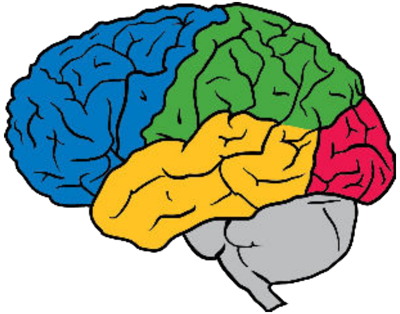


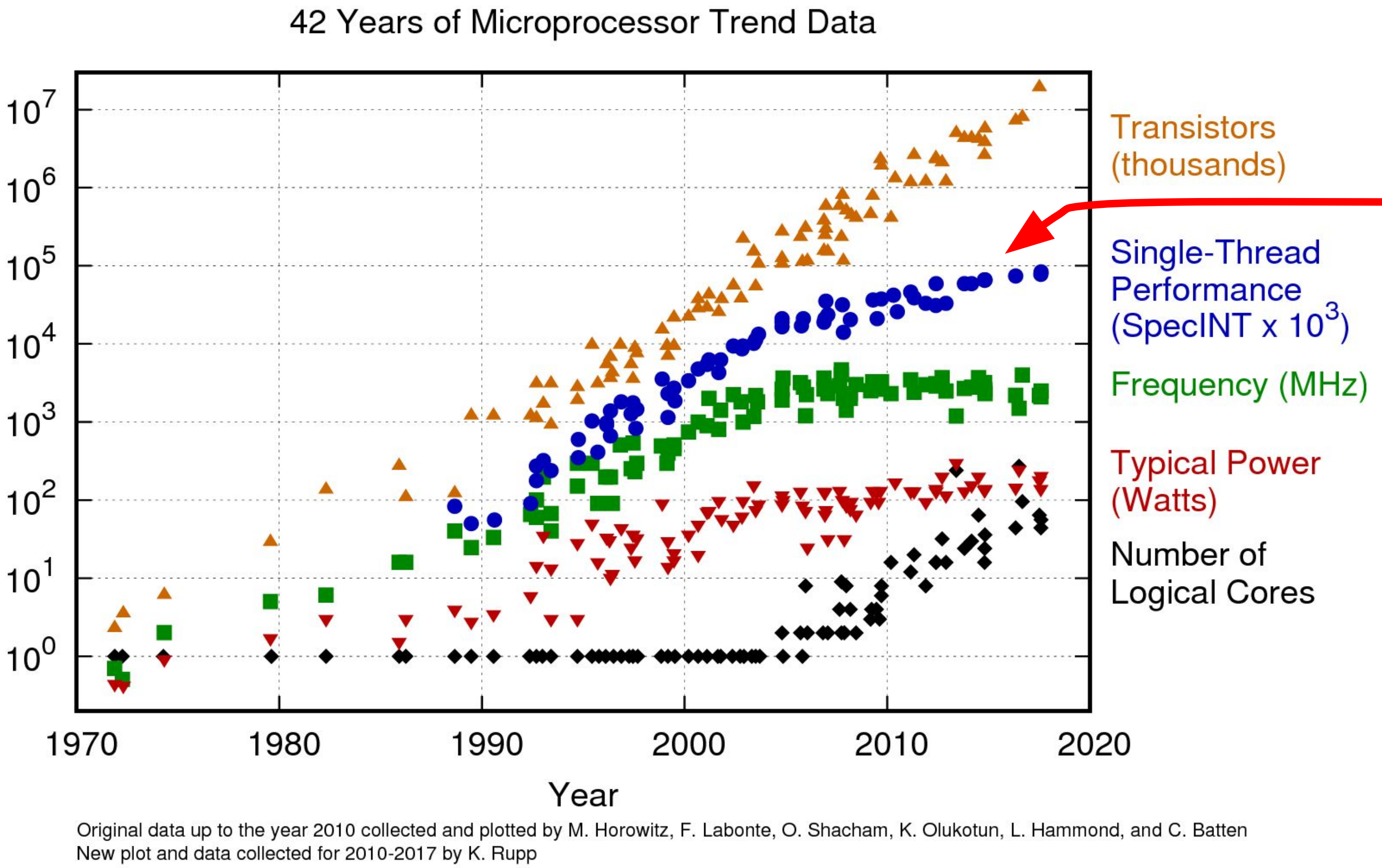
Systems and Machine Learning

Jeff Dean
Google Brain team
g.co/brain

Presenting the work of **many** people at Google

Systems for Machine Learning

General Purpose Processor Performance Trends



Single-core performance plateauing after decades of exponential growth

Just when deep learning is creating insatiable computation demands

Training powerful models that are computationally-expensive on:

- Terabyte or petabyte-sized training datasets

  Plus techniques like AutoML ("Learning to learn", Neural Architecture Search, etc.) can multiply desired training computation by 5-1000X

Inference using expensive deep models in systems with:

- hundreds of thousands of requests per second
- latency requirements of tens of milliseconds
- billions of users

2008: U.S. National Academy of Engineering publishes

Grand Engineering Challenges for 21st Century

- Make solar energy affordable
- Provide energy from fusion
- Develop carbon sequestration methods
- Manage the nitrogen cycle
- Provide access to clean water
- Restore & improve urban infrastructure
- Advance health informatics
- Engineer better medicines
- Reverse-engineer the brain
- Prevent nuclear terror
- Secure cyberspace
- Enhance virtual reality
- Advance personalized learning
- Engineer the tools for scientific discovery

www.engineeringchallenges.org/challenges.aspx

Restore & improve urban infrastructure

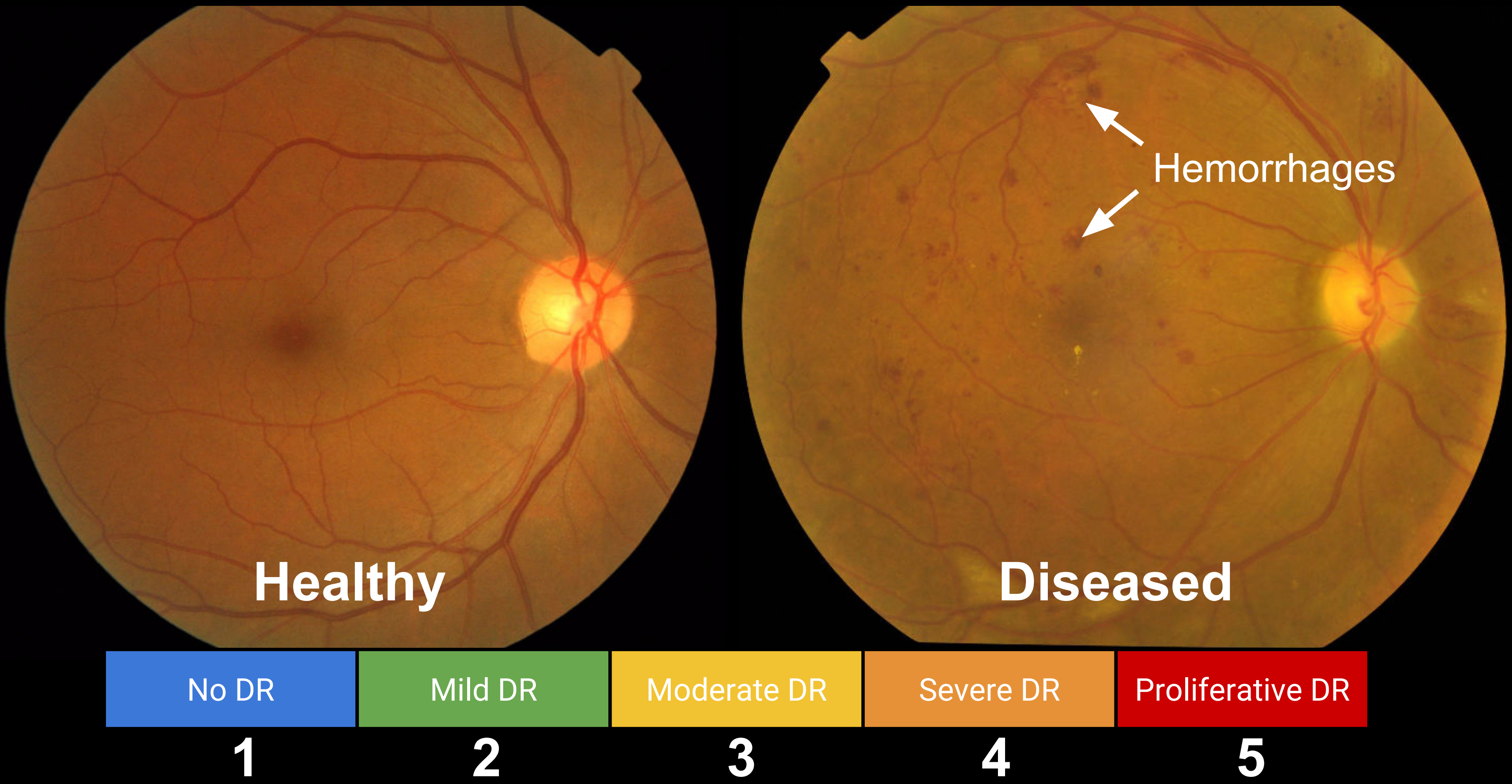
WAYMO

3 million miles self-driven

We drive more than 25,000 autonomous miles each week, largely on complex city streets. That's on top of 1 billion simulated miles we drove just in 2016.

https://waymo.com/tech/

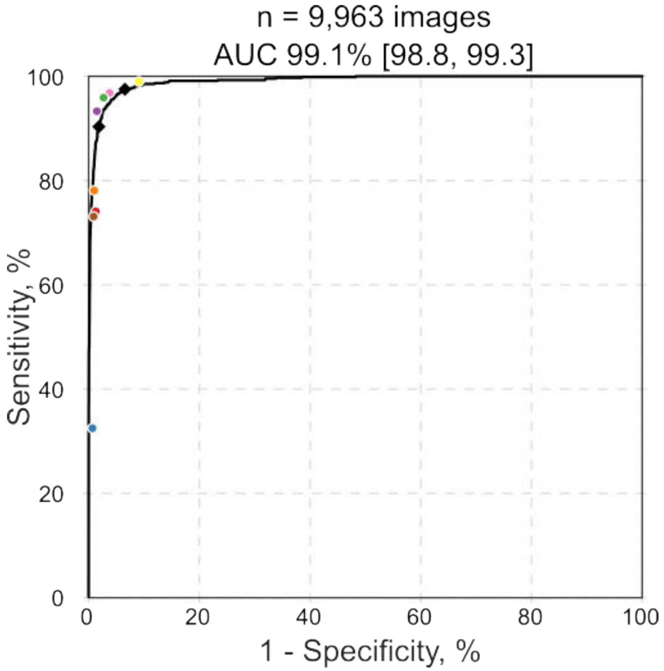
Advance health informatics


Hemorrhages
Healthy
Diseased
No DR
Mild DR
Moderate DR
Severe DR
Proliferative DR
1
2
3
4
5

JAMA The Journal of the American Medical Association

JAMA | Original Investigation | INNOVATIONS IN HEALTH CARE DELIVERY

Development and Validation of a Deep Learning Algorithm for Detection of Diabetic Retinopathy in Retinal Fundus Photographs



F-score

0.95	0.91
Algorithm	Ophthalmologist (median)

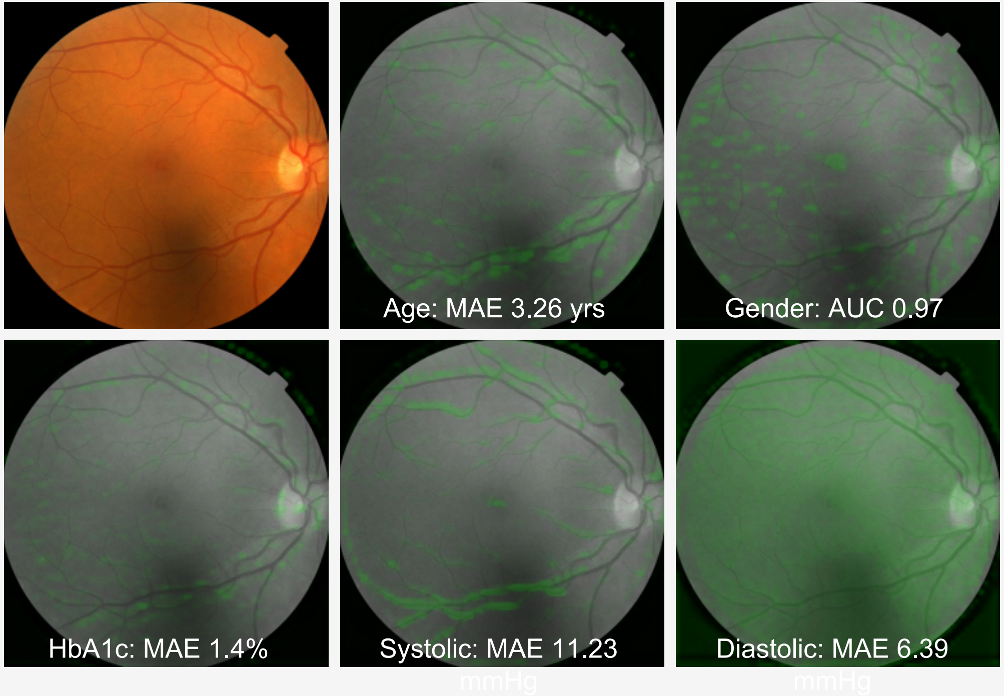
"The study by Gulshan and colleagues **truly represents the brave new world in medicine.**"

Dr. Andrew Beam, Dr. Isaac Kohane
Harvard Medical School

"Google just published this paper in JAMA (impact factor 37) [...] **It actually lives up to the hype**."

Dr. Luke Oakden-Rayner
University of Adelaide

Completely new, novel scientific discoveries



Predicting things that doctors can’t predict from imaging

Potential as a new biomarker

Preliminary 5-yr MACE AUC: 0.7

Can we predict cardiovascular risk? If so, this is a very nice non-invasive way of doing so

Can we also predict treatment response?

R. Poplin, A. Varadarajan *et al.* Predicting Cardiovascular Risk Factors from Retinal Fundus Photographs using Deep Learning. *Nature Biomedical Engineering, 2018.*

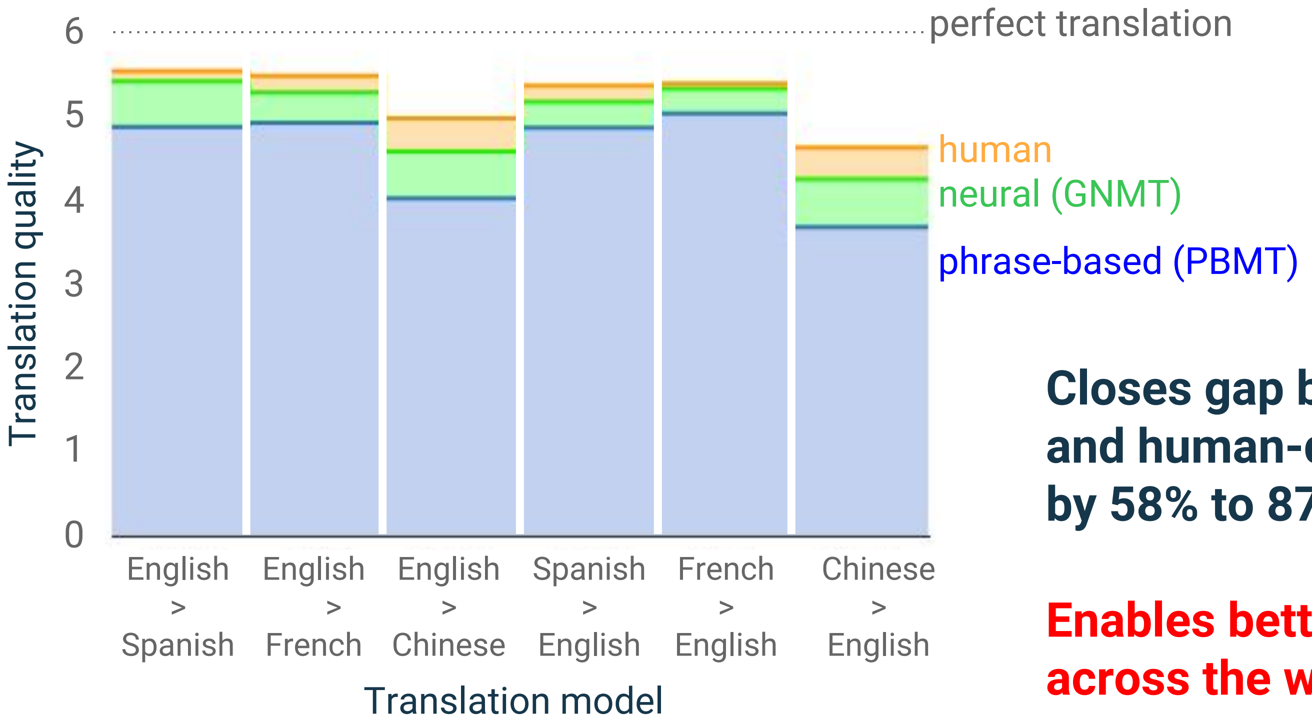
Predictive tasks for healthcare

Given a patient's electronic medical record data, **can we predict the future**?

Deep learning methods for sequential prediction are becoming extremely good

e.g. recent improvements in Google Translation

Neural Machine Translation



Closes gap between old system and human-quality translation by 58% to 87%

Enables better communication across the world

research.googleblog.com/2016/09/a-neural-network-for-machine.html

Predictive tasks for healthcare

Given a large corpus of training data of de-identified medical records, can we predict interesting aspects of the future for a patient not in the training set?

- *will patient be readmitted to hospital in next N days?*
- *what is the likely length of hospital stay for patient checking in?*
- *what are the most likely diagnoses for the patient right now?*
- *what medications should a doctor consider prescribing?*
- *what tests should be considered for this patient?*
- *which patients are at highest risk for X in next month?*

and why*?*

Collaborating with several healthcare organizations, including UCSF, Stanford, and Univ. of Chicago.

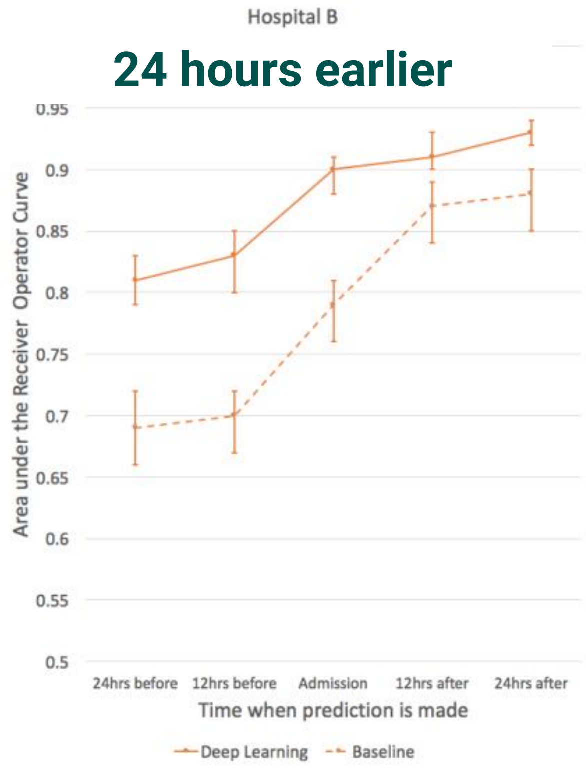
Medical Records Prediction Results

Scalable and accurate deep learning for electronic health records

Alvin Rajkomar*[1,2], Eyal Oren*[1], Kai Chen[1], Andrew M. Dai[1], Nissan Hajaj[1], Peter J. Liu[1], Xiaobing Liu[1], Mimi Sun[1], Patrik Sundberg[1], Hector Yee[1], Kun Zhang [1], Yi Zhang[1], Gavin E. Duggan[1], Gerardo Flores[1], Michaela Hardt[1], Jamie Irvine[1], Quoc Le[1], Kurt Litsch[1], Jake Marcus[1], Alexander Mossin[1], Justin Tansuwan[1], De Wang[1], James Wexler[1], Jimbo Wilson[1], Dana Ludwig[2], Samuel L. Volchenboum[4], Katherine Chou[1], Michael Pearson[1], Srinivasan Madabushi[1], Nigam H. Shah[3], Atul J. Butte[2], Michael Howell[1], Claire Cui[1], Greg Corrado[1], and Jeff Dean[1]

[1]Google Inc, Mountain View, California
[2]University of California, San Francisco, San Francisco, California
[3]Stanford University, Stanford, California
[4]University of Chicago Medicine, Chicago, Illinois

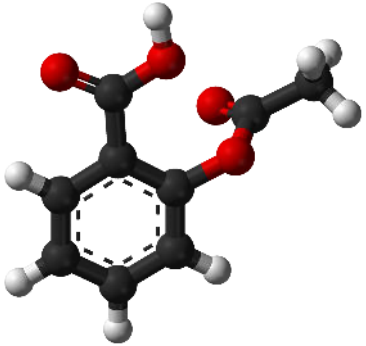
January 2018



https://arxiv.org/abs/1801.07860

Engineer better medicines
and maybe...
Make solar energy affordable
Develop carbon sequestration methods
Manage the nitrogen cycle

Predicting Properties of Molecules



DFT (density functional theory) simulator

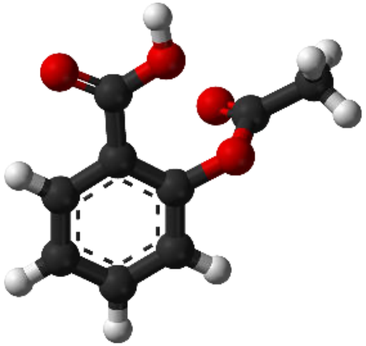
$\sim 10^3$ seconds

Toxic?

