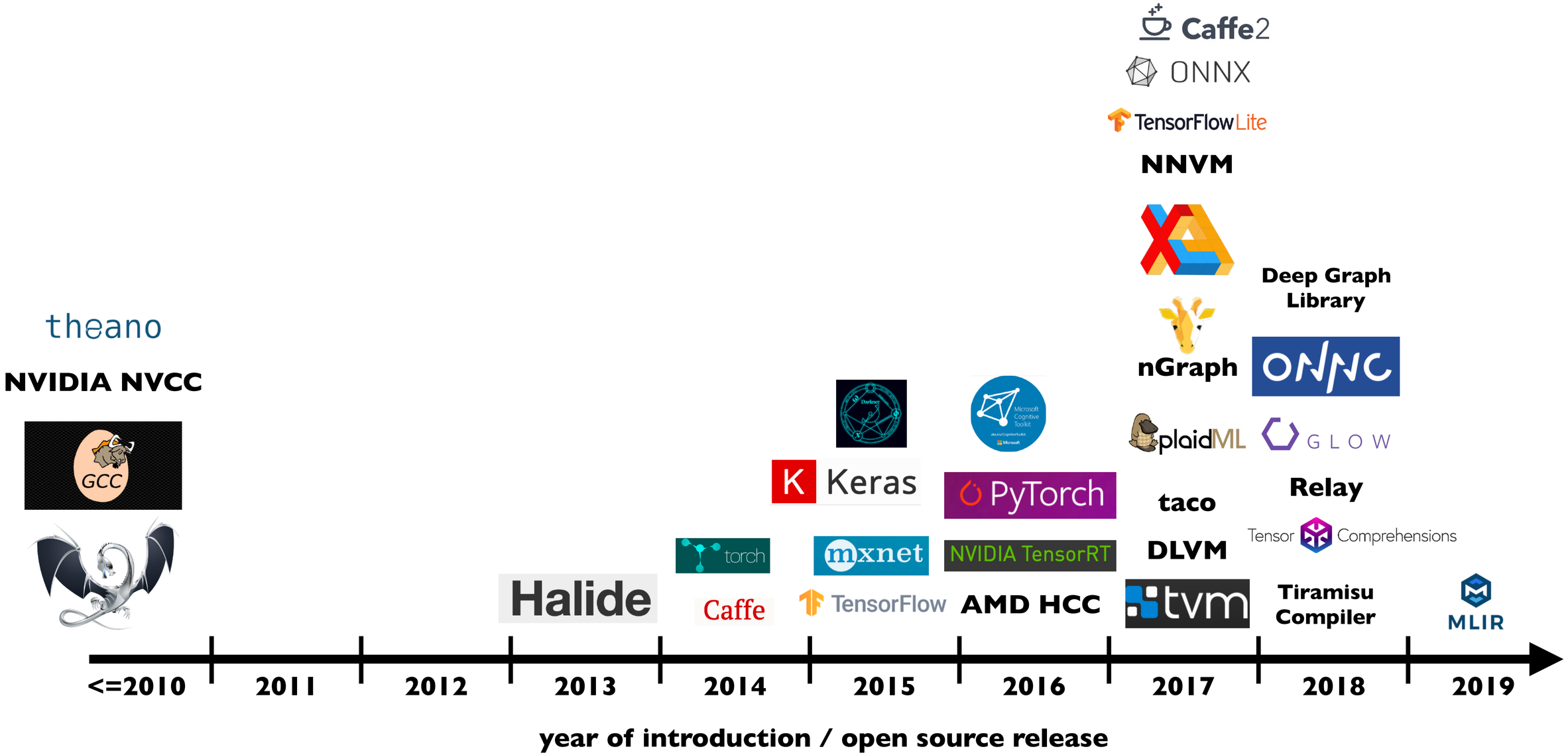
ONNX
TensorFlowLite
theano
Keras
Caffe
torch
PyTorch
TensorFlow
mxnet
Caffe2
Deep Graph Library

