ML
Commons

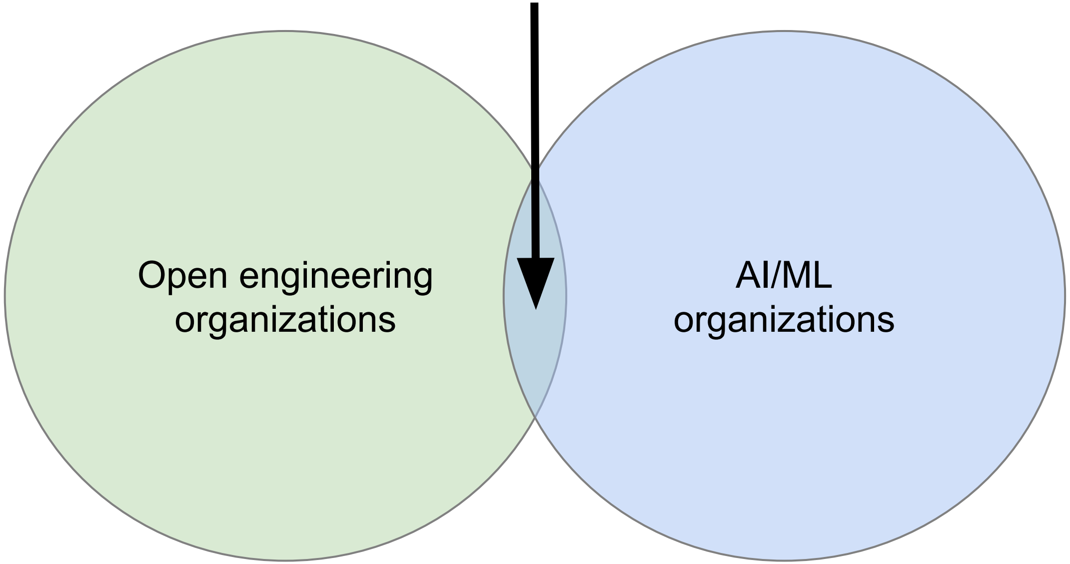
Challenges and Directions in ML System Performance: The MLPerf™ Story

David Kanter
Executive Director

September 6th, 2021

We want a **new open engineering organization** to create better ML for everyone

ML
C



MLCommons™ is a global community

Founding Members

Alibaba Group, AMD, arm, Baidu 百度, CenTaur Technology, cerebras, cisco, DELL EMC, d-MATRIX, Enflame, FACEBOOK AI, FUJITSU, FURIOSA, GIGABYTE, Google, GML GrAI Matter Labs, GRAPHCORE, groq, 地平线 Horizon Robotics, Hewlett Packard Enterprise, inspur, intel AI, KALRAY, LANDING AI, MEDIATEK, Microsoft, Myrtle.ai, Nettrix 宁畅, NVIDIA, oppo, Qualcomm, Red Hat, SambaNova Systems, SAMSUNG Exynos, SYNTIANT, Tenstorrent, vmware, VMind, XILINX