Bind with a given protein?

Quantum properties: E, ω_0, ...

Predicting Properties of Molecules



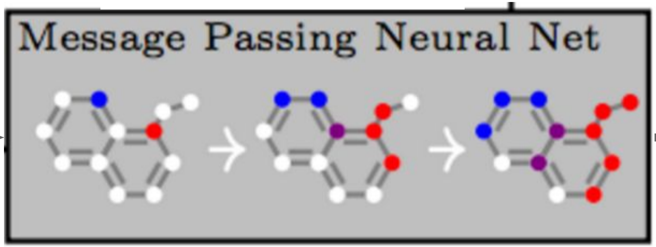
DFT (density functional theory) simulator

$\sim 10^3$ seconds

Toxic?

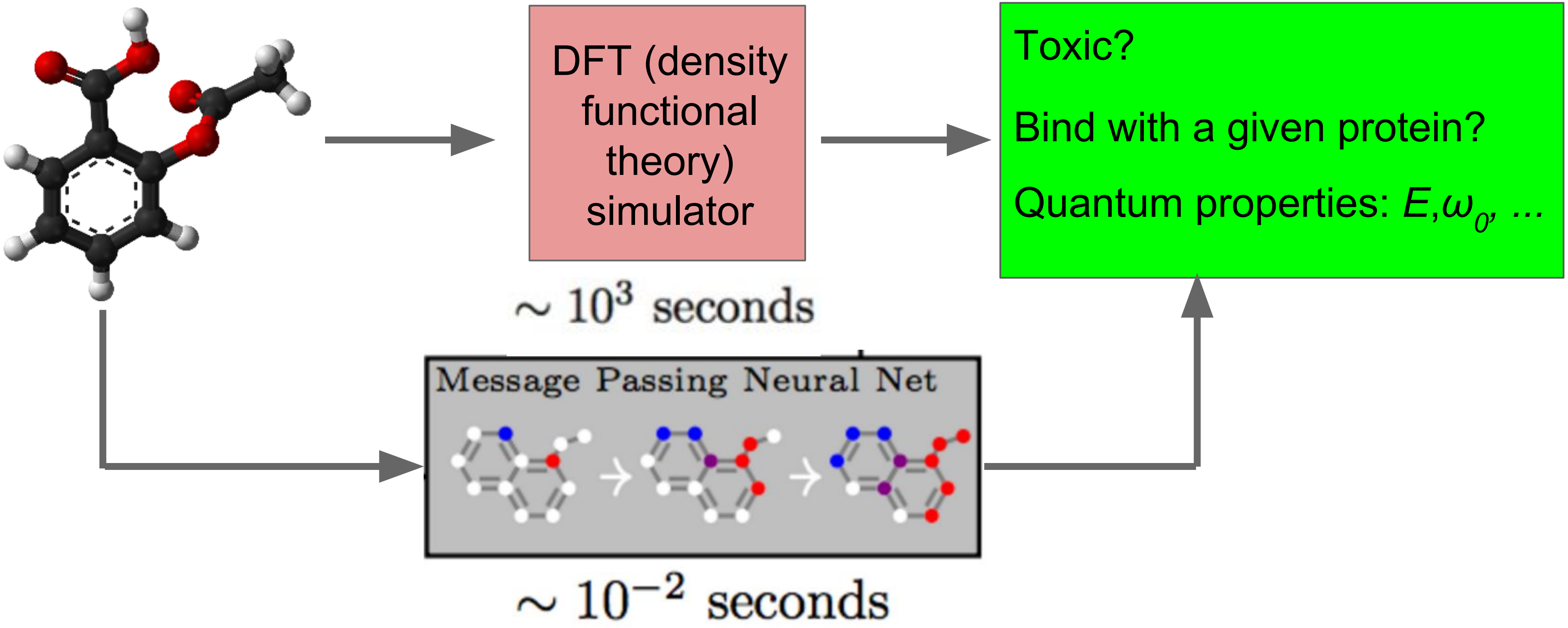
Bind with a given protein?

Quantum properties: E, ω_0, ...



$\sim 10^{-2}$ seconds

Predicting Properties of Molecules

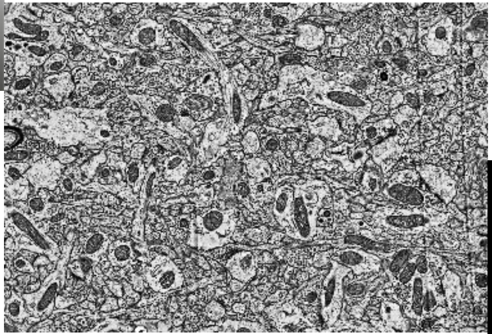


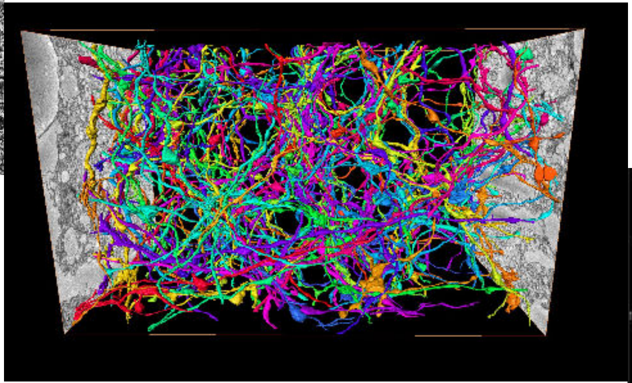
- State of the art results predicting output of expensive quantum chemistry calculations, but **~300,000 times faster**

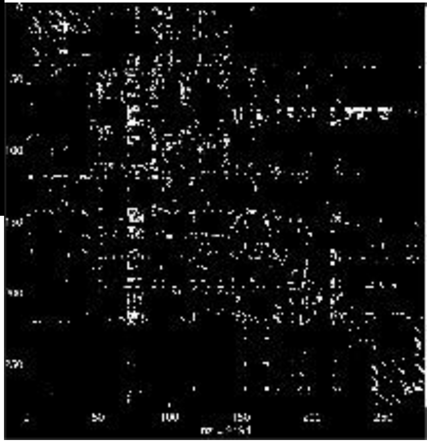
https://research.googleblog.com/2017/04/predicting-properties-of-molecules-with.html and https://arxiv.org/abs/1702.05532 and https://arxiv.org/abs/1704.01212 (latter to appear in ICML 2017)

Reverse engineer the brain

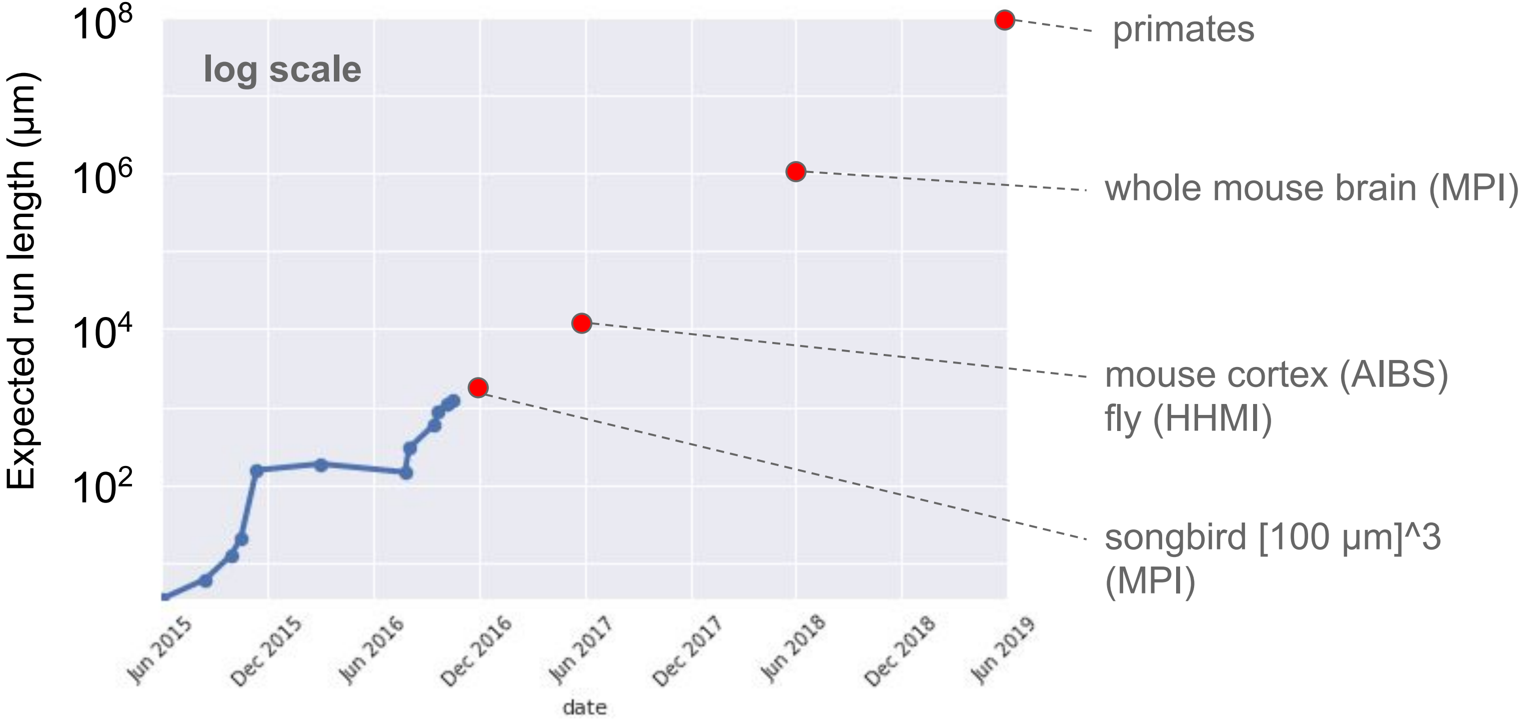
Connectomics: Reconstructing Neural Circuits from High-Resolution Brain Imaging







Automated Reconstruction Progress at Google



Metric: Expected Run Length (ERL)
"mean microns between failure" of automated neuron tracing

Google

New Technology: Flood Filling Networks

Flood-Filling Networks

Michał Januszewski
Google
mjanusz@google.com

Jeremy Maitin-Shepard
Google
jbms@google.com

Peter Li
Google
phli@google.com

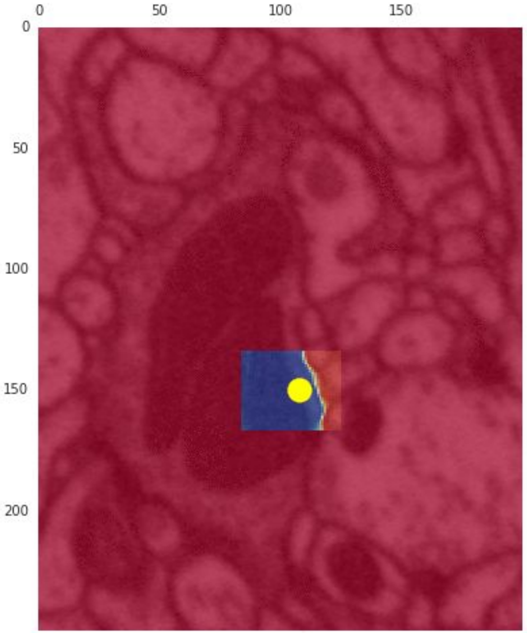
Jörgen Kornfeld
Max Planck Institute for Neurobiology
kornfeld@neuro.mpg.de

Winfried Denk
Max Planck Institute for Neurobiology
winfried.denk@neuro.mpg.de

Viren Jain
Google
viren@google.com

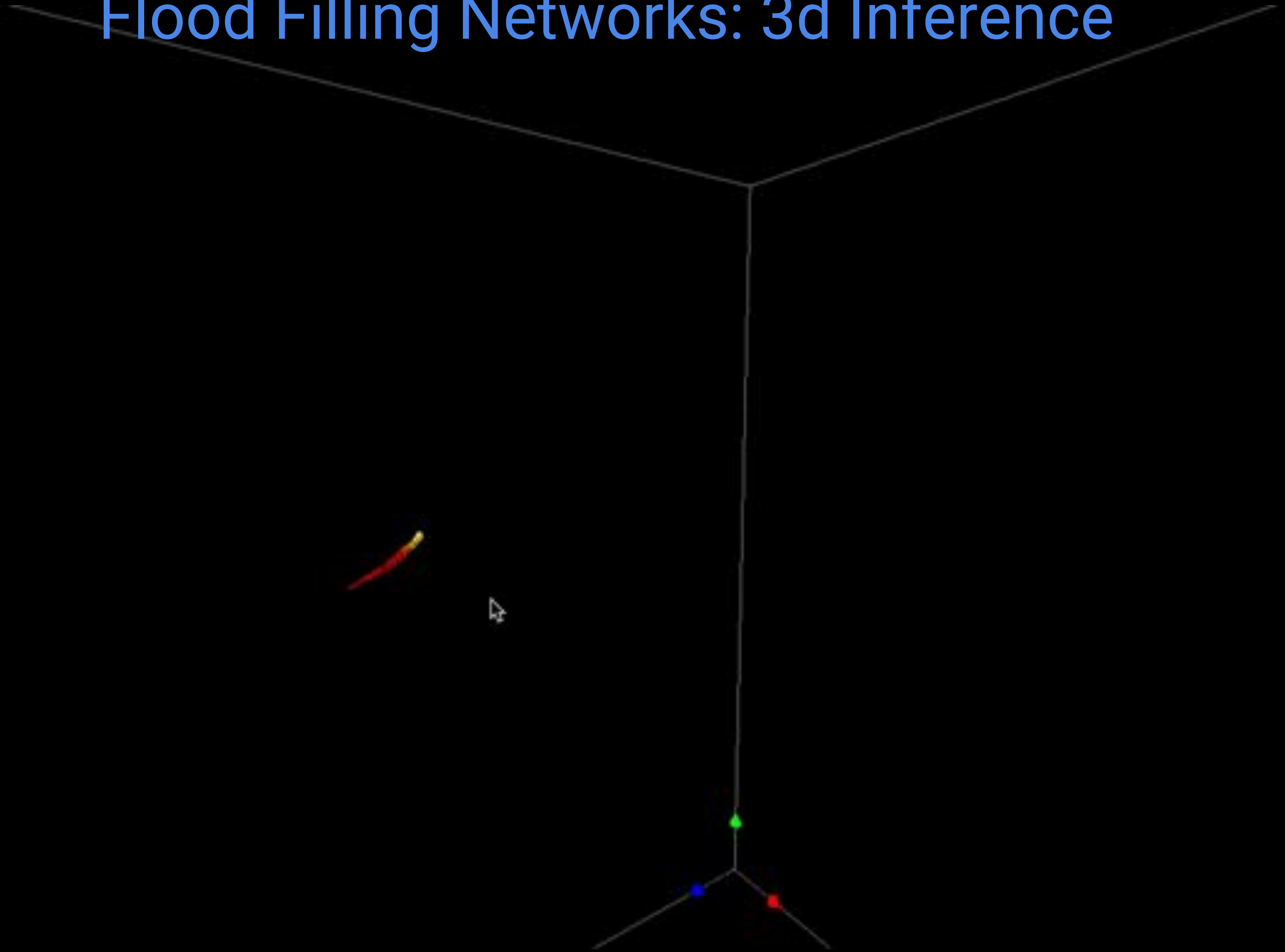
- Start with a seed point
- Recurrent neural network iteratively fills out an object based on image content and its own previous predictions

2d Inference

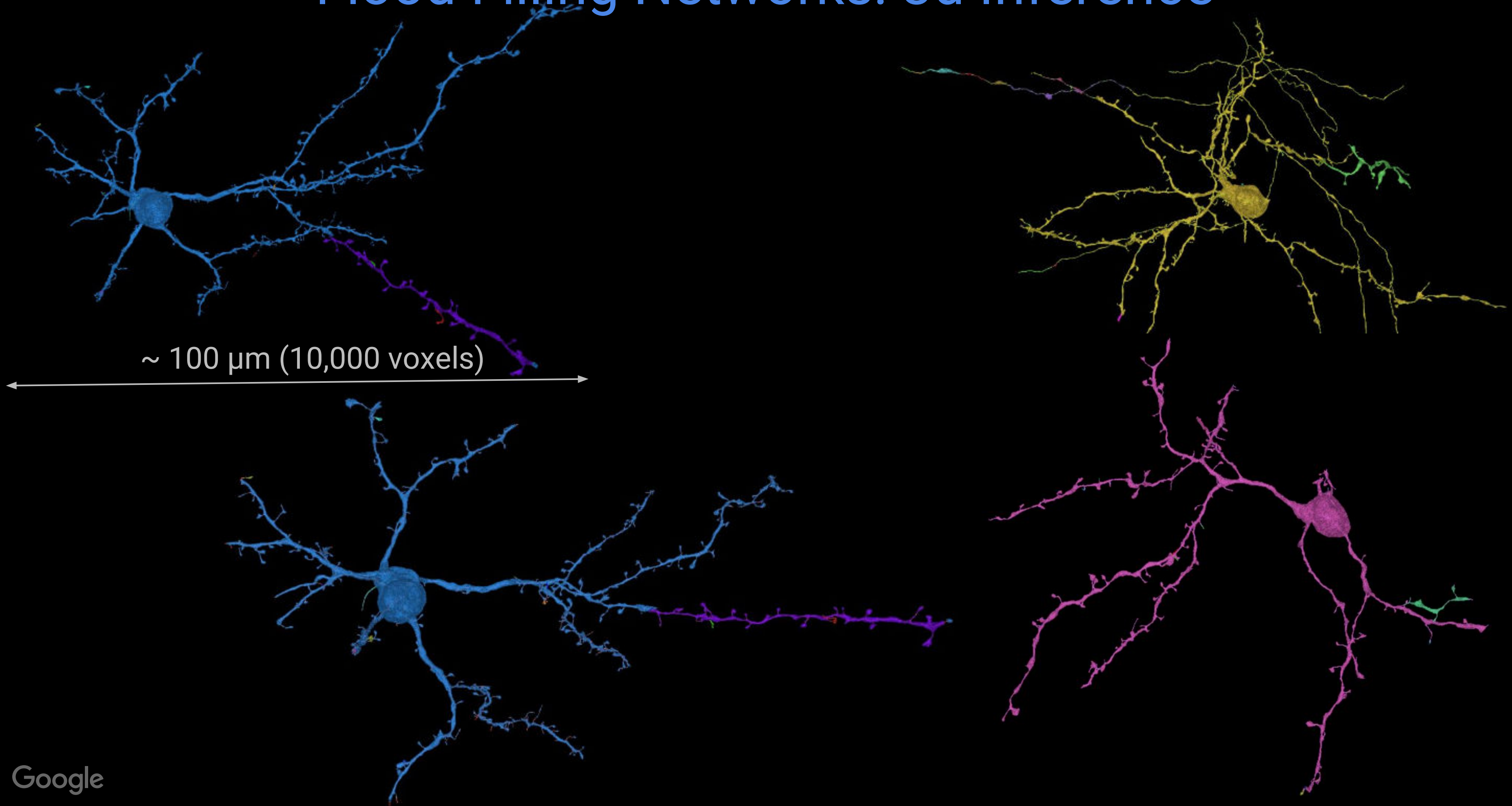


https://arxiv.org/abs/1611.00421

Flood Filling Networks: 3d Inference

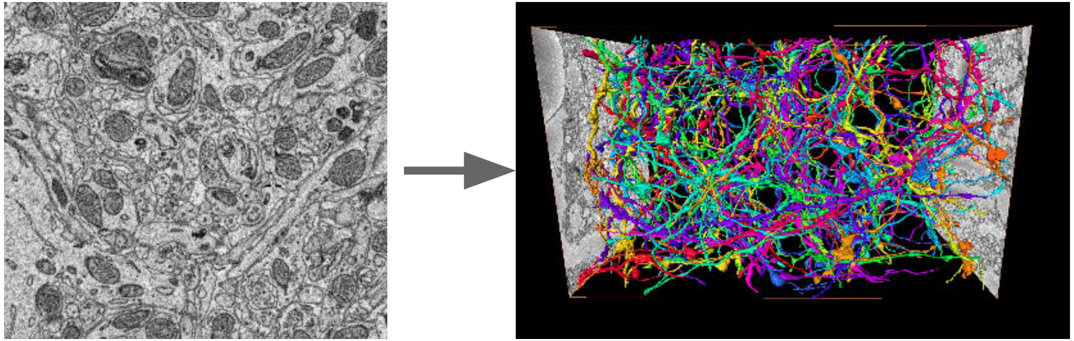


Flood Filling Networks: 3d Inference



Songbird Brain Wiring Diagram

- Raw data produced by Max Planck Institute for Neurobiology using serial block face scanning electron microscopy
- 10,600 × 10,800 × 5,700 voxels = ~600 billion voxels
- Goal: Reconstruct **complete connectivity** and use to **test specific hypotheses** related to how biological nervous systems produce precise, sequential motor behaviors and perform reinforcement learning.