nGraph
NNVM
Relay
GLOW
ONNC
NVIDIA TensorRT

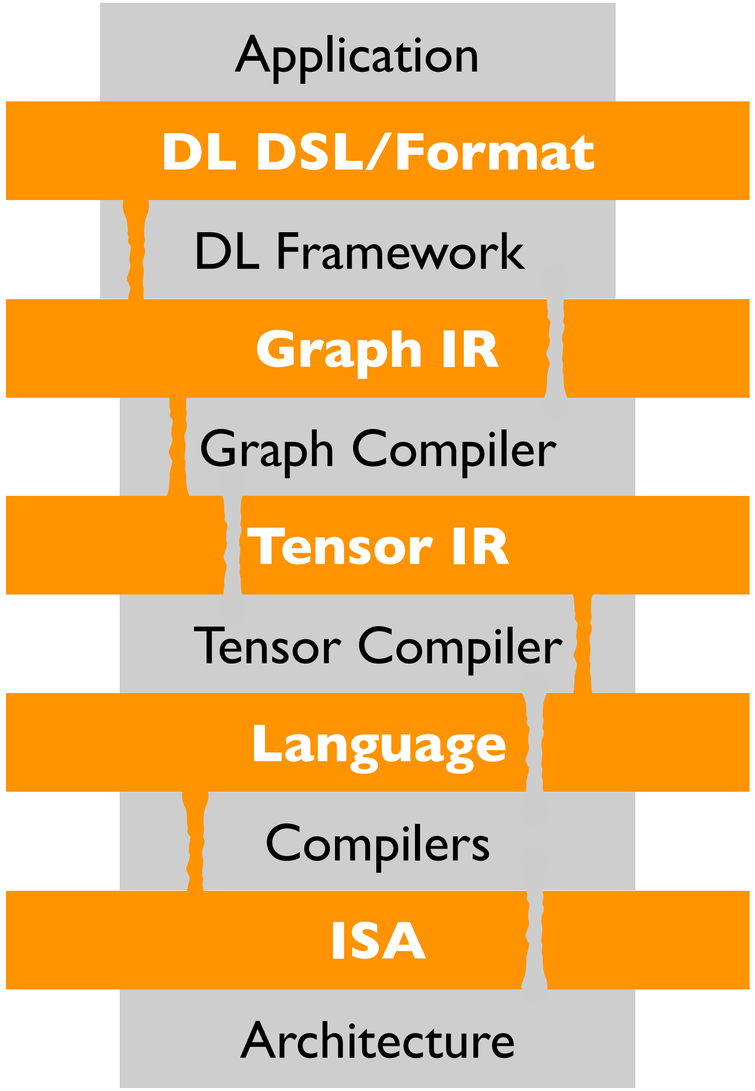
Halide
tvm
Tiramisu Compiler
MLIR
Tensor Comprehensions
DLVM
taco
plaidML