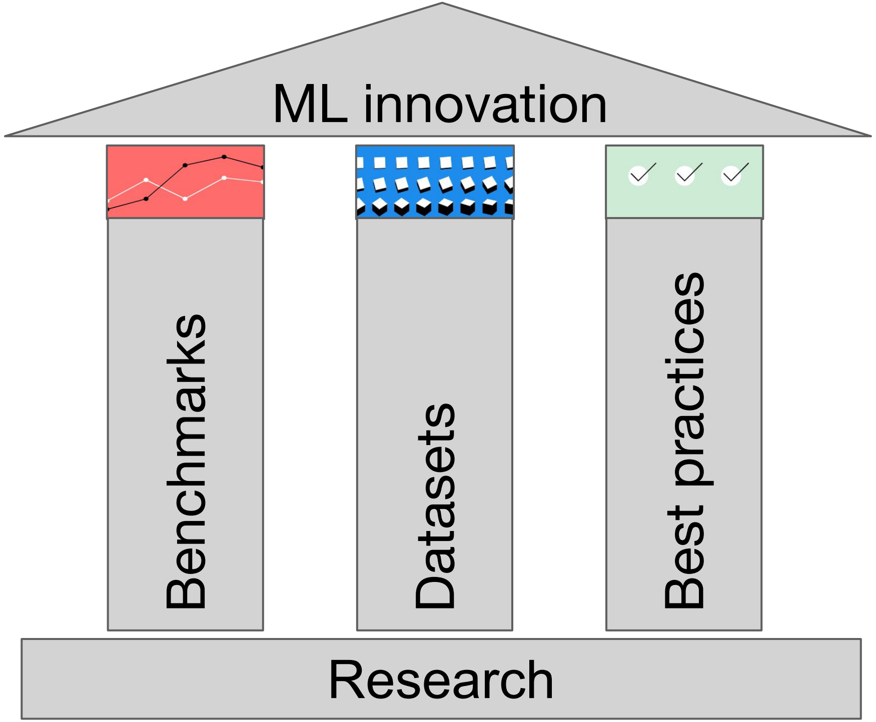
Members

EDGECORTIX, HUAWEI, Lenovo, LTECH, MOREH, NEUREALITY, NUTANIX, OctoML, rebellions_, SUPERMICRO

Academics from institutions including:

Harvard University
Indiana University
McGill University
Polytechnique Montreal
Peng Cheng Laboratory
Stanford University
University of California, Berkeley
University of Toronto
University of Tübingen
University of York, United Kingdom
Yonsei University

Mission: Better ML for Everyone



MLPerf breadth: µWatts to MegaWatts

Evolution over time

Scale	**2018**	**2019**	**2020**	**2021**
Training - HPC				
Training				
Inference - Datacenter				
Inference - Edge				
Inference - Mobile				
Inference - Tiny (IoT)				
Storage				‘21?

Improving technical maturity

New training/inference benchmarks
- Recommendation: DLRM + 1TB dataset
- Medical imaging: 3D U-NET
- Speech-to-text: RNN-T
- NLP: BERT + wikipedia

Standardized methodology for Training
- Optimizer definitions
- Hyperparameter definitions
- Reference Convergence Points (RCP)

Adding power measurement to Inference

Mobile App on Android, iOS

Tiny launched in June 2021

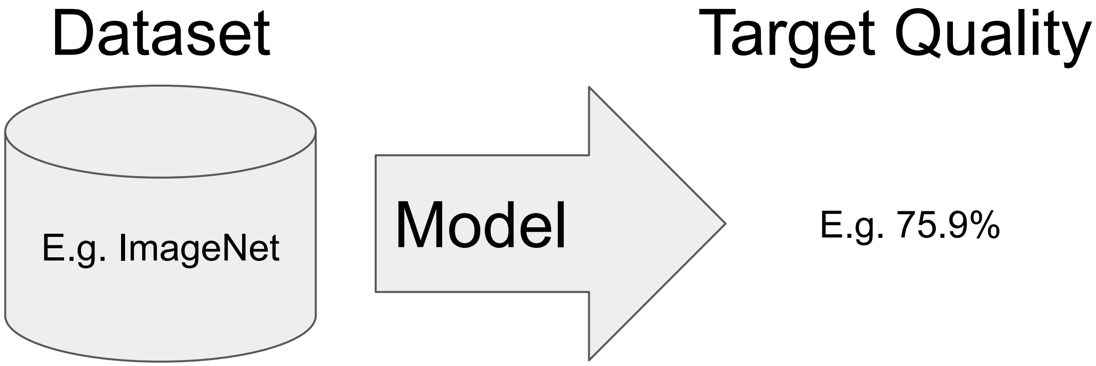
MLPerf **Training** Benchmark

Peter Mattson, Christine Cheng, Cody Coleman, Greg Diamos, Paulius Micikevicius, David Patterson, Hanlin Tang, Gu-Yeon Wei, Peter Bailis, Victor Bittorf, David Brooks, Dehao Chen, Debojyoti Dutta, Udit Gupta, Kim Hazelwood, Andrew Hock, Xinyuan Huang, Atsushi Ike, Bill Jia, Daniel Kang, **David Kanter**, Naveen Kumar, Jeffery Liao, Guokai Ma, Deepak Narayanan, Tayo Oguntebi, Gennady Pekhimenko, Lillian Pentecost, Vijay Janapa Reddi, Taylor Robie, Tom St. John, Tsuguchika Tabaru, Carole-Jean Wu, Lingjie Xu, Masafumi Yamazaki, Cliff Young, and Matei Zaharia