Courtesy Jorgen Kornfeld & Winfried Denk, MPI

Engineer the Tools of Scientific Discovery

TensorFlow

http://tensorflow.org/

and

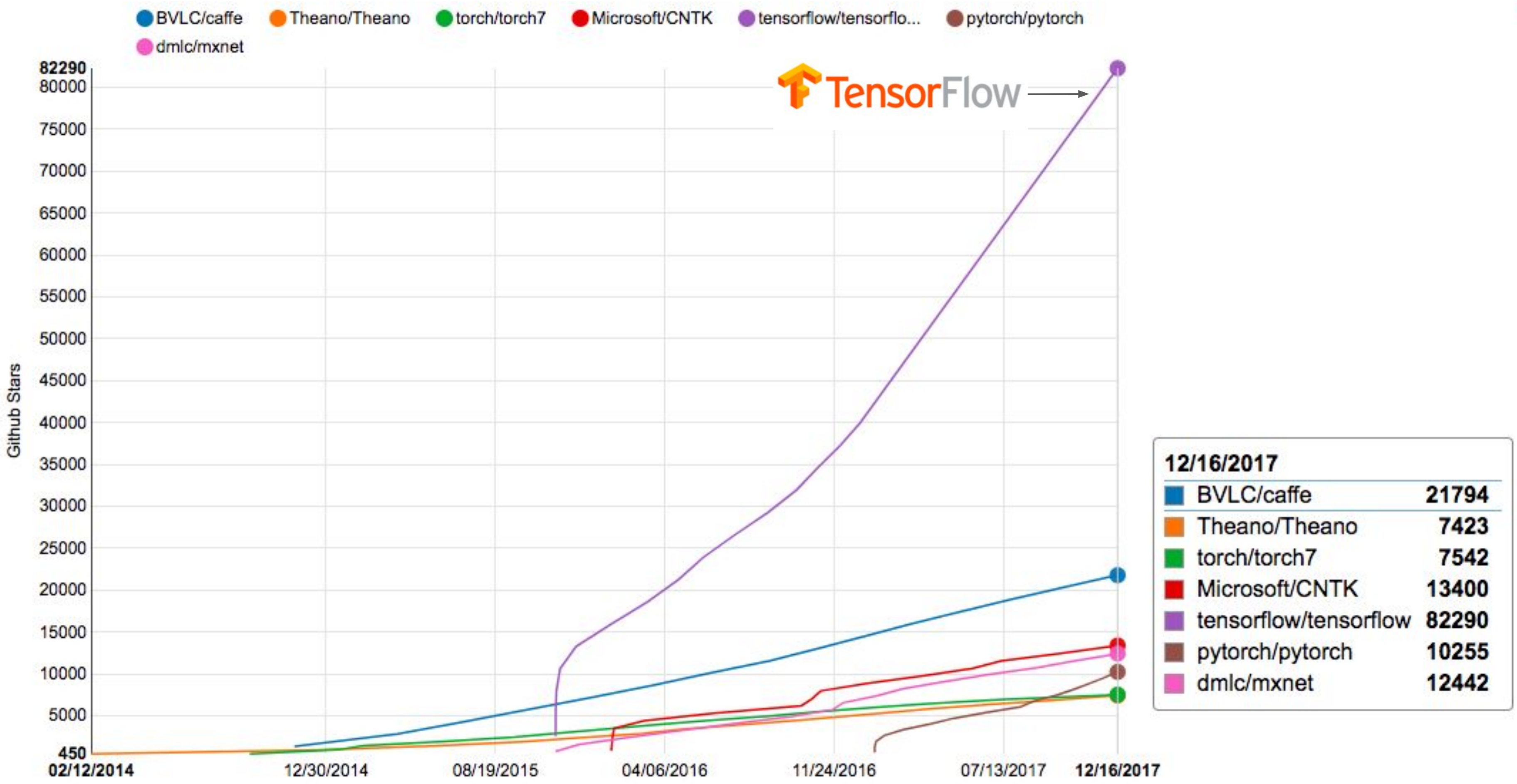
https://github.com/tensorflow/tensorflow

Open, standard software for general machine learning

Great for Deep Learning in particular

First released Nov 2015

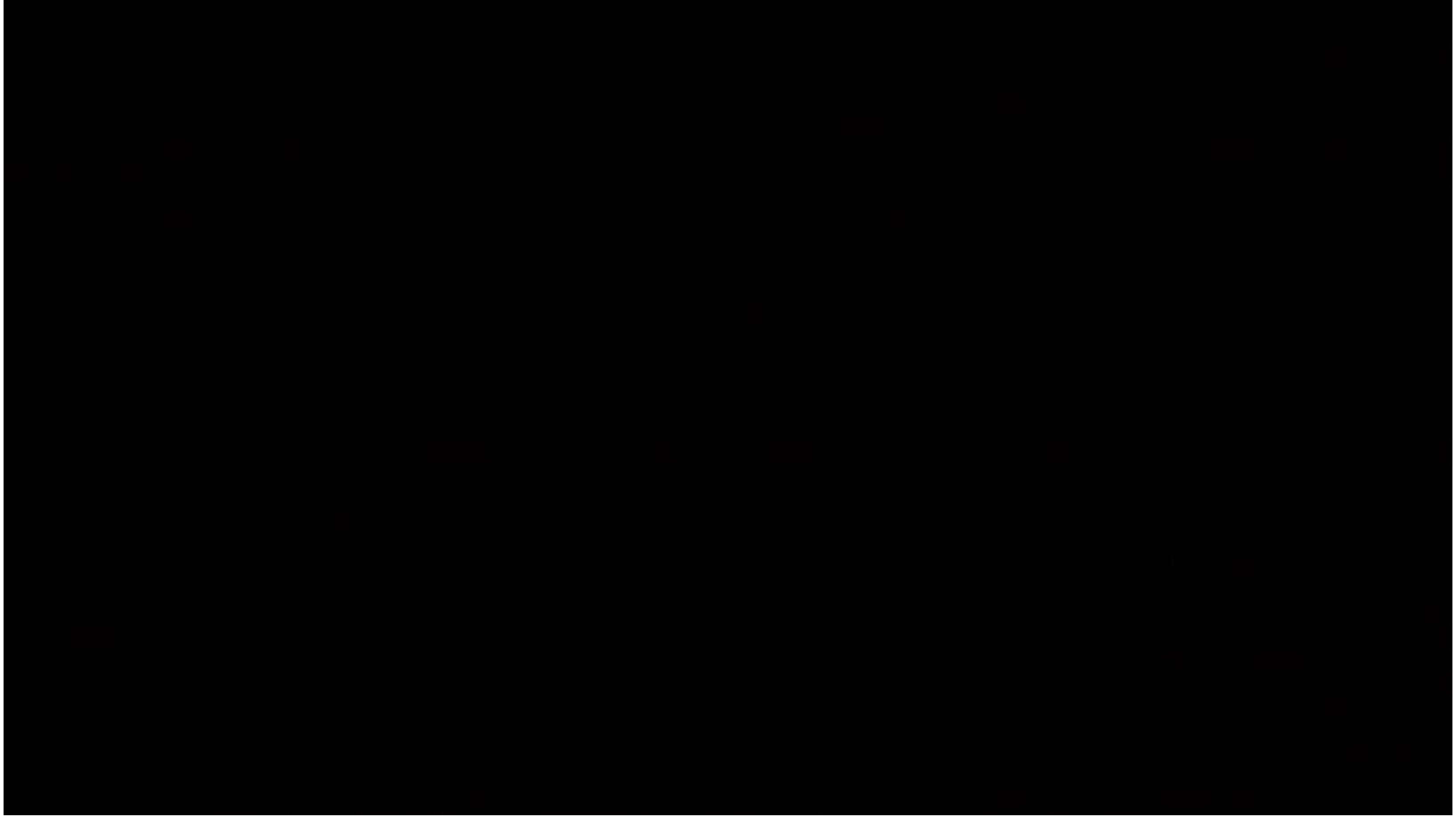
Apache 2.0 license



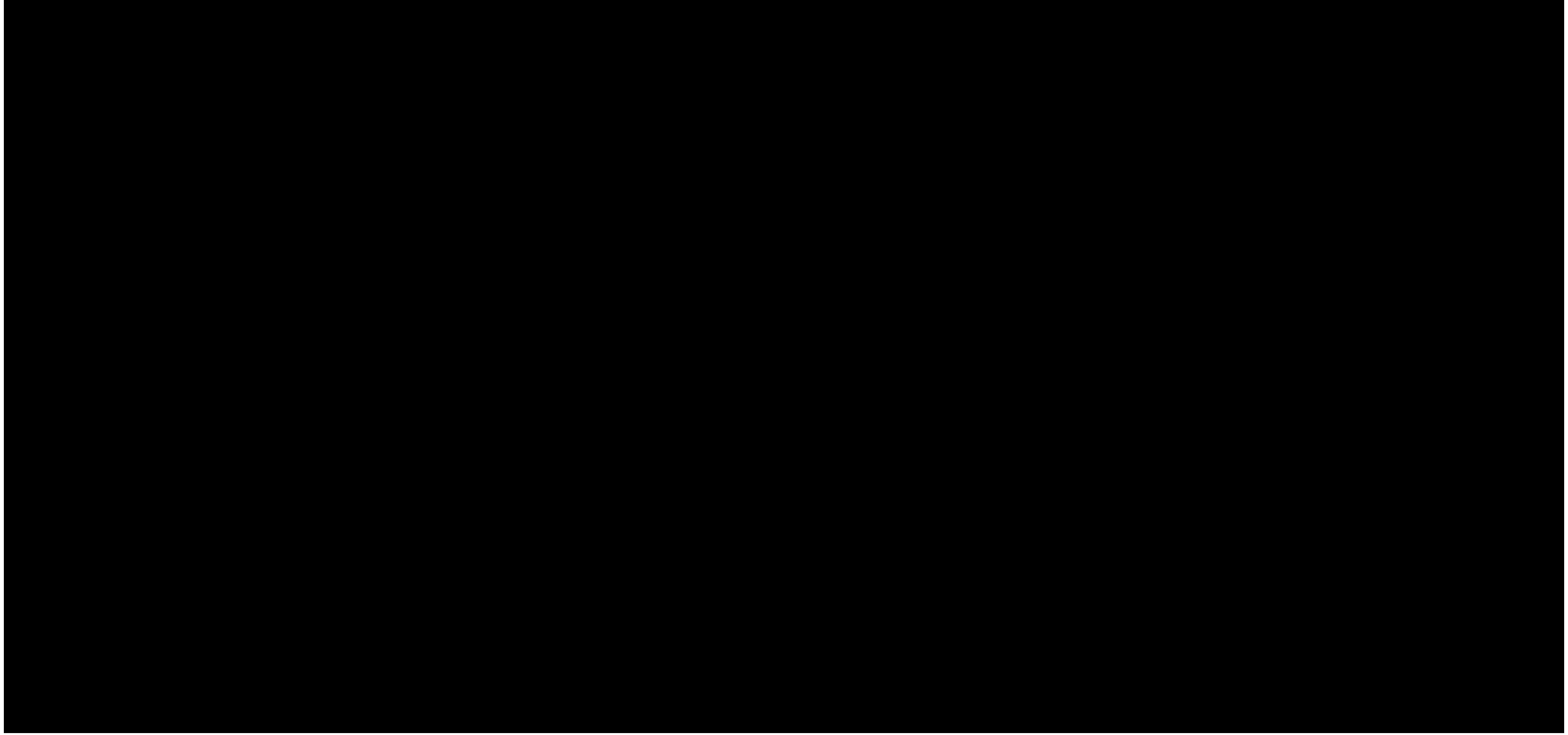
BVLC/caffe
Theano/Theano
torch/torch7
Microsoft/CNTK
tensorflow/tensorflo...
pytorch/pytorch
dmlc/mxnet
TensorFlow
Github Stars
82290
80000
75000
70000
65000
60000
55000
50000
45000
40000
35000
30000
25000
20000
15000
10000
5000
450
02/12/2014
12/30/2014
08/19/2015
04/06/2016
11/24/2016
07/13/2017
12/16/2017
12/16/2017
BVLC/caffe 21794
Theano/Theano 7423
torch/torch7 7542
Microsoft/CNTK 13400
tensorflow/tensorflow 82290
pytorch/pytorch 10255
dmlc/mxnet 12442

Machine Learning for Finding Planets

Google



www.nasa.gov/press-release/artificial-intelligence-nasa-data-used-to-discover-eighth-planet-circling-distant-star
Blog: www.blog.google/topics/machine-learning/hunting-planets-machine-learning/
Paper: [Shallue & Vandenburg], www.cfa.harvard.edu/~avanderb/kepler90i.pdf



IDENTIFYING EXOPLANETS WITH DEEP LEARNING: A FIVE PLANET RESONANT CHAIN AROUND KEPLER-80 AND AN EIGHTH PLANET AROUND KEPLER-90

CHRISTOPHER J. SHALLUE[†1] & ANDREW VANDERBURG[*, 2,3]

www.nasa.gov/press-release/artificial-intelligence-nasa-data-used-to-discover-eighth-planet-circling-distant-star
Blog: www.blog.google/topics/machine-learning/hunting-planets-machine-learning/
Paper: [Shallue & Vandenburg], www.cfa.harvard.edu/~avanderb/kepler90i.pdf

IDENTIFYING EXOPLANETS WITH DEEP LEARNING: A FIVE PLANET RESONANT CHAIN AROUND KEPLER-80 AND AN EIGHTH PLANET AROUND KEPLER-90

CHRISTOPHER J. SHALLUE† [1] & ANDREW VANDERBURG*, [2,3]

www.nasa.gov/press-release/artificial-intelligence-nasa-data-used-to-discover-eighth-planet-circling-distant-star

Blog: www.blog.google/topics/machine-learning/hunting-planets-machine-learning/

Paper: [Shallue & Vandenburg], www.cfa.harvard.edu/~avanderb/kepler90i.pdf

https://www.blog.google/topics/machine-learning/using-tensorflow-keep-farmers-happy-and-cows-healthy/

https://www.blog.google/topics/machine-learning/fight-against-illegal-deforestation-tensorflow/

AutoML: Automated machine learning
(“learning to learn”)

Current:

Solution = ML expertise + data + computation

Current:

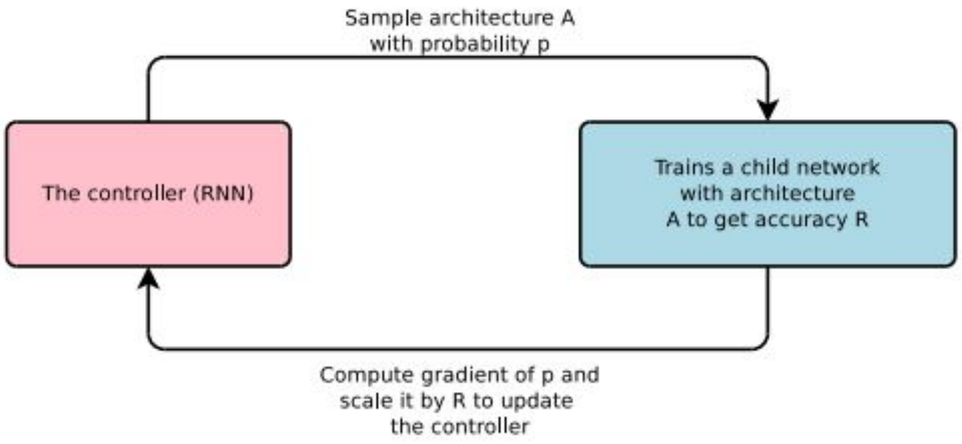
Solution = ML expertise + data + computation

Can we turn this into:

Solution = data + 100X computation

???