GCC
NVIDIA NVCC
AMD HCC

DL stack has evolved rapidly in a few years…



DL compute stack today

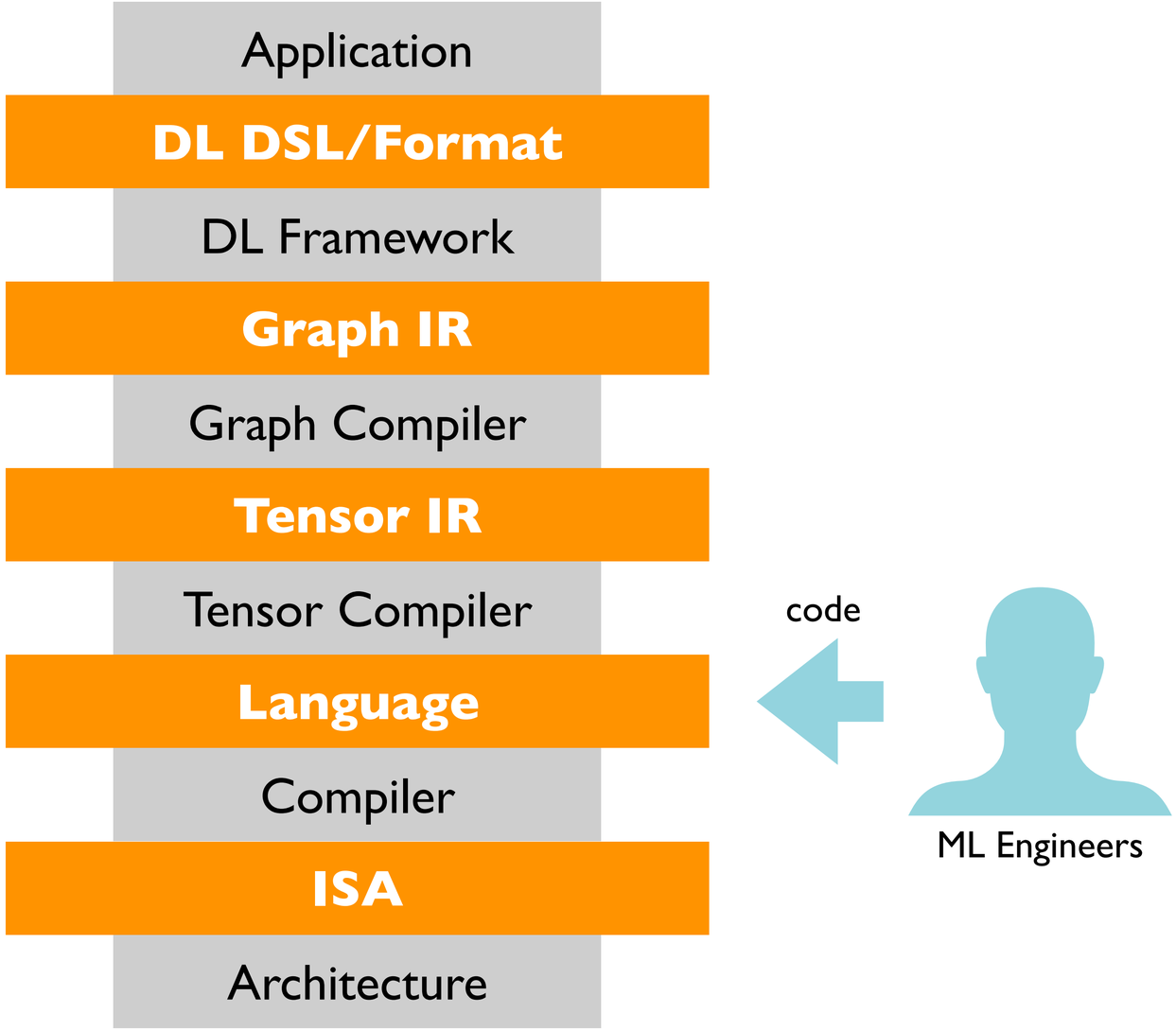


1) Abstractions are more malleable and permeable

- "Wild west of DL accelerators"

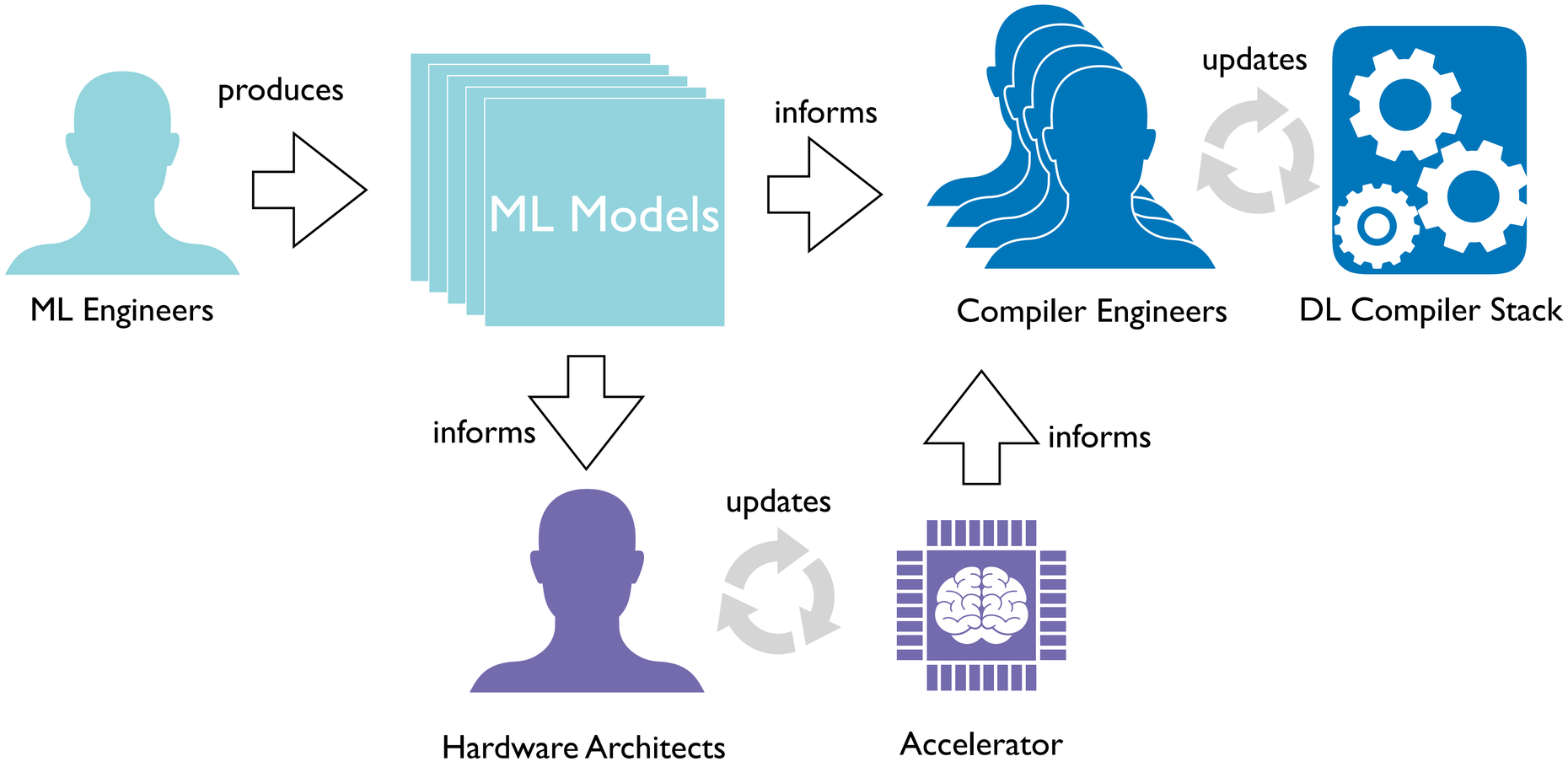
2) Compiler layers are more vertically integrated

DL compute stack today (recap)