Neural Architecture Search

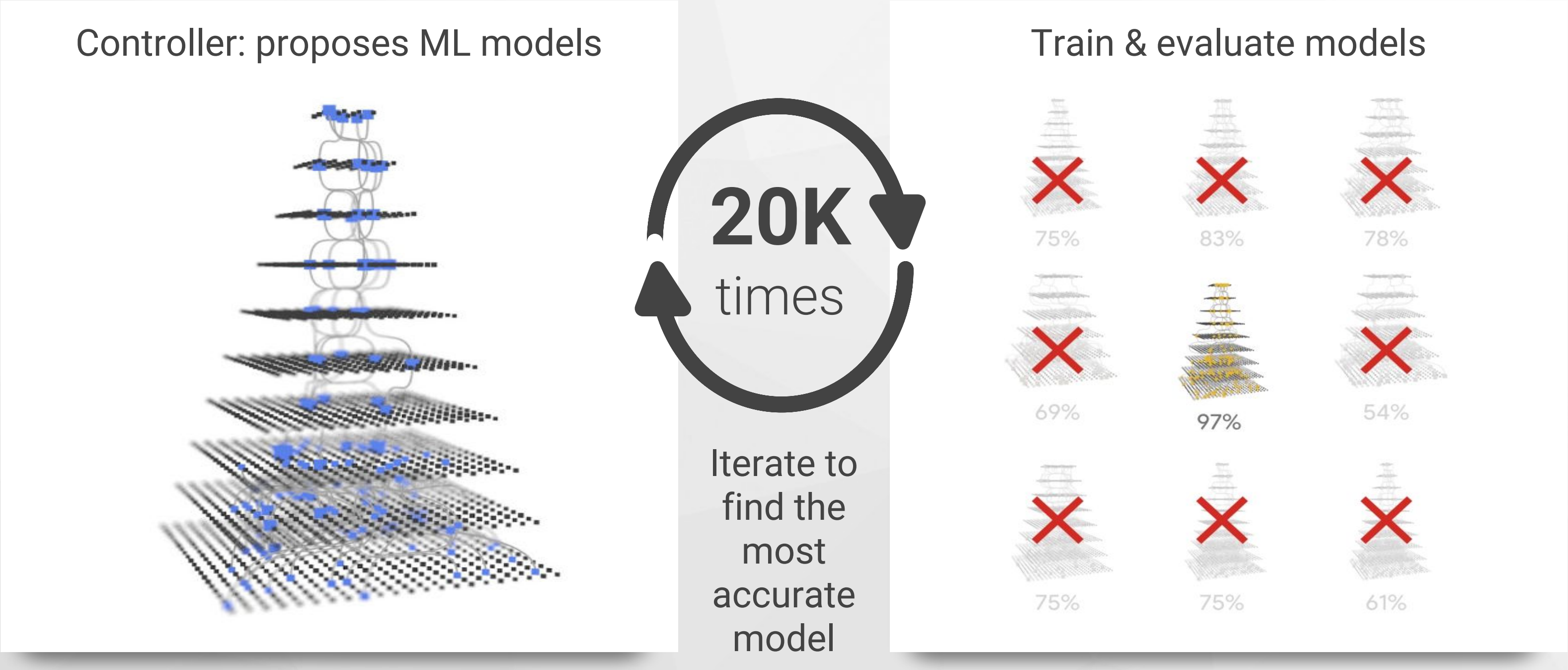


Idea: model-generating model trained via reinforcement learning

(1) Generate ten models
(2) Train them for a few hours
(3) Use loss of the generated models as reinforcement learning signal

Neural Architecture Search with Reinforcement Learning, Zoph & Le, ICLR 2016
arxiv.org/abs/1611.01578

Neural Architecture Search to find a model



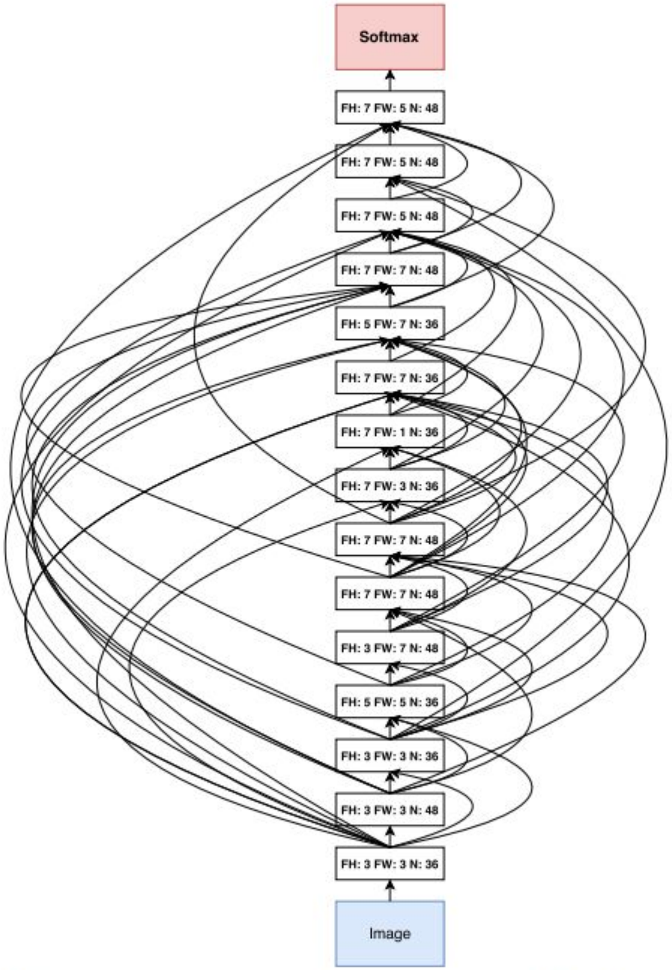
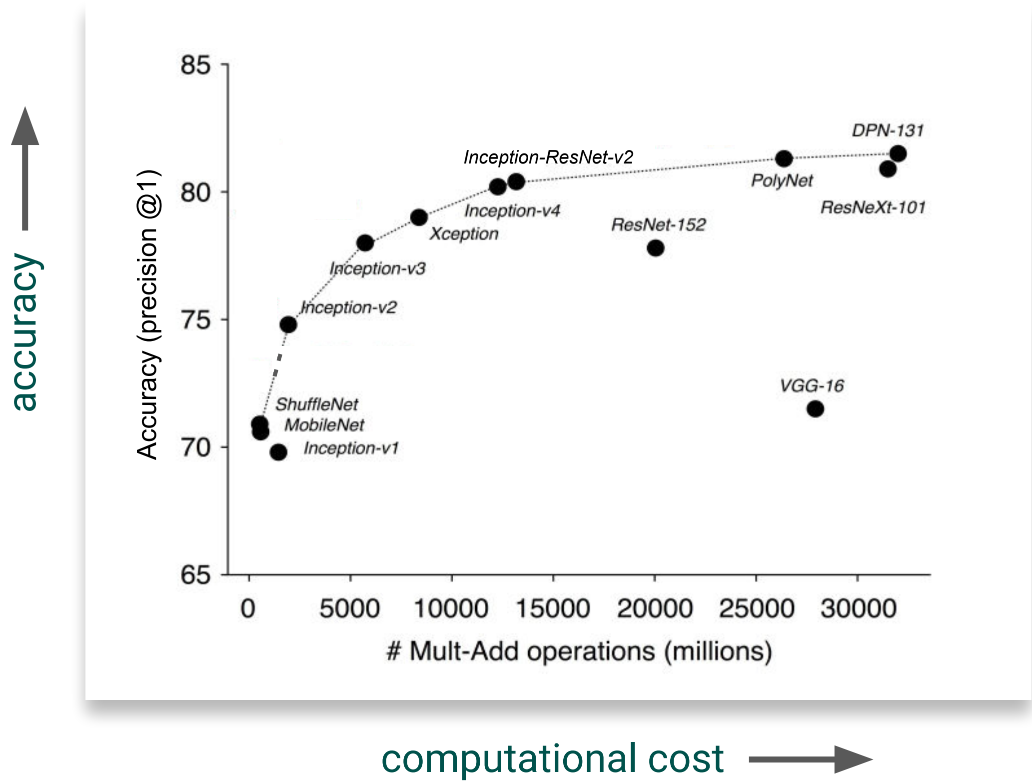


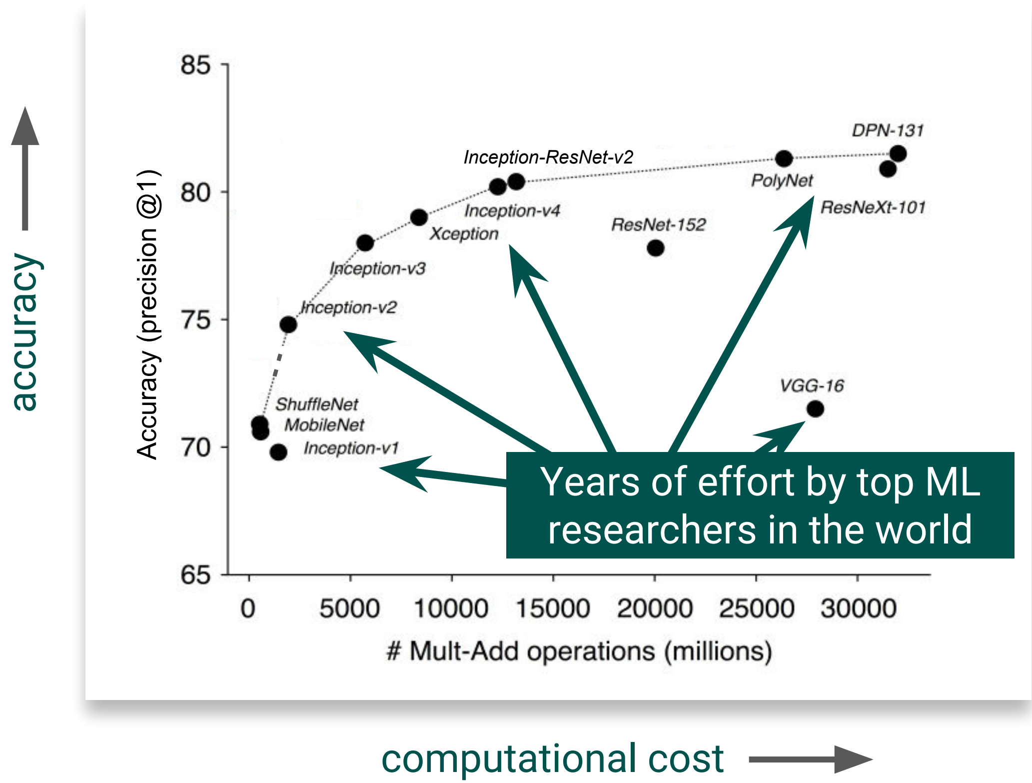
Figure 7: Convolutional architecture discovered by our method, when the search space does not have strides or pooling layers. FH is filter height, FW is filter width and N is number of filters.

AutoML outperforms handcrafted models



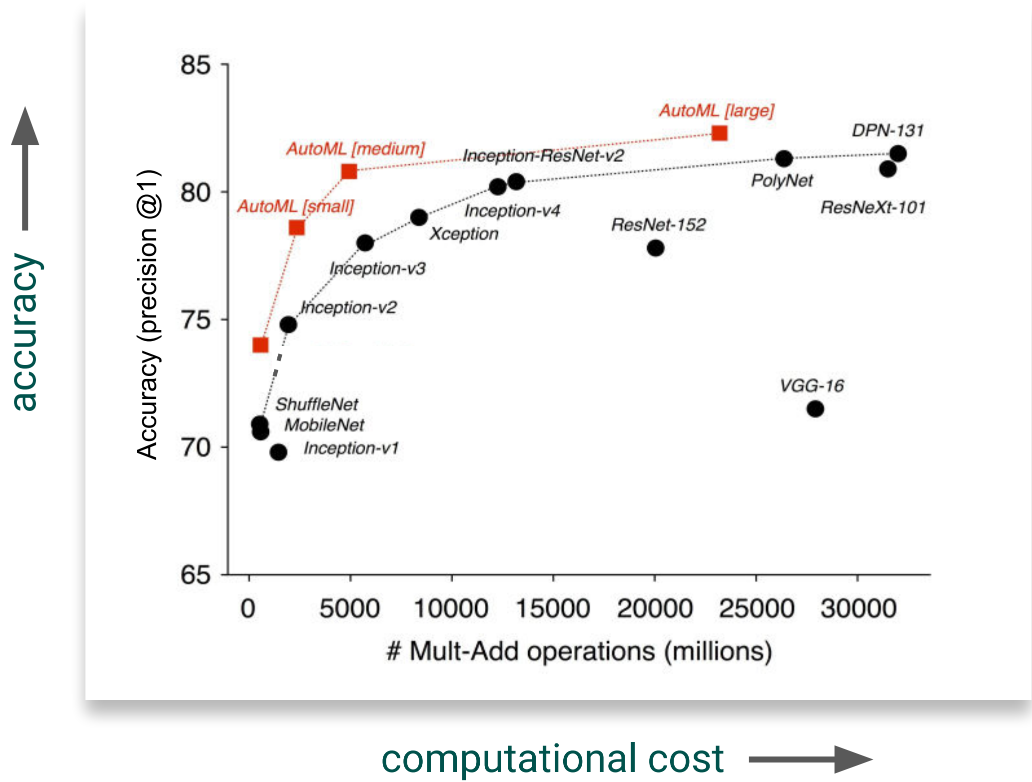
Learning Transferable Architectures for Scalable Image Recognition, Zoph et al. 2017, https://arxiv.org/abs/1707.07012

AutoML outperforms handcrafted models



Learning Transferable Architectures for Scalable Image Recognition, Zoph et al. 2017, https://arxiv.org/abs/1707.07012

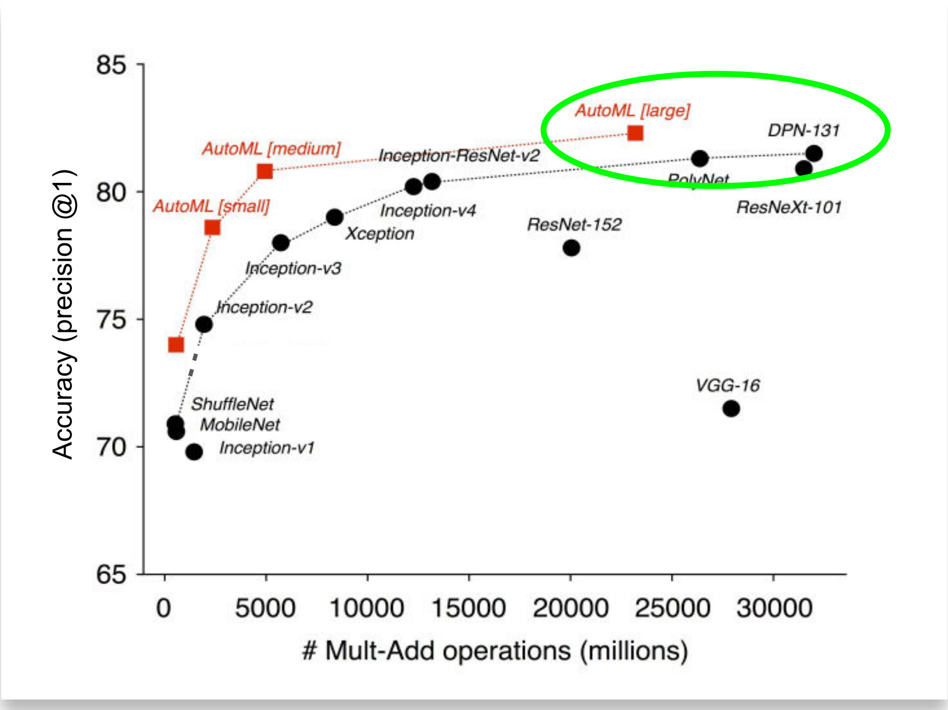
AutoML outperforms handcrafted models



Learning Transferable Architectures for Scalable Image Recognition, Zoph et al. 2017, https://arxiv.org/abs/1707.07012

AutoML outperforms handcrafted models

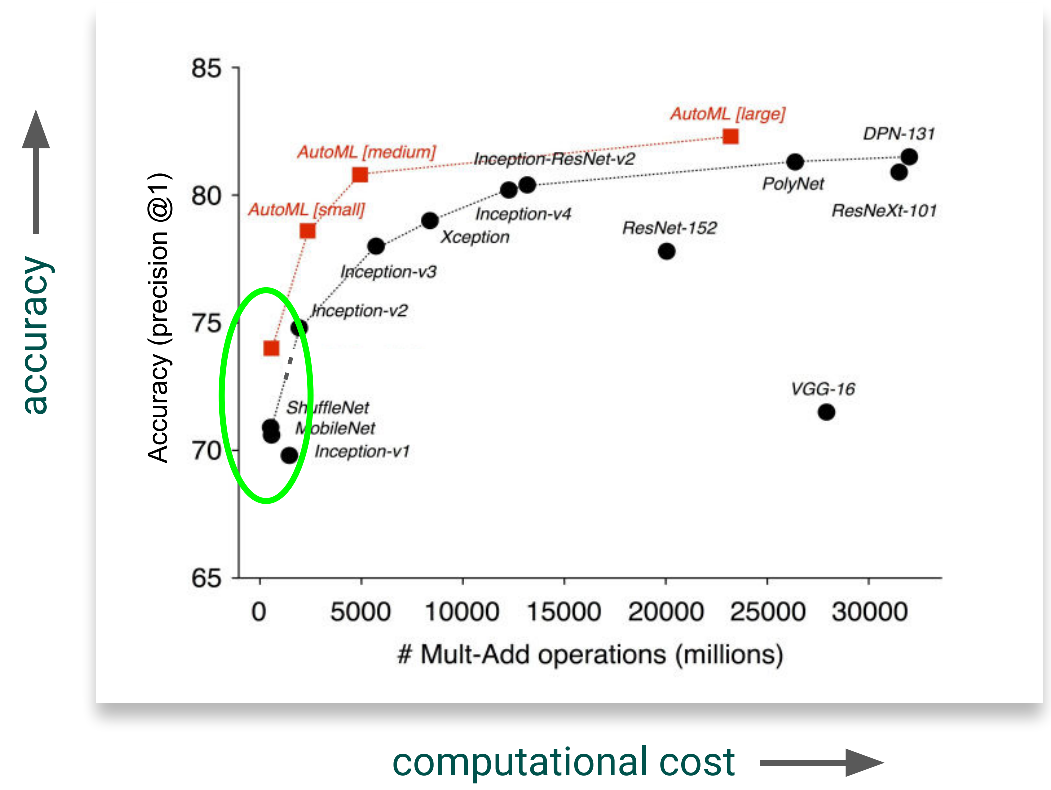
accuracy



computational cost

Learning Transferable Architectures for Scalable Image Recognition, Zoph et al. 2017, https://arxiv.org/abs/1707.07012

AutoML outperforms handcrafted models



Learning Transferable Architectures for Scalable Image Recognition, Zoph et al. 2017, https://arxiv.org/abs/1707.07012

CLOUD AUTOML ALPHA

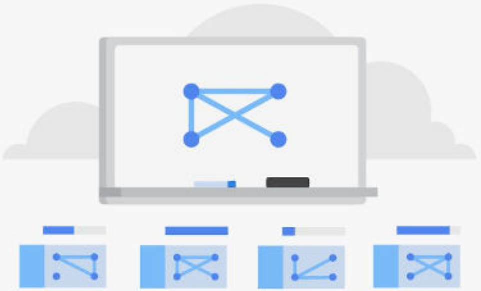
Train high quality custom machine learning models with minimum effort and machine learning expertise

REQUEST ACCESS

Train Custom Machine Learning Models

Cloud AutoML is a suite of Machine Learning products that enables developers with limited machine learning expertise to train high quality models by leveraging Google's state of the art transfer learning, and Neural Architecture Search technology.

AutoML Vision is the first product to be released. It is a simple, secure and flexible ML service that lets you train custom vision models for your own use cases. Soon, Cloud AutoML will release other services for all other major fields of AI.



https://cloud.google.com/automl/

More computational power needed

Deep learning is transforming how we design computers

Special computation properties

reduced precision ok

about 1.2
× about 0.6
about 0.7

NOT

~~1.21042~~
~~× 0.61127~~
~~0.73989343~~

Special computation properties

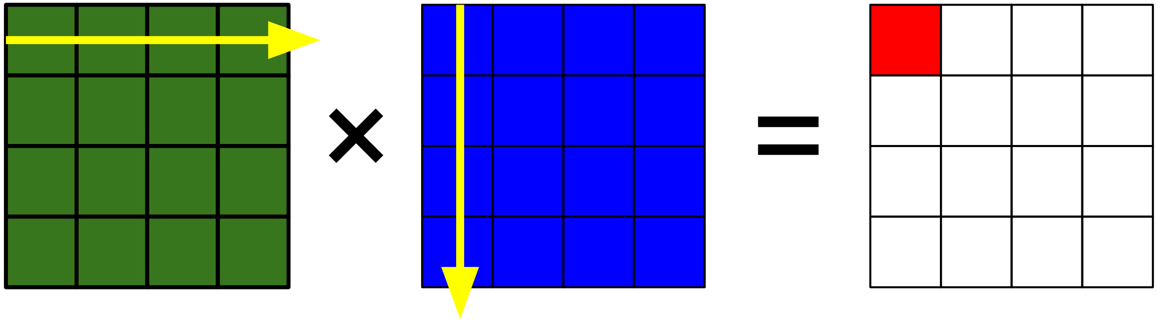
reduced precision ok

about 1.2
× about 0.6
about 0.7

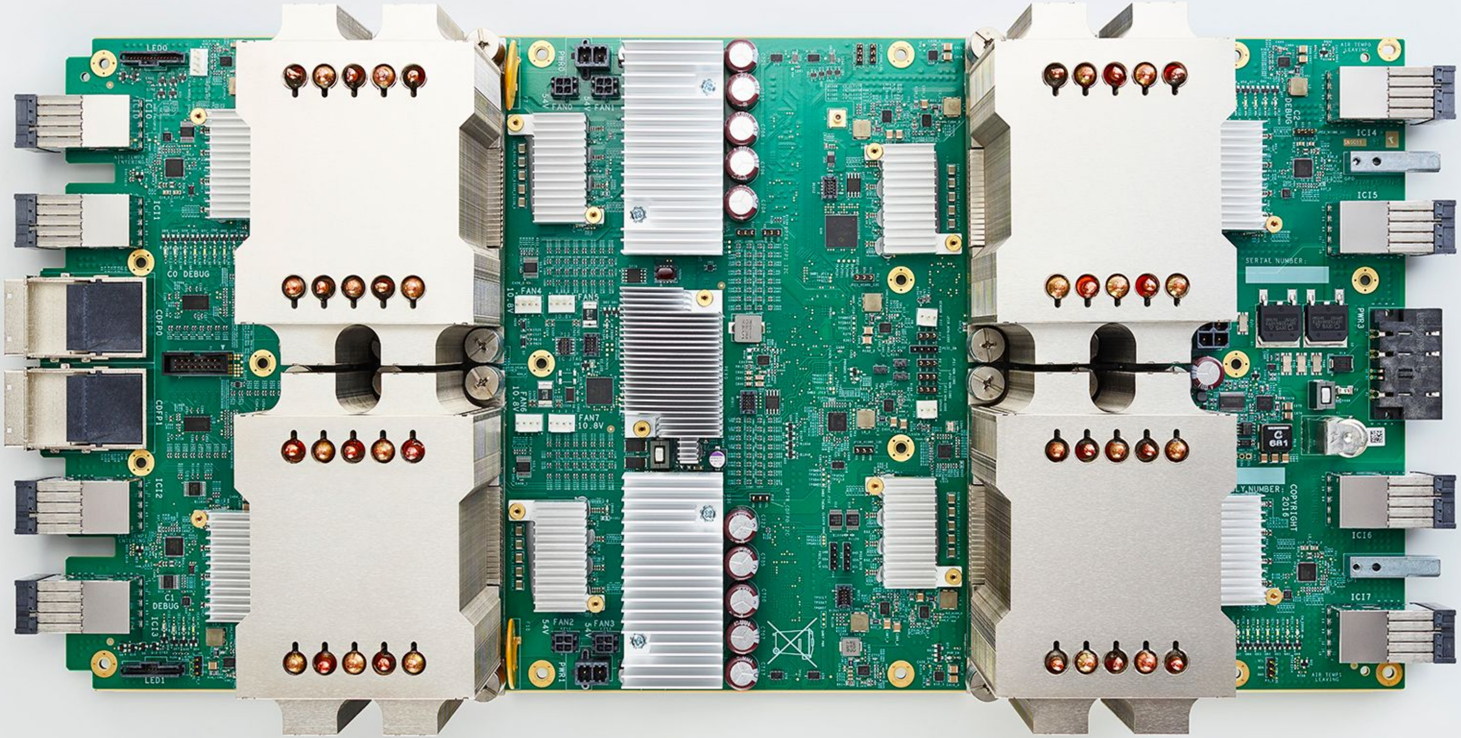
NOT

~~1.21042~~
~~× 0.61127~~
~~0.73989343~~

handful of specific operations

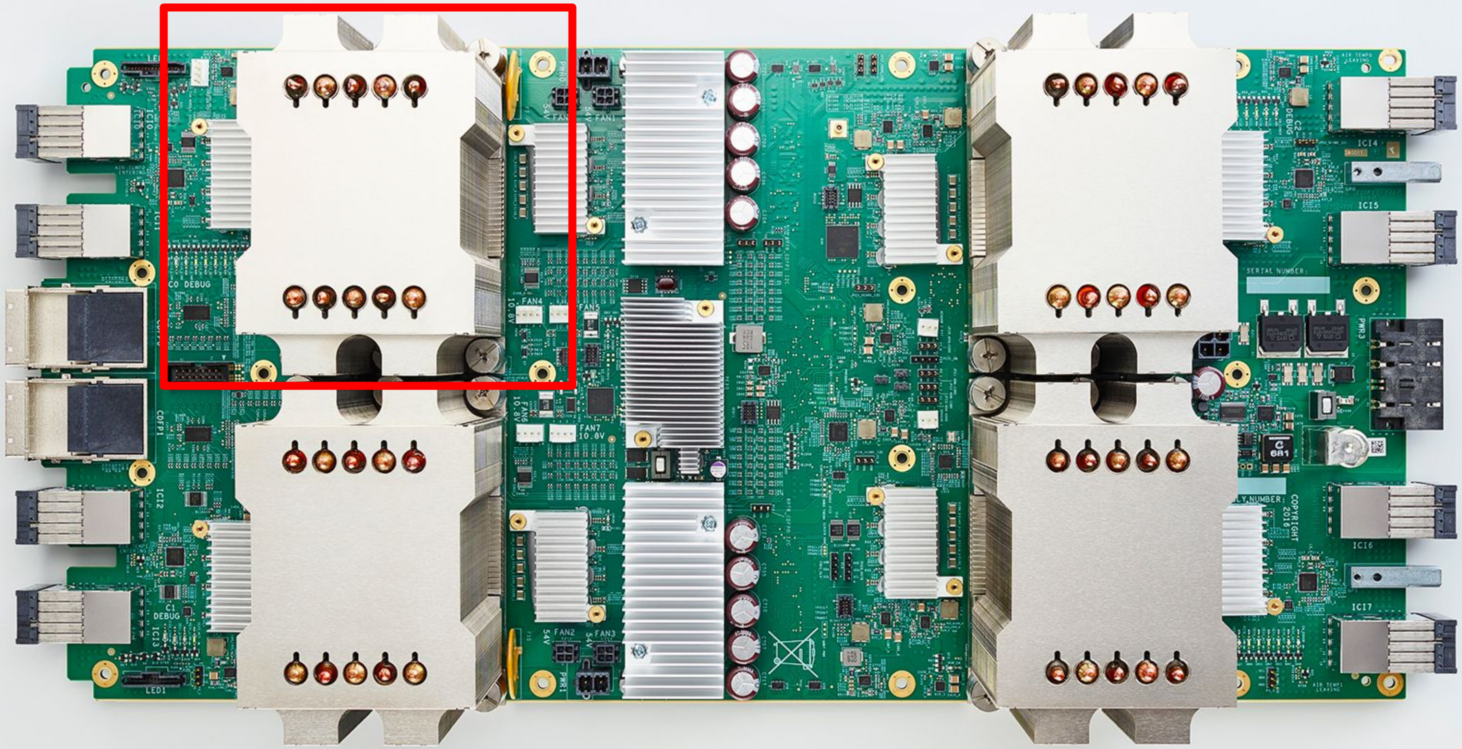


Tensor Processing Unit v2



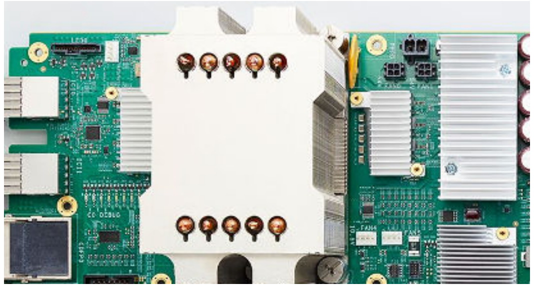
Google-designed device for neural net **training** and **inference**

Tensor Processing Unit v2

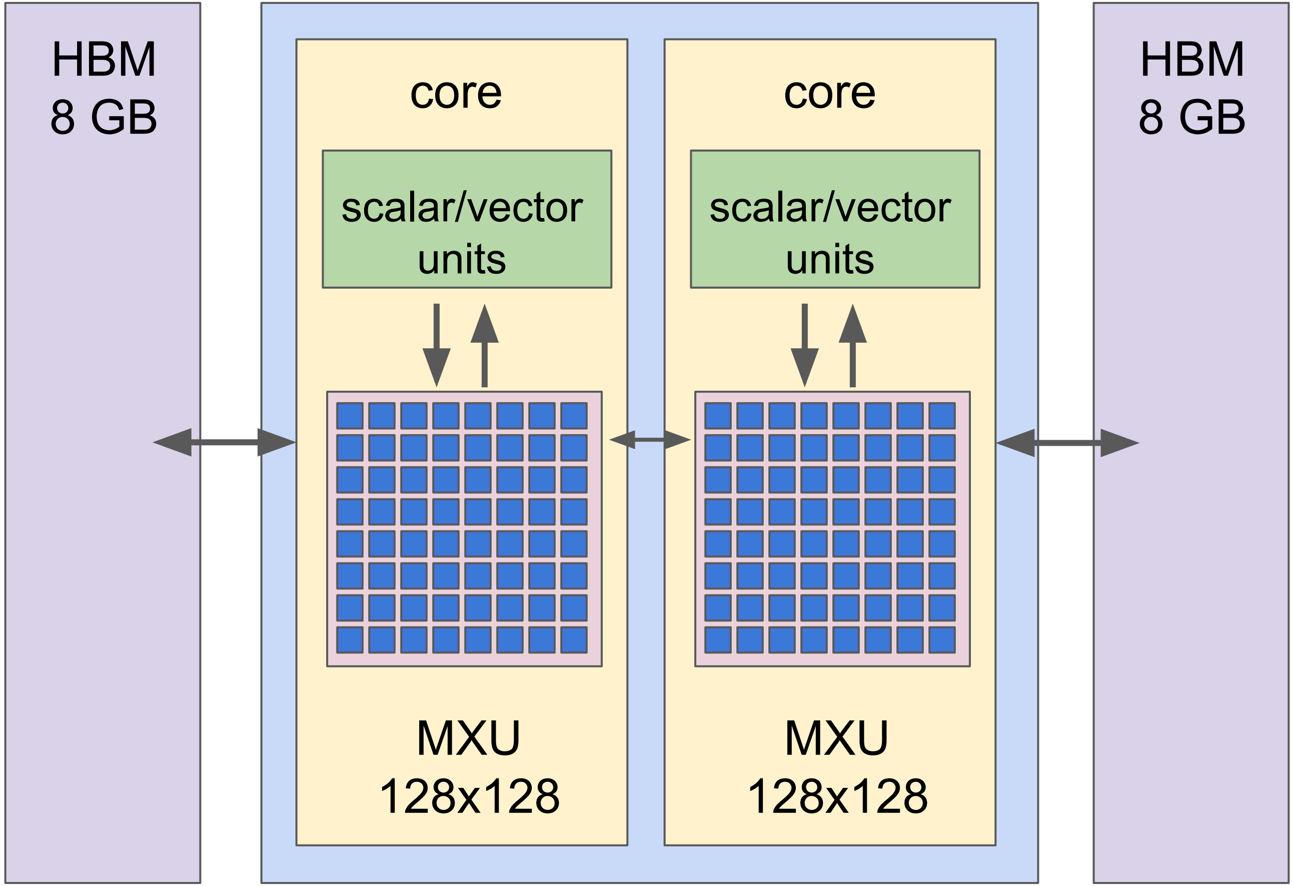


Google-designed device for neural net **training** and **inference**

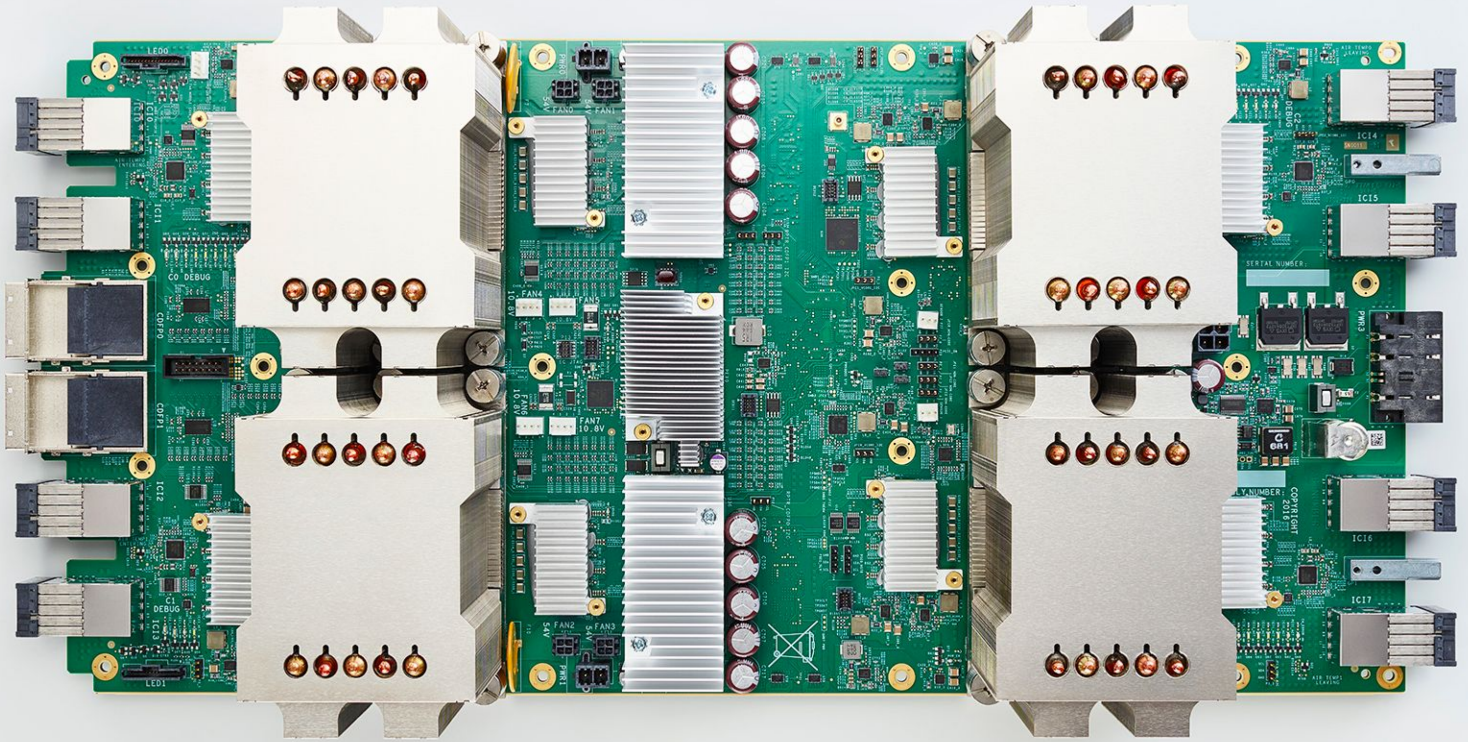
TPUv2 Chip



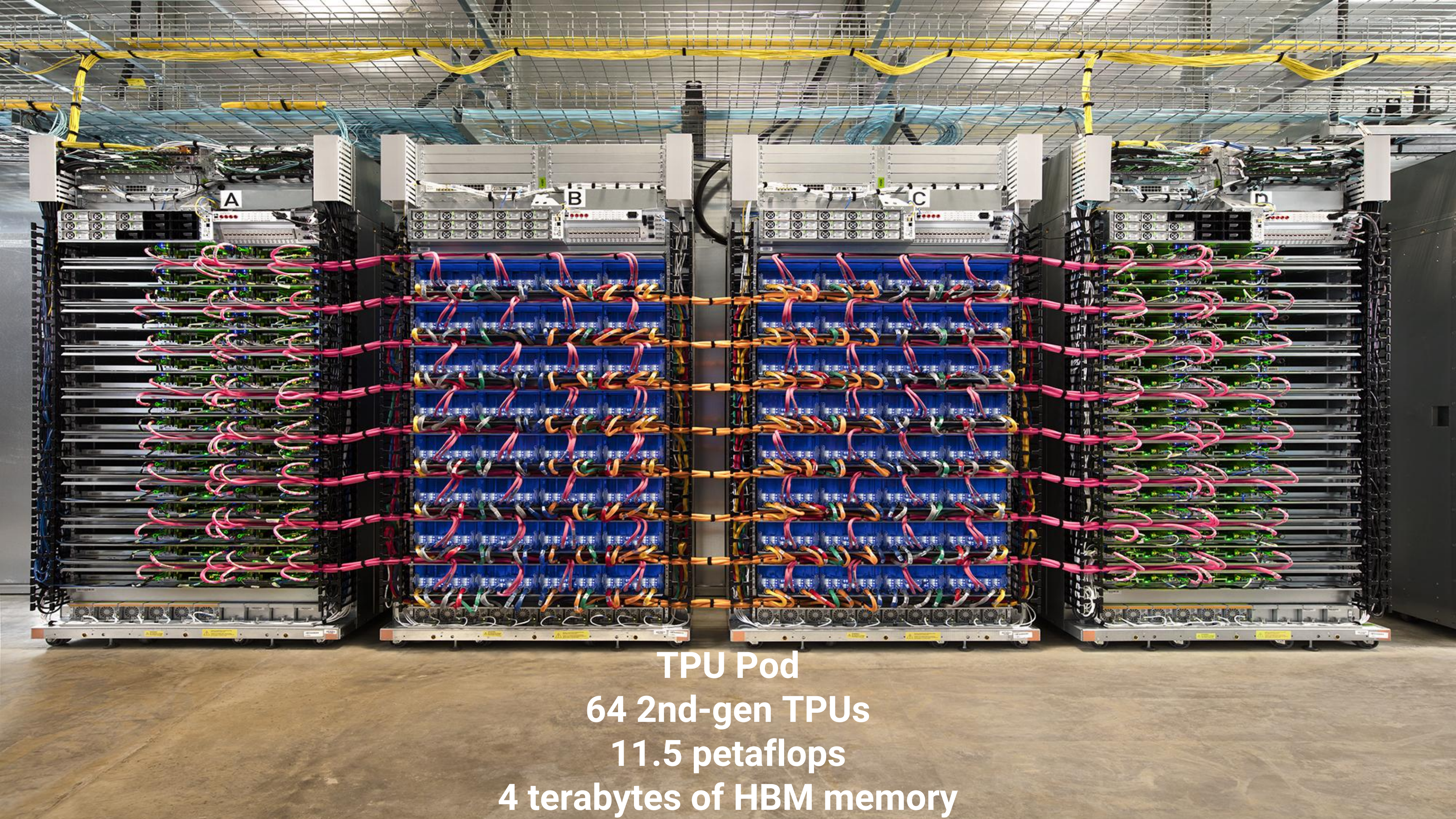
- 16 GB of HBM
- 600 GB/s mem BW
- Scalar/vector units: 32b float
- MXU: 32b float accumulation but reduced precision for multipliers
- 45 TFLOPS



Tensor Processing Unit v2



- 180 teraflops of computation, 64 GB of HBM memory, 2400 GB/s mem BW
- Designed to be connected together into larger configurations

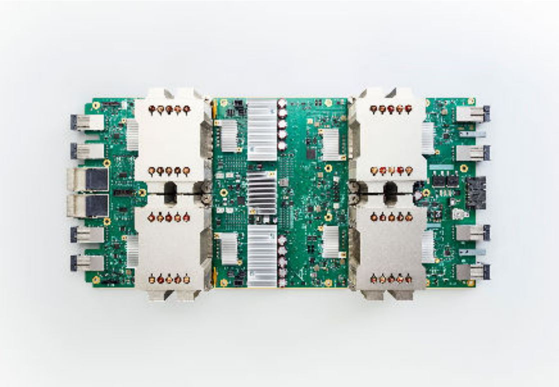


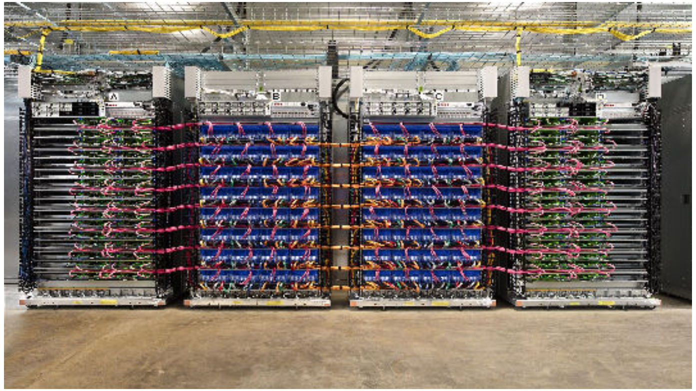
TPU Pod
64 2nd-gen TPUs
11.5 petaflops
4 terabytes of HBM memory

Programmed via TensorFlow

Same program will run w/only minor modifications on CPUs, GPUs, & TPUs

Same program scales via synchronous data parallelism without modification on TPU pods

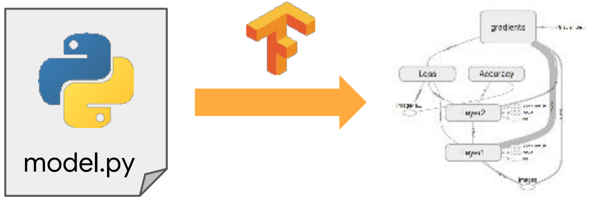




Accelerated Linear Algebra (XLA)

- JIT / AOT compiler for linear algebra
- Targets multiple backends, e.g. CPUs, GPUs, and TPUs
- Compiler, runtime, and accelerator-specific optimizer
- Compiler plus CPU and GPU backends open-sourced as part of TensorFlow

The life of a neural network:



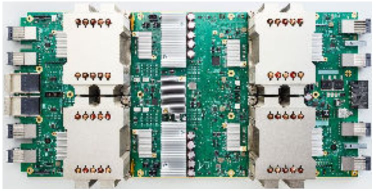
TF Estimator code

TF Graph

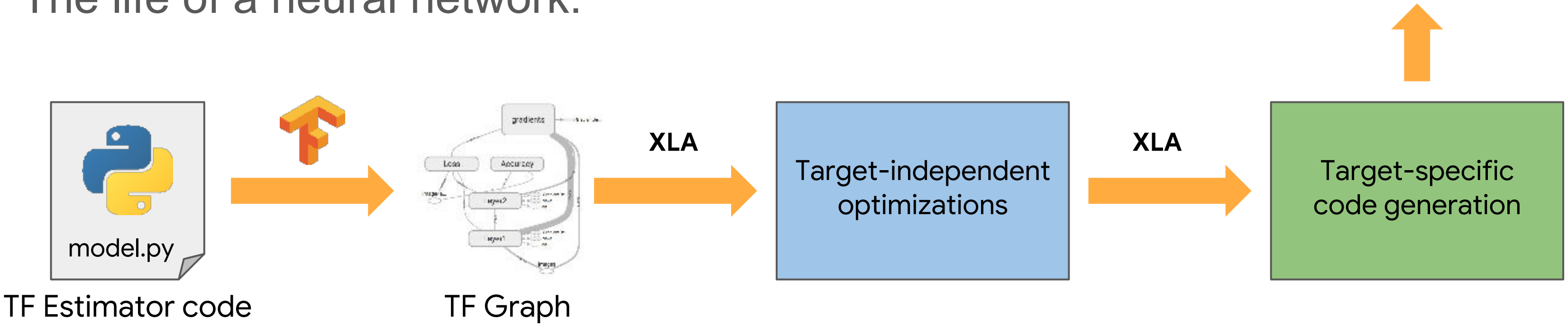
github.com/tensorflow/tensorflow/tree/master/tensorflow/compiler

Accelerated Linear Algebra (XLA)

- JIT / AOT compiler for linear algebra
- Targets multiple backends, e.g. CPUs, GPUs, and TPUs
- Compiler, runtime, and accelerator-specific optimizer
- Compiler plus CPU and GPU backends open-sourced as part of TensorFlow



The life of a neural network:



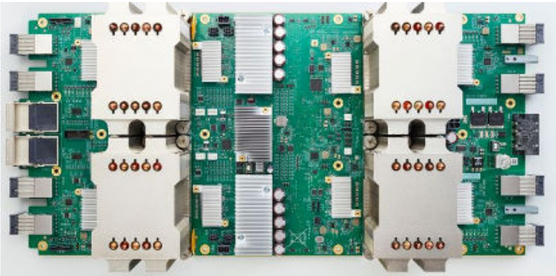
github.com/tensorflow/tensorflow/tree/master/tensorflow/compiler

Google Cloud Platform Blog

Cloud TPU machine learning accelerators now available in beta

Monday, February 12, 2018

Cloud TPU - host w/180 TFLOPS TPUv2 device attached



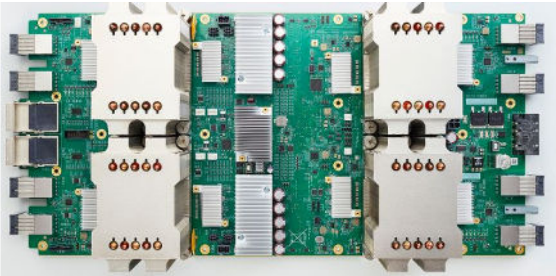
cloudplatform.googleblog.com/2018/02/Cloud-TPU-machine-learning-accelerators-now-available-in-beta.html

Google Cloud Platform Blog

Cloud TPU machine learning accelerators now available in beta

Monday, February 12, 2018

Cloud TPU - host w/180 TFLOPS TPUv2 device attached



"Since working with Google Cloud TPUs, we've been extremely impressed with their speed—what could normally take days can now take hours."
— Anantha Kancherla, Head of Software, Self-Driving Level 5, Lyft

"We found that moving TensorFlow workloads to TPUs has boosted our productivity by greatly reducing both the complexity of programming new models and the time required to train them."
— Alfred Spector, Chief Technology Officer, Two Sigma

cloudplatform.googleblog.com/2018/02/Cloud-TPU-machine-learning-accelerators-now-available-in-beta.html

TPUs run a wide & growing variety of open-source reference models

- **Image Classification**
  - ResNet 50/101/152/200, Inception v2/v3/v4, MobileNet, SqueezeNet, DenseNet
- **Object Detection**
  - RetinaNet
- **Machine translation, language modeling, sentiment analysis**
  - Transformer

Coming soon:

- **AmoebaNet that achieves 80% top-1 ImageNet validation accuracy**
  - Architecture discovered through evolutionary search on TPU (arxiv.org/abs/1802.01548)
- **Transformer-Based Speech Recognition**
  - Preview in Tensor2Tensor today
- **DeepVariant**
  - High-accuracy variant calling for genomic sequencing
- **Transformer-Based Image Generation**

https://github.com/tensorflow/tpu/

Some TPU Success Stories

Internal search ranking model training:

14.2X: ~9 hours on 1/4 pod vs. ~132 hours on 275 high end CPU machines

Internal image model training:

9.8X: ~22 hours on 1/4 pod vs. ~216 hours on previous production setup

WaveNet production model inference:

Generates speech at **20X real time**

Some TPU Success Stories (December 2017)

Resnet-50 to >76% accuracy:

1402 minutes on single TPUv2 device

45 minutes on 1/2 pod (32 TPUv2 devices)

Resnet-50 to 75% accuracy:

22 minutes on full pod (64 TPUv2 devices)

same code,
no special tricks

Some TPU Success Stories (today)

Resnet-50 to >76% accuracy:

~~1402~~ **785 minutes** on single TPUv2 device

~~45~~ **24.5 minutes** on 1/2 pod (32 TPUv2 devices)

Resnet-50 to 75% accuracy:

~~22~~ **12.2 minutes** on full pod (64 TPUv2 devices)

same code,
no special tricks

Some TPU Success Stories (today)

Resnet-50 to >76% accuracy:

~~1402~~ **785 minutes** on single TPUv2 device

~~45~~ **24.5 minutes** on 1/2 pod (32 TPUv2 devices)

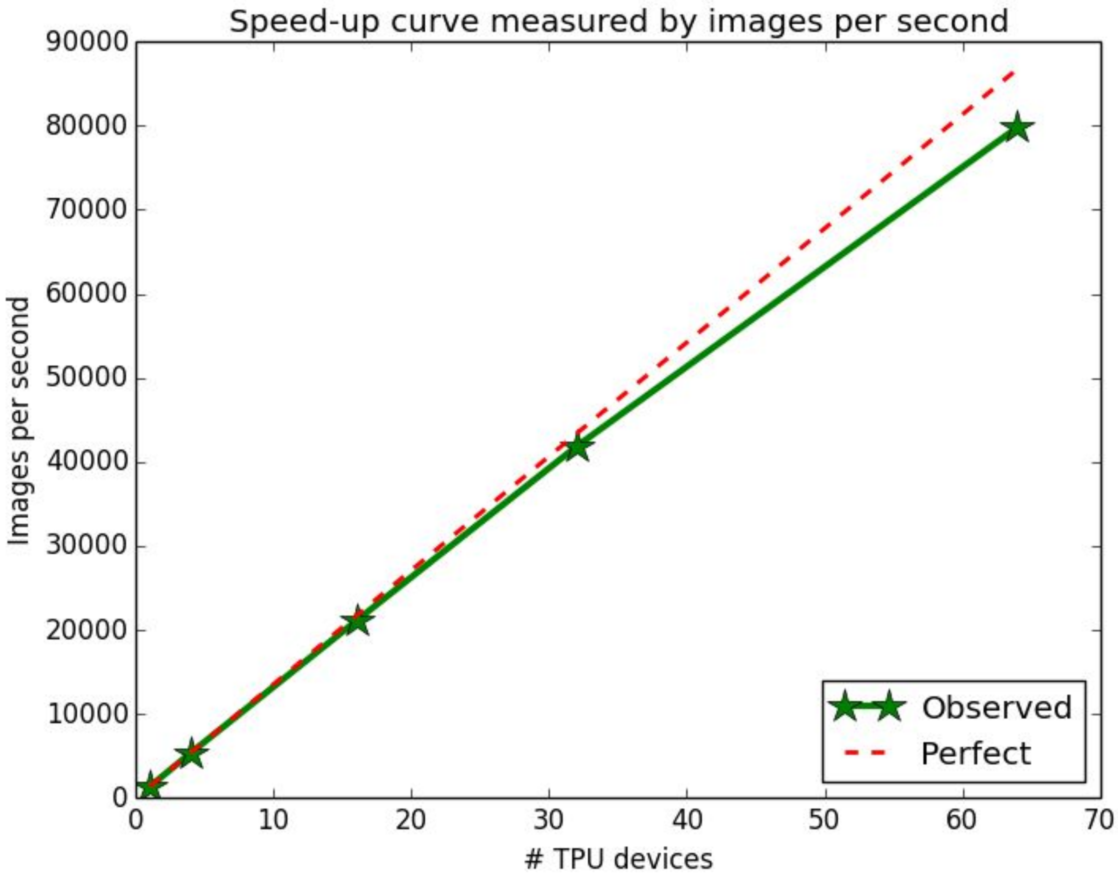
same code,
no special tricks

Resnet-50 to 75% accuracy:

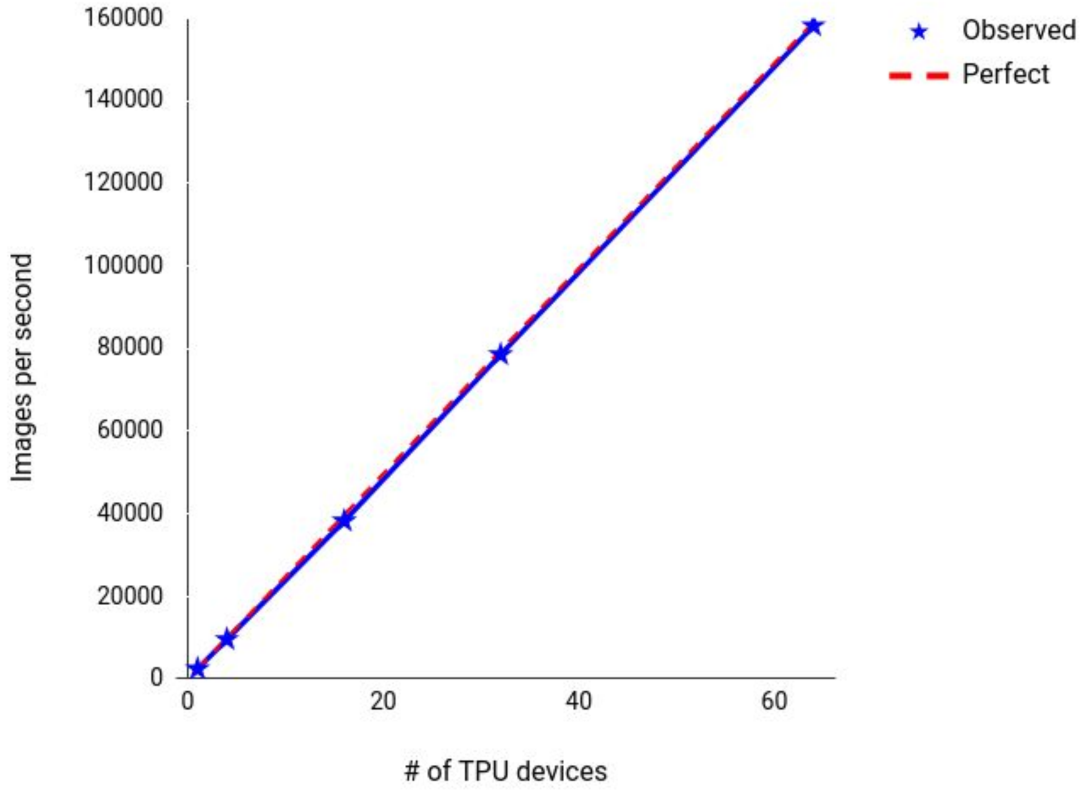
~~22~~ **12.2 minutes** on full pod (64 TPUv2 devices)

ImageNet training epoch (1.2M images) every ~8 seconds

TPU Scaling for ResNet-50 (December 2017)



TPU Scaling for ResNet-50 (today)

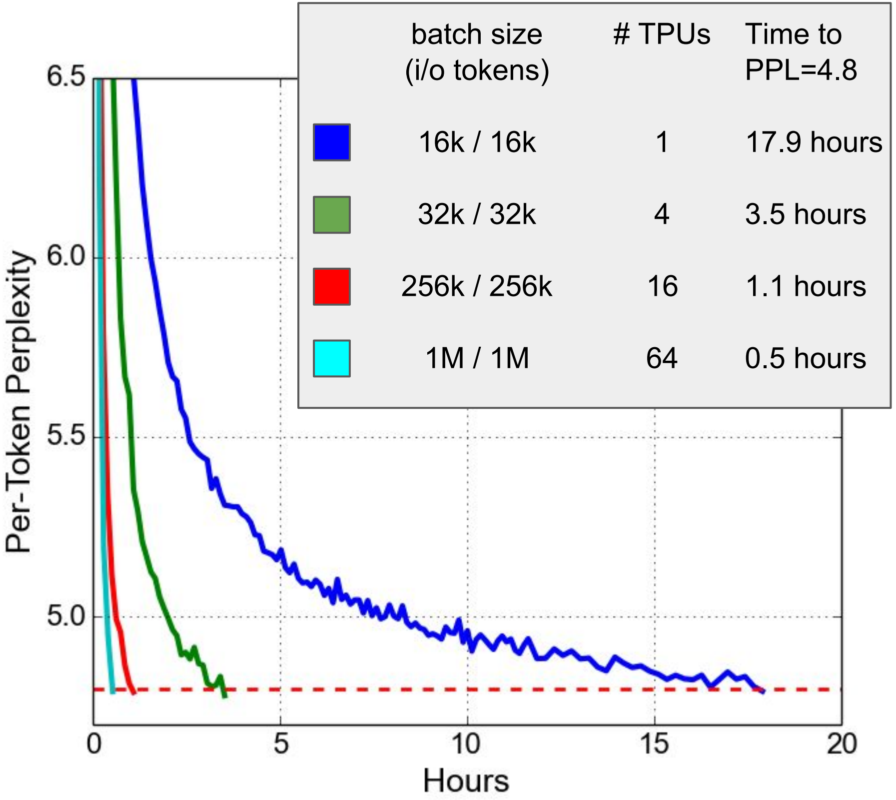


More than just ImageNet

Transformer model from "Attention is All You Need"
(2017 A. Vaswani et. al., NIPS 2017)

WMT'14 English-German translation task

Adam optimizer - same learning rate schedule across configurations



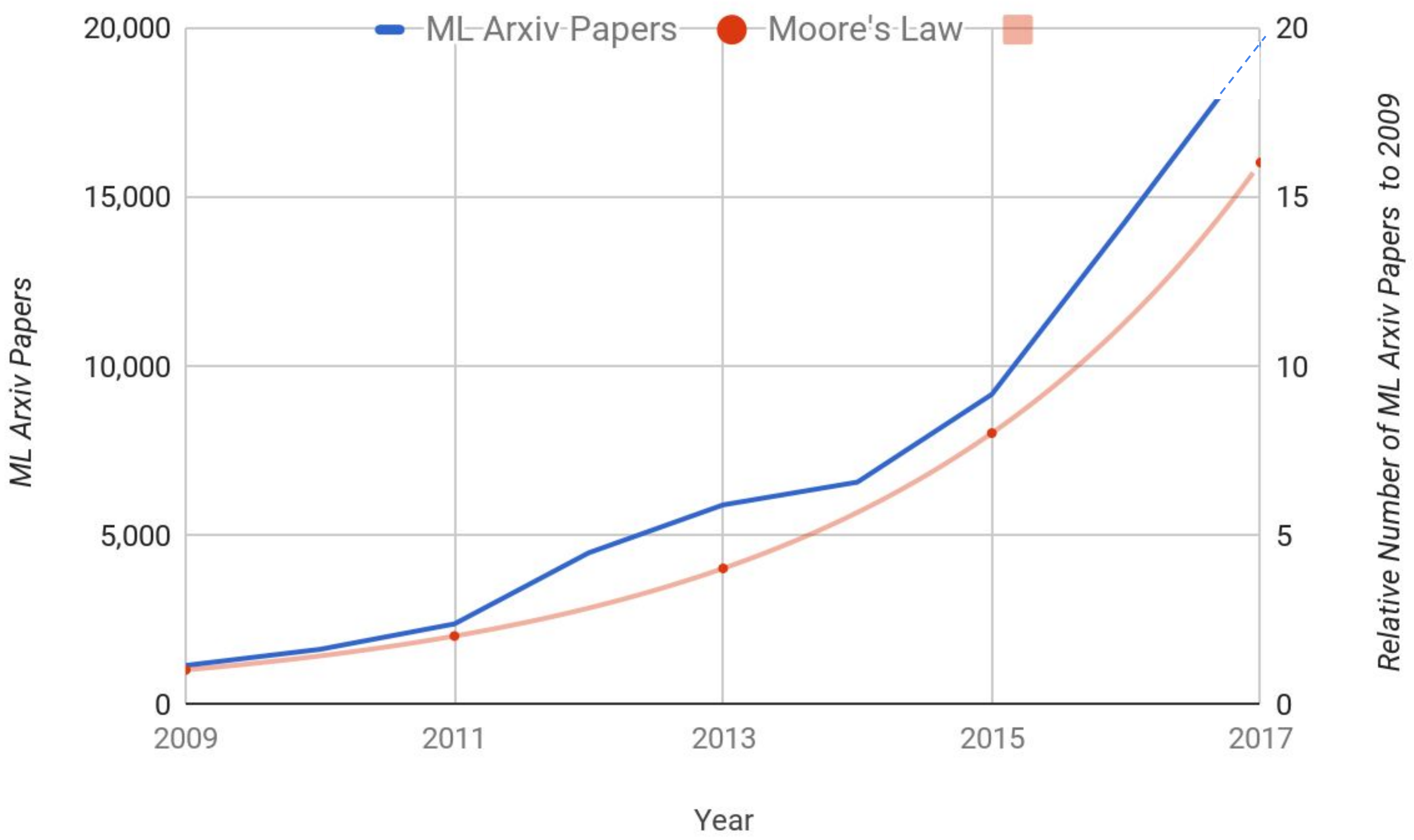
TensorFlow
RESEARCH CLOUD

1000 **Cloud TPUs** available **for free** to top researchers who are committed to **open machine learning research**

We’re excited to see what researchers will do with much more computation!
TFRC signup: g.co/tpusignup

What should we build in future ML accelerators?

ML Arxiv Papers per Year



If you start an ASIC machine learning accelerator design today, ...

Starts to get deployed into production in ~2 years

Must remain relevant through ~5 years from now

Can We See The Future Clearly Enough?
What should we bet on?

Some Example Questions

Precision:

Will very-low precision training (1-4 bit weights, 1-4 bit activations) work in general across all problems we care about?

Sparsity and embeddings: How should we handle:

Dynamic routing like the sparsely-gated Mixture of Experts work (ICLR'17)

Very large embeddings for some problems (e.g. 1B items x 1000D)

Batch size:

Should we build machines for very large batch sizes? Or batch size 1?

Training algorithms:

Will SGD-like algorithms remain the dominant training paradigm?

Or will large-batch second-order methods like K-FAC be better?

Machine Learning for Systems

Learning Should Be Used Throughout our Computing Systems

Traditional low-level systems code (operating systems, compilers, storage systems) **does not** make extensive use of machine learning today

This should change!

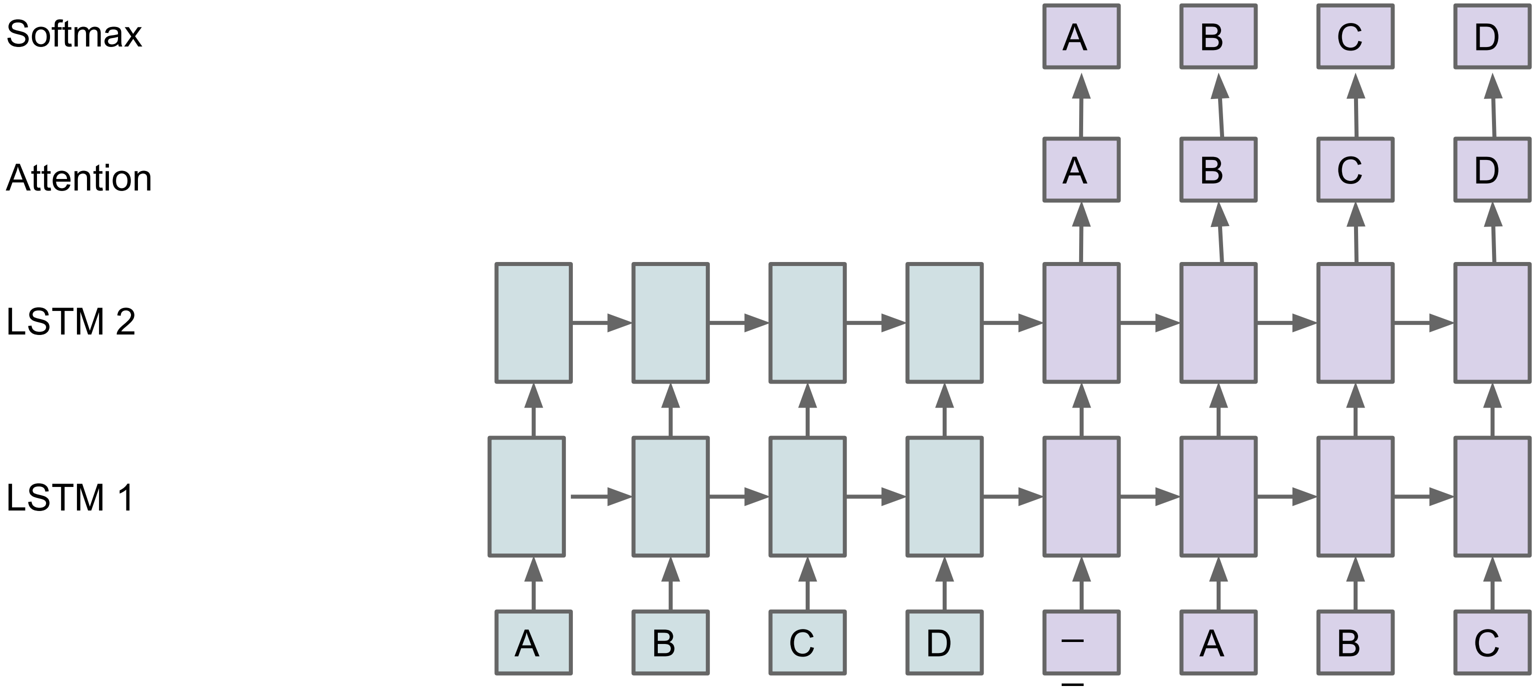
A few examples and some opportunities...

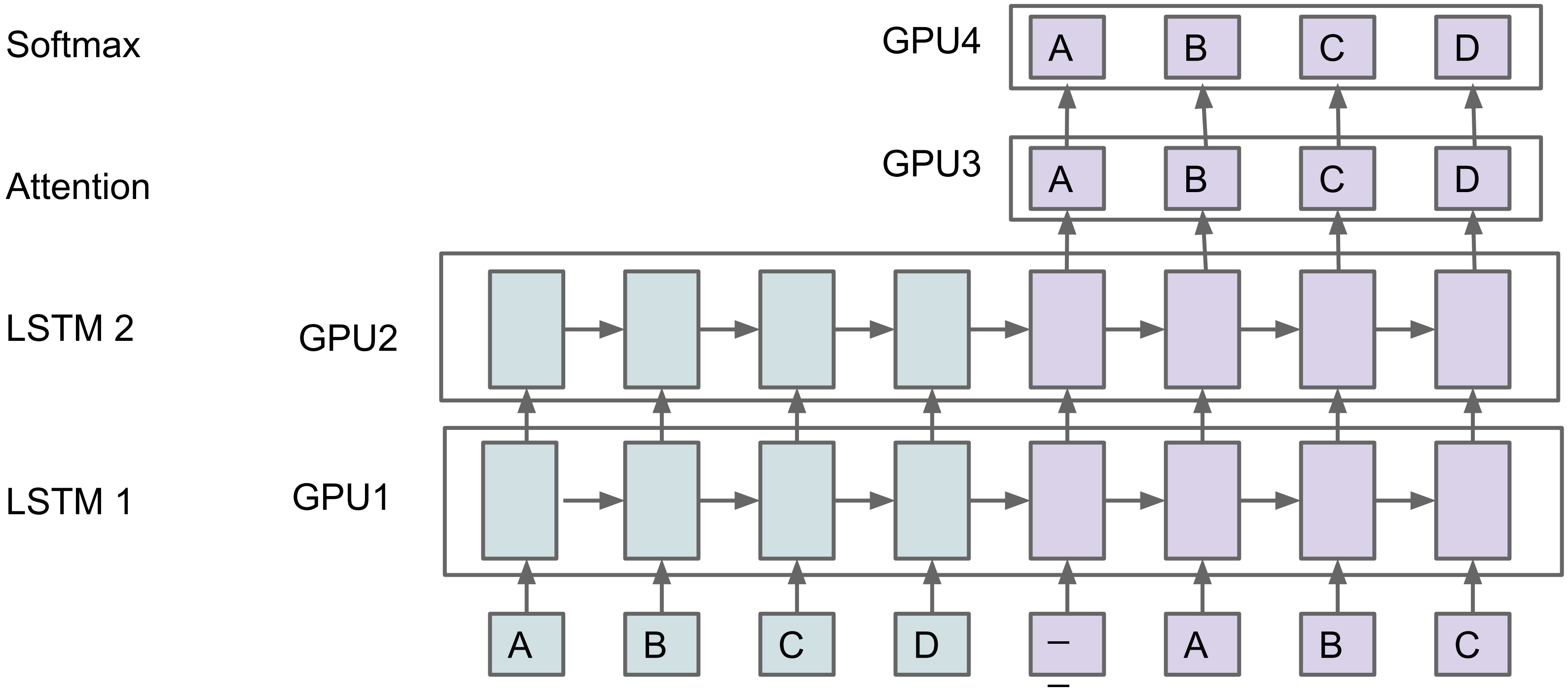
Machine Learning for
Higher Performance Machine Learning
Models

For large models, model parallelism is important

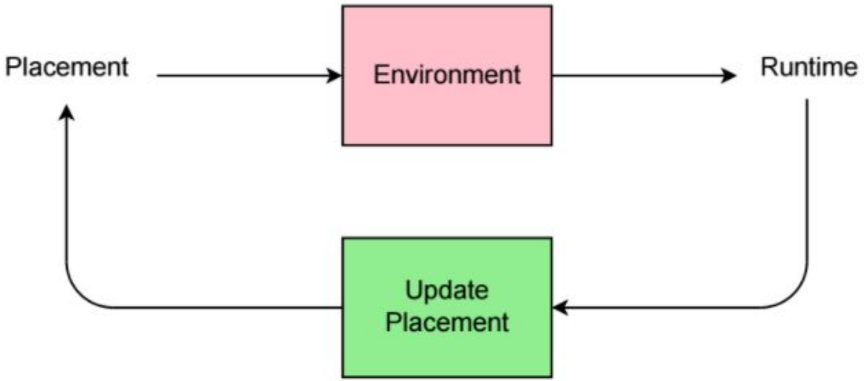
For large models, model parallelism is important

But getting good performance given multiple computing devices is non-trivial and non-obvious


Softmax
Attention
LSTM 2
LSTM 1
A
B
C
D
A
B
C
D
A
B
C
D
_
A
B
C


Softmax
Attention
LSTM 2
LSTM 1
GPU4
GPU3
GPU2
GPU1
A
B
C
D
A
B
C
D
A
B
C
D
_
A
B
C

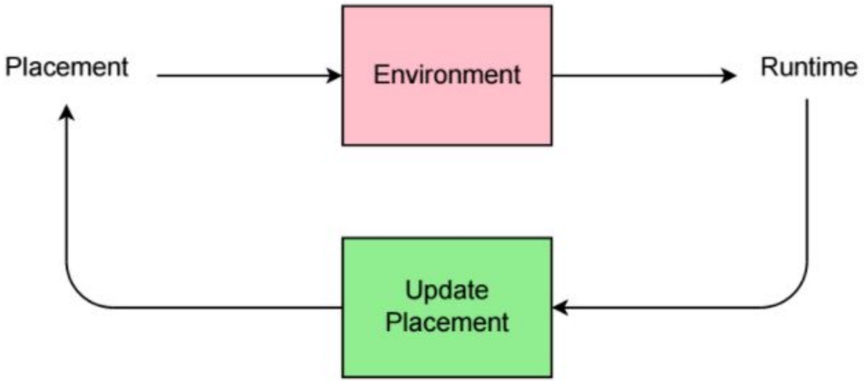
Reinforcement Learning for Higher Performance Machine Learning Models



Device Placement Optimization with Reinforcement Learning,
Azalia Mirhoseini, Hieu Pham, Quoc Le, Mohammad Norouzi, Samy Bengio, Benoit Steiner, Yuefeng Zhou, Naveen Kumar, Rasmus Larsen, and Jeff Dean, ICML 2017, arxiv.org/abs/1706.04972

Reinforcement Learning for Higher Performance Machine Learning Models

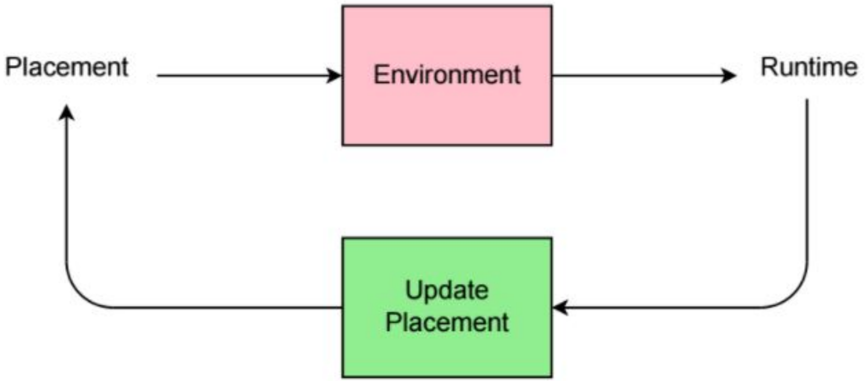
Placement model (trained via RL) gets graph as input + set of devices, outputs device placement for each graph node



Device Placement Optimization with Reinforcement Learning,
Azalia Mirhoseini, Hieu Pham, Quoc Le, Mohammad Norouzi, Samy Bengio, Benoit Steiner, Yuefeng Zhou, Naveen Kumar, Rasmus Larsen, and Jeff Dean, ICML 2017, arxiv.org/abs/1706.04972

Reinforcement Learning for Higher Performance Machine Learning Models

Placement model (trained via RL) gets graph as input + set of devices, outputs device placement for each graph node

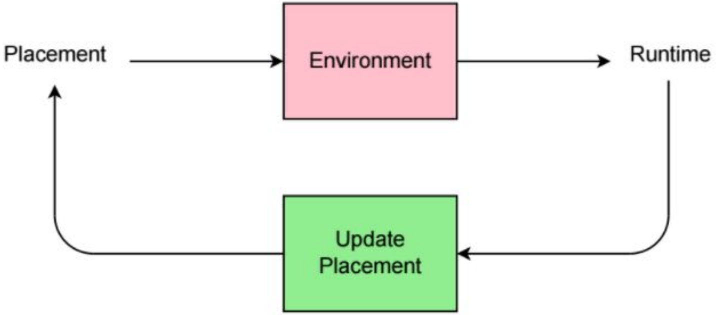


Measured time per step gives RL reward signal

Device Placement Optimization with Reinforcement Learning,
Azalia Mirhoseini, Hieu Pham, Quoc Le, Mohammad Norouzi, Samy Bengio, Benoit Steiner, Yuefeng Zhou, Naveen Kumar, Rasmus Larsen, and Jeff Dean, ICML 2017, arxiv.org/abs/1706.04972

Device Placement with Reinforcement Learning

Placement model (trained via RL) gets graph as input + set of devices, outputs device placement for each graph node



Measured time per step gives RL reward signal

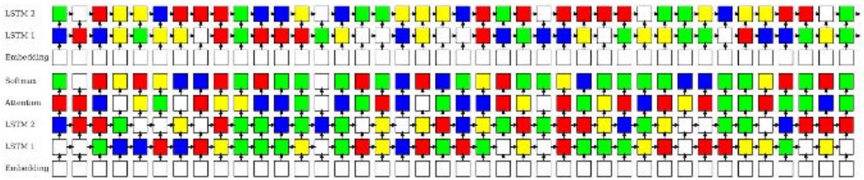


Figure 4. RL-based placement of Neural MT graph. Above: encoder, Below: decoder. Devices are denoted by colors, where the transparent color represents an operation on a CPU and each other unique color represents a different GPU. This placement achieves an improvement of 19.3% in running time compared to the fine-tuned hand-crafted placement.

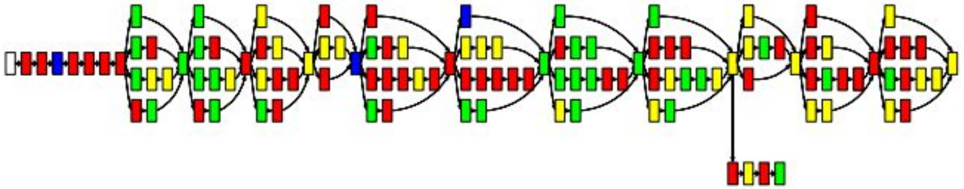


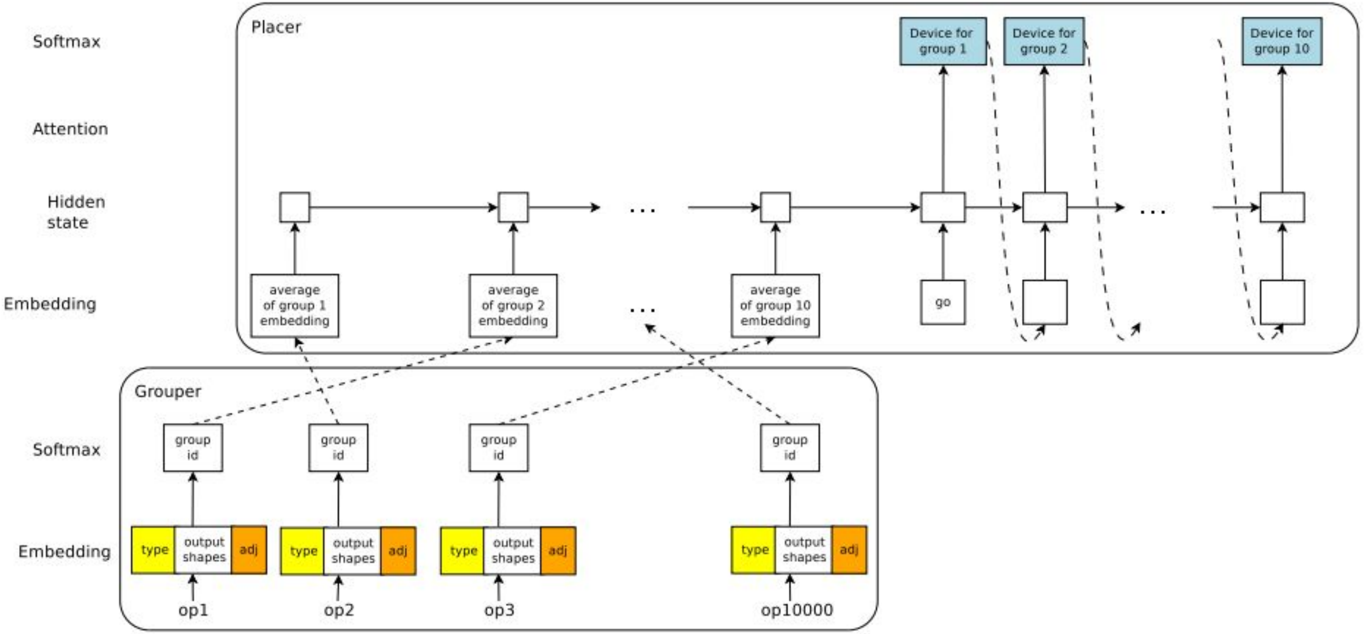
Figure 5. RL-based placement of Inception-V3. Devices are denoted by colors, where the transparent color represents an operation on a CPU and each other unique color represents a different GPU. RL-based placement achieves the improvement of 19.7% in running time compared to expert-designed placement.

+19.3% faster vs. expert human for neural translation model

+19.7% faster vs. expert human for InceptionV3 image model

Device Placement Optimization with Reinforcement Learning,
Azalia Mirhoseini, Hieu Pham, Quoc Le, Mohammad Norouzi, Samy Bengio, Benoit Steiner, Yuefeng Zhou, Naveen Kumar, Rasmus Larsen, and Jeff Dean, ICML 2017, arxiv.org/abs/1706.04972

A Hierarchical Model for Device Placement



A Hierarchical Model for Device Placement,
Azalia Mirhoseini, Anna Goldie, Hieu Pham, Benoit Steiner, Quoc V. Le, and Jeff Dean, to appear in ICLR 2018,
openreview.net/forum?id=Hkc-TeZ0W

A Hierarchical Model for Device Placement

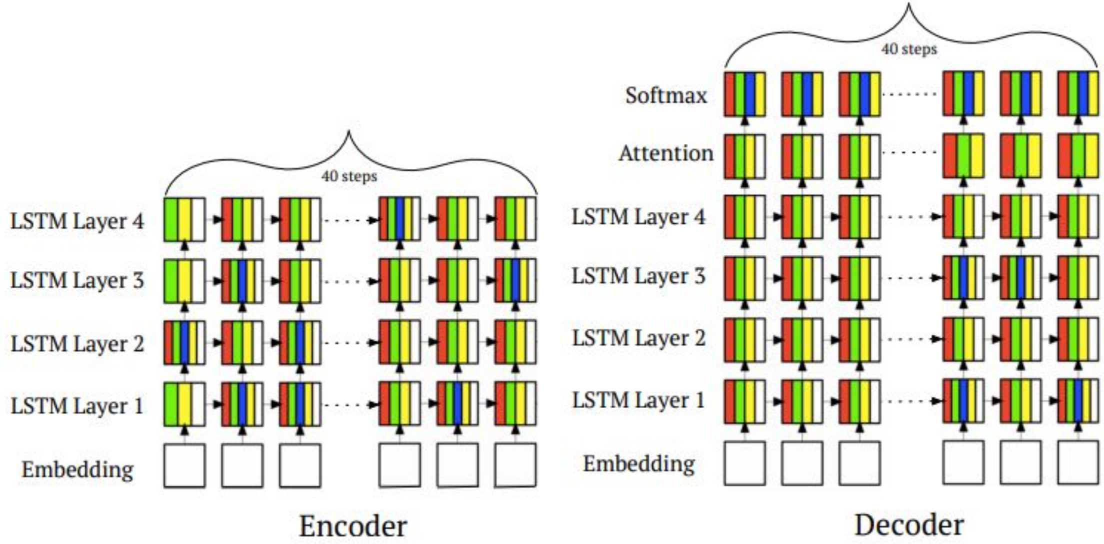


Figure 2: The Hierarchical Planner's placement of a NMT (4-layer) model. White denotes CPU and the four colors each represent one of the GPUs. Note that every step of every layer is allocated across multiple GPUs. This placement is 53.7% faster than that generated by a human expert.