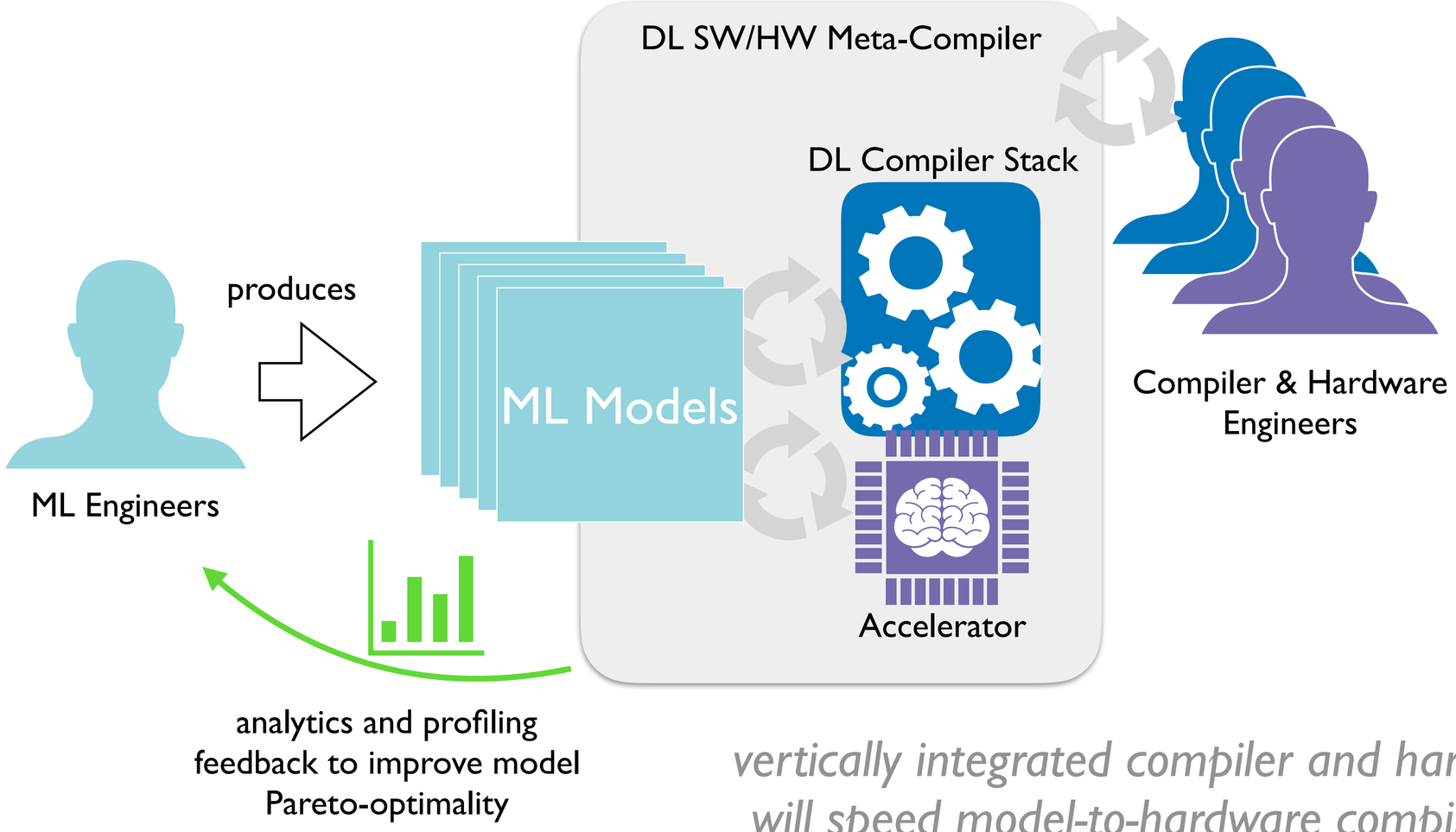
1) raised programming abstraction level
2) ease of model & hyper parameter exploration
3) more performance portability, scalability

Model to hardware flow today



this separation of applications/compilers/hardware gets us from model to efficient hardware execution in months

Novel model to hardware flow tomorrow



vertically integrated compiler and hardware will speed model-to-hardware compilation

TVM: an open source deep learning system stack for diverse hardware (see tvm.ai)

Relay: High-Level Differentiable IR

TVM: Tensor Expression IR

LLVM | CUDA | Metal

VTA Runtime & JIT Compiler

VTA Meta-ISA

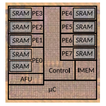
VTA Meta-Architecture

ARM/x86 CPU

GPU

iOS

FPGA



ASIC

Model translation

Quantization

Graph-rewriting

Layout re-write

Operator fusion

Operator lowering

Tiling

Virtual threading

Tensorization

Hardware variant exploration with TVM/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 @ 307GOPs
- #2 Design BBB @ 307GOPs
- #3 Design CCC @ 307GOPs
- #4 Design DDD @ 256GOPs

Needs to pass place & route and pass timing closure

Moreau et al., IEEE Micro S.I. 2019

Schedule exploration with TVM/VTA

VTA Candidate Designs

#1 Design AAA @ 307GOPs

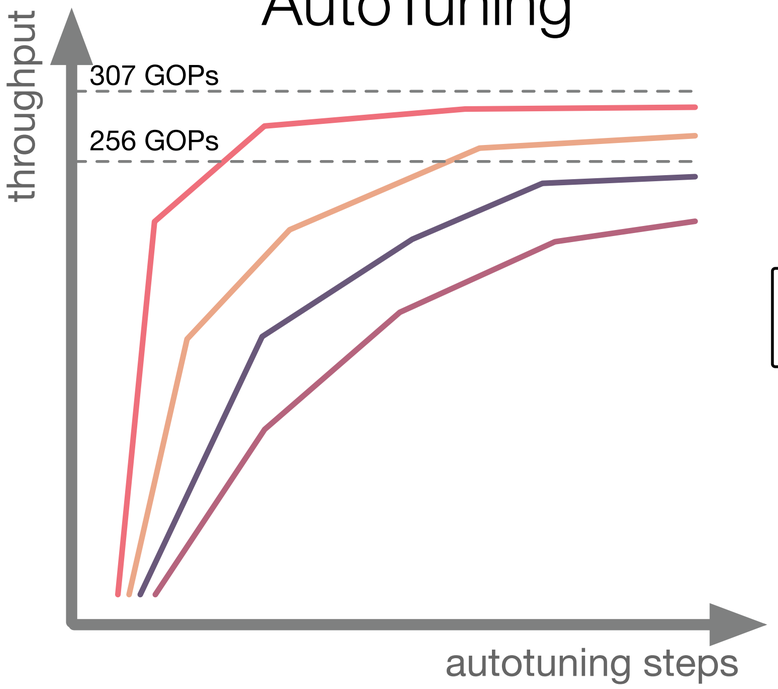
#2 Design BBB @ 307GOPs

#3 Design CCC @ 307GOPs

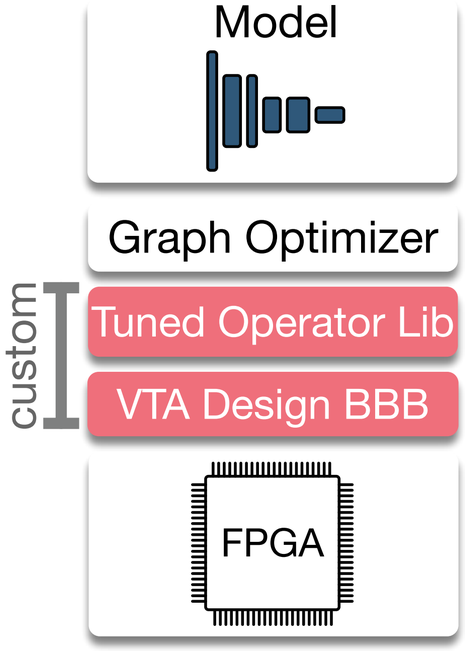
#4 Design DDD @ 256GOPs

Needs to pass place & route and pass timing closure

Operator Performance AutoTuning



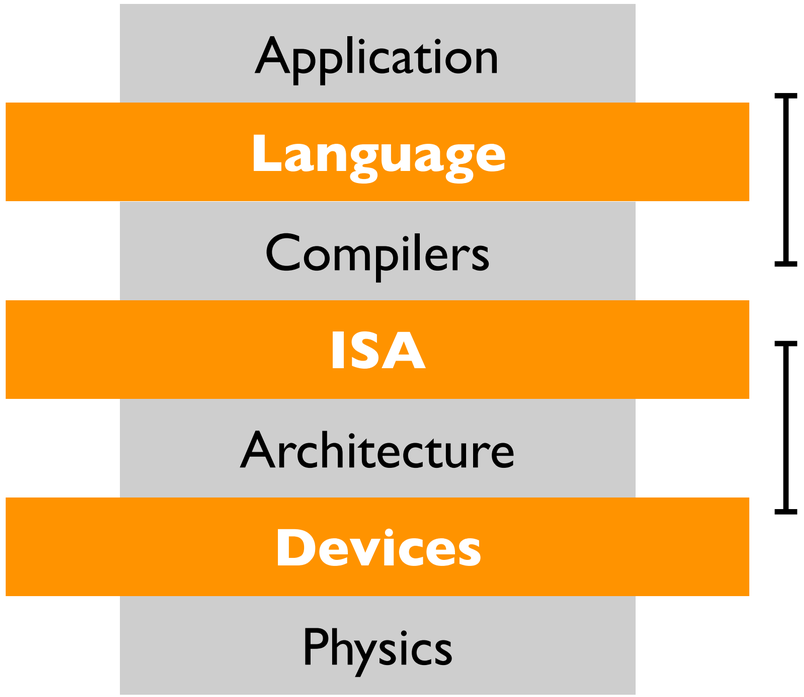
Deliverable



HW SW co-design challenges that remain to be solved

- Richer architectural and micro-architectural exploration
- Hardware cross-compilation to ASIC, other forms of FPGA accelerators
- Improved integration with existing SoC ecosystems for heterogeneous execution
- Improved automated optimization across the stack (HW/SW co-design)

Towards a bespoke HW Future?



DL compiler IRs and abstractions have evolved and diversified greatly in the recent years

The future will most probably lie in the co-design of novel hardware IRs that let us more easily explore and optimize new accelerator designs

Towards a bespoke HW Future?

Compilers

meta-ISA

meta-Architecture

Micro Arch IR

Micro Architecture

Structural HW DSL

Hardware Modules

RTL Language

Gates & Registers

Devices

meta-ISA define hardware/software interface for a vast collection of hardware variants

microarchitecture IRs will let us explore a rich space of architectural designs (e.g. can you turn a TPU into an NVDLA with a set of passes?)