+53.7% faster vs. expert human for neural machine translation model

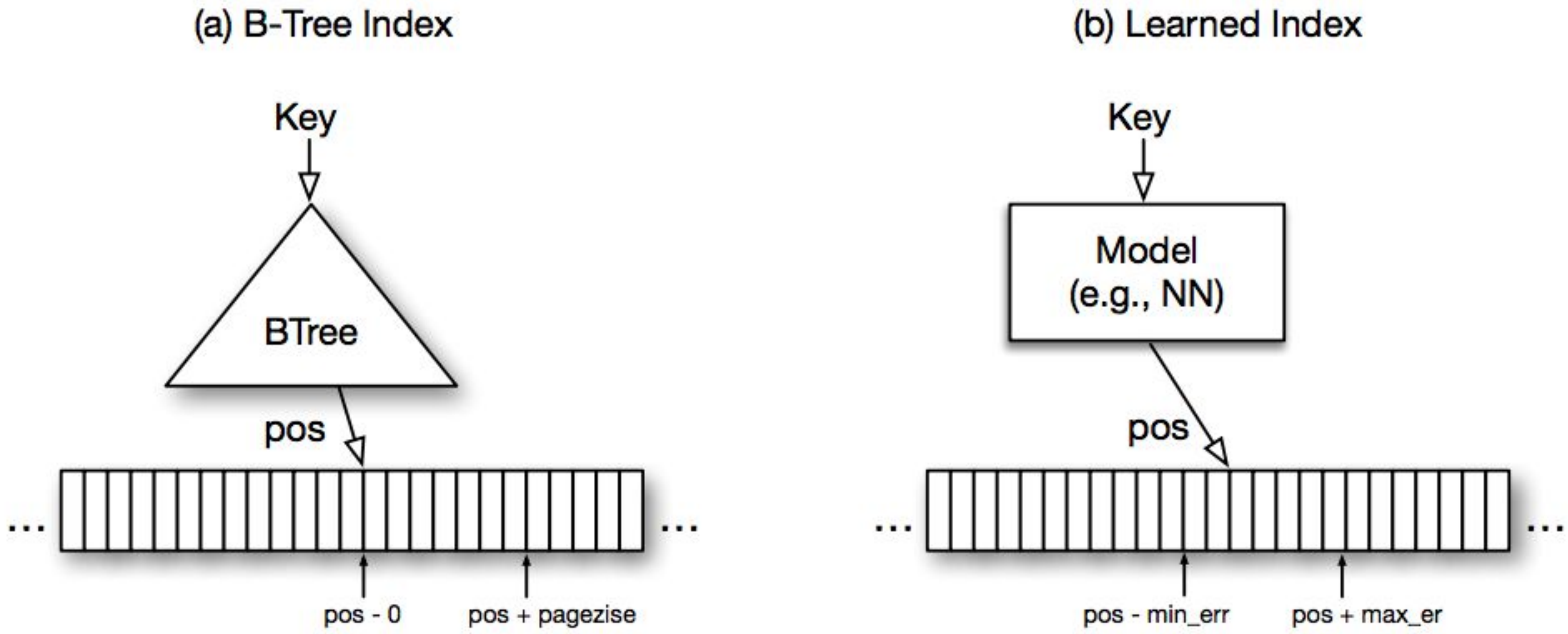
A Hierarchical Model for Device Placement,
Azalia Mirhoseini, Anna Goldie, Hieu Pham, Benoit Steiner, Quoc V. Le, and Jeff Dean, to appear in ICLR 2018,
openreview.net/forum?id=Hkc-TeZ0W

Learned Index Structures

not

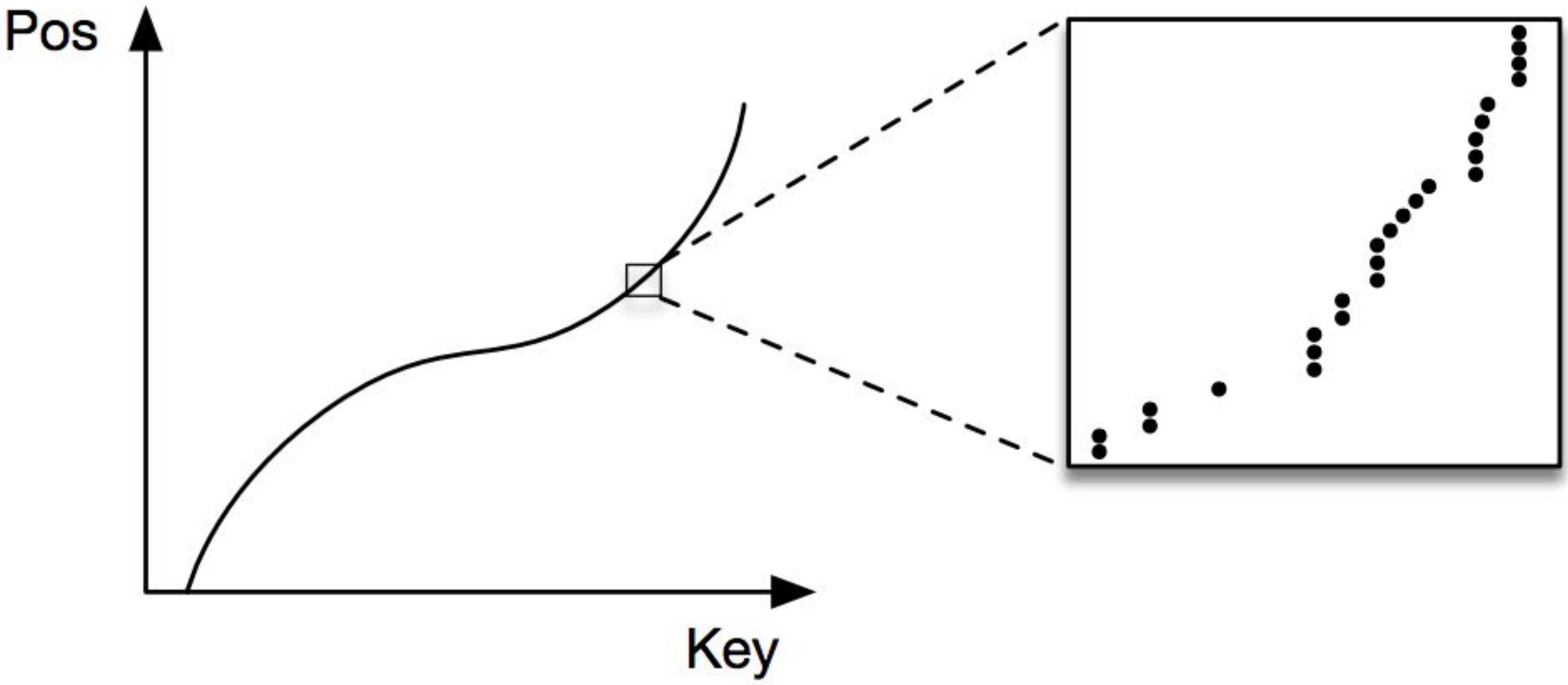
Conventional Index Structures

B-Trees are Models



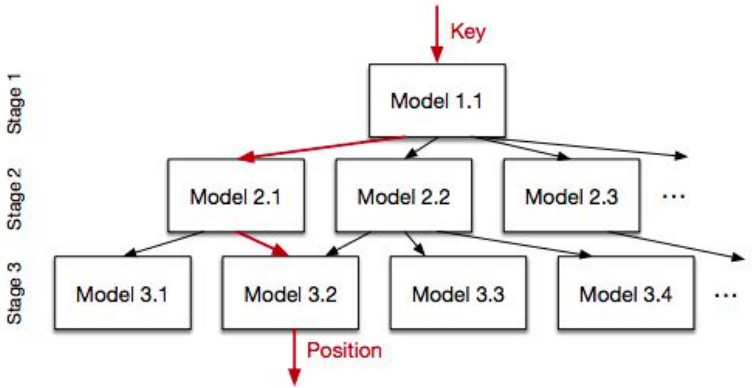
The Case for Learned Index Structures, Tim Kraska, Alex Beutel, Ed Chi, Jeffrey Dean & Neoklis Polyzotis, arxiv.org/abs/1712.01208

Indices as CDFs



The Case for Learned Index Structures, Tim Kraska, Alex Beutel, Ed Chi, Jeffrey Dean & Neoklis Polyzotis, arxiv.org/abs/1712.01208

Does it Work?



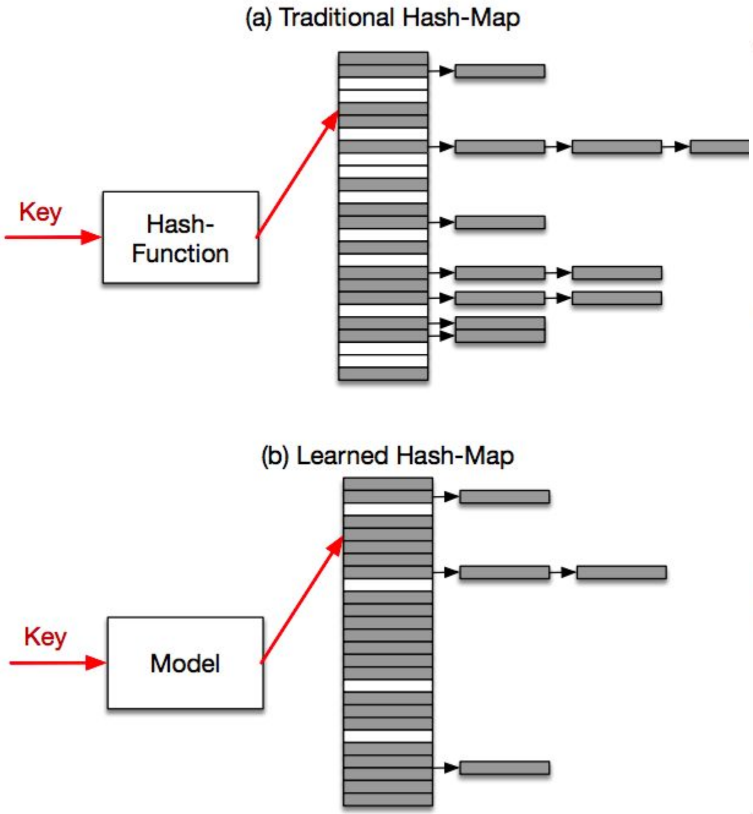
Index of 200M web service log records

Type	Config	Lookup time	Speedup vs. Btree	Size (MB)	Size vs. Btree
BTree	page size: 128	260 ns	1.0X	12.98 MB	1.0X
Learned index	2nd stage size: 10000	222 ns	1.17X	0.15 MB	0.01X
Learned index	2nd stage size: 50000	**162 ns**	**1.60X**	**0.76 MB**	**0.05X**
Learned index	2nd stage size: 100000	144 ns	1.67X	1.53 MB	0.12X
Learned index	2nd stage size: 200000	126 ns	2.06X	3.05 MB	0.23X

60% faster at 1/20th the space, or 17% faster at 1/100th the space

The Case for Learned Index Structures, Tim Kraska, Alex Beutel, Ed Chi, Jeffrey Dean & Neoklis Polyzotis, arxiv.org/abs/1712.01208

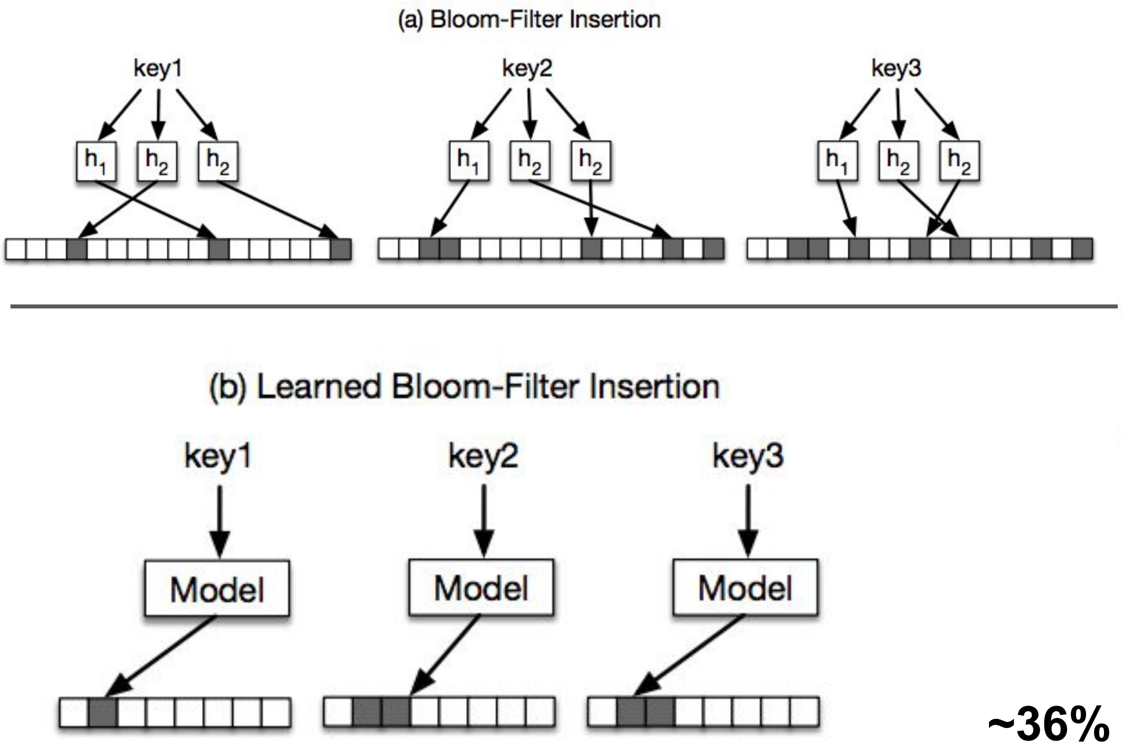
Hash Tables

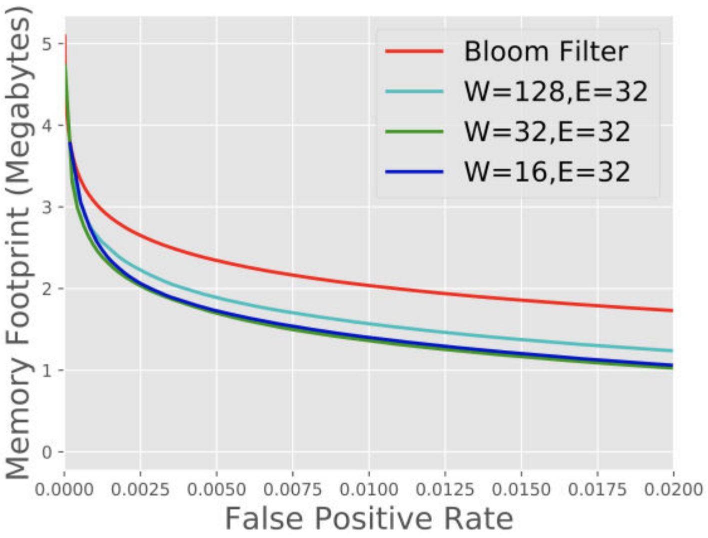


Dataset	Slots	Hash Type	Search Time (ns)	Empty Slots	Space Improvement
Map	75%	Model Hash	67	0.63GB (05%)	-20%
		Random Hash	52	0.80GB (25%)	
	100%	Model Hash	53	1.10GB (08%)	-27%
		Random Hash	48	1.50GB (35%)	
	125%	Model Hash	64	2.16GB (26%)	-6%
		Random Hash	49	2.31GB (43%)	
Web Log	75%	Model Hash	78	0.18GB (19%)	-78%
		Random Hash	53	0.84GB (25%)	
	100%	Model Hash	63	0.35GB (25%)	-78%
		Random Hash	50	1.58GB (35%)	
	125%	Model Hash	77	1.47GB (40%)	-39%
		Random Hash	50	2.43GB (43%)	
Log Normal	75%	Model Hash	79	0.63GB (20%)	-22%
		Random Hash	52	0.80GB (25%)	
	100%	Model Hash	66	1.10GB (26%)	-30%
		Random Hash	46	1.50GB (35%)	
	125%	Model Hash	77	2.16GB (41%)	-9%
		Random Hash	46	2.31GB (44%)	

The Case for Learned Index Structures, Tim Kraska, Alex Beutel, Ed Chi, Jeffrey Dean & Neoklis Polyzotis, arxiv.org/abs/1712.01208

Bloom Filters





Model is simple RNN
W is number of units in RNN layer
E is width of character embedding

~36% space improvement over Bloom Filter at same false positive rate

The Case for Learned Index Structures, Tim Kraska, Alex Beutel, Ed Chi, Jeffrey Dean & Neoklis Polyzotis, arxiv.org/abs/1712.01208

Where Else Could We Use Learning?

Computer Systems are Filled With Heuristics

Compilers, Networking code, Operating Systems, …

Heuristics have to work well “in general case”

Generally don’t adapt to actual pattern of usage

Generally don’t take into account available context

Anywhere We're Using Heuristics To Make a Decision!

Compilers: instruction scheduling, register allocation, loop nest parallelization strategies, …

Networking: TCP window size decisions, backoff for retransmits, data compression, ...

Operating systems: process scheduling, buffer cache insertion/replacement, file system prefetching, …

Job scheduling systems: which tasks/VMs to co-locate on same machine, which tasks to pre-empt, ...

ASIC design: physical circuit layout, test case selection, …

Anywhere We’ve Punted to a User-Tunable Performance Option!

Many programs have huge numbers of tunable command-line flags, usually not changed from their defaults

```
--eventmanager_threads=16
--bigtable_scheduler_batch_size=8
--mapreduce_merge_memory=134217728
--lexicon_cache_size=1048576
--storage_server_rpc_freelist_size=128
...
```

Meta-learn everything

ML:

- learning placement decisions
- learning fast kernel implementations
- learning optimization update rules
- learning input preprocessing pipeline steps
- learning activation functions
- learning model architectures for specific device types, or that are fast for inference on mobile device X, learning which pre-trained components to reuse, …

Computer architecture/datacenter networking design:

- learning best design properties by exploring design space automatically (via simulator)

Keys for Success in These Settings

(1) Having a numeric metric to measure and optimize
(2) Having a clean interface to easily integrate learning into all of these kinds of systems

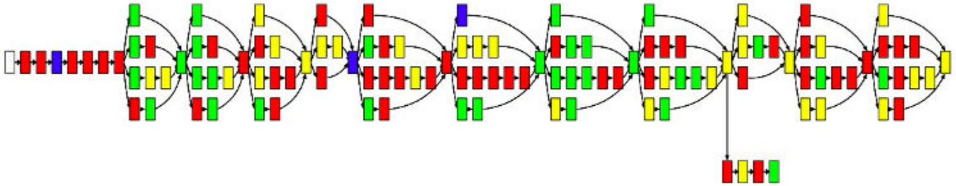
Current work: exploring APIs and implementations
Basic ideas:

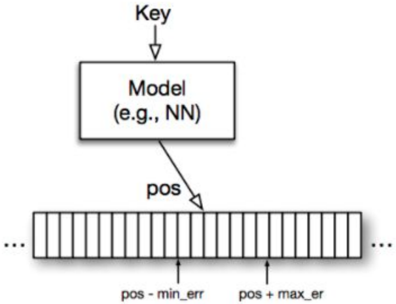
- Make a sequence of choices in some context
- Eventually get feedback about those choices
- Make this all work with very low overhead, even in distributed settings
- Support many implementations of core interfaces

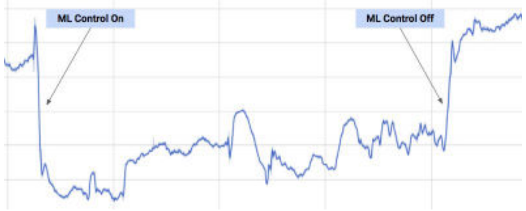
Conclusions

ML hardware is at its infancy. Even faster systems and wider deployment will lead to many more breakthroughs across a wide range of domains.

Learning in the core of all of our computer systems will make them better/more adaptive. There are many opportunities for this.







More info about our work at g.co/brain