structural DSLs will let us specialize modules to target different fabrication nodes, FPGAs / CGRAs to meet PPA targets

traditional Verilog, VHDL compilers that target FPGAs or ASICs

recommend reading: Lenny Truong, A Golden Age of Hardware Specialization at SNAPL 2019

Future DL HW/SW meta compilers will determine the next AI achievements

	2012	2019	2026
Model	AlexNet	Grover Max	“FutureNet”
DL Processor	GTX580 GPU	TPUv3	autoPU
Compute Throughput	3.2 TFLOPS	13.4 PFLOPs	~60 EFlops
Time to Train	1 week	2 weeks	2-4 weeks
Task	Human-Level Image Classification	Human-Level Generation / Detection of Fake News	Human-Level <TBD>

The Past, Present, and Future of Deep Learning Acceleration Stacks

Thierry Moreau, Post Doctorate Researcher at University of Washington

ARM Research Summit, RAAW Workshop, September 16th 2019

Faculty

- Luis Ceze, Professor
- Carlos Guestrin, Professor
- Arvind Krishnamurthy, Professor
- Matthai Philipose, Affiliate Professor
- Zachary Tatlock, Assistant Professor

Researchers

- Thierry Moreau

Graduate Students

- Tianqi Chen
- Meghan Cowan
- Josh Fromm
- Ziheng Jiang
- Liang Luo
- Steven Lyubomirsky
- Pratyush Patel
- Jared Roesch
- Gus Smith
- Luis Vega
- Logan Weber
- Eddie Yan

sampl

PAUL G. ALLEN SCHOOL OF COMPUTER SCIENCE & ENGINEERING

tvm.ai

Discussion

- Will the Cambrian explosion of IRs, DSLs and hardware continue or will natural selection take over?
- Can we continue to think of hardware vs. software companies in an era of codesign?
- What role will academia play in the next years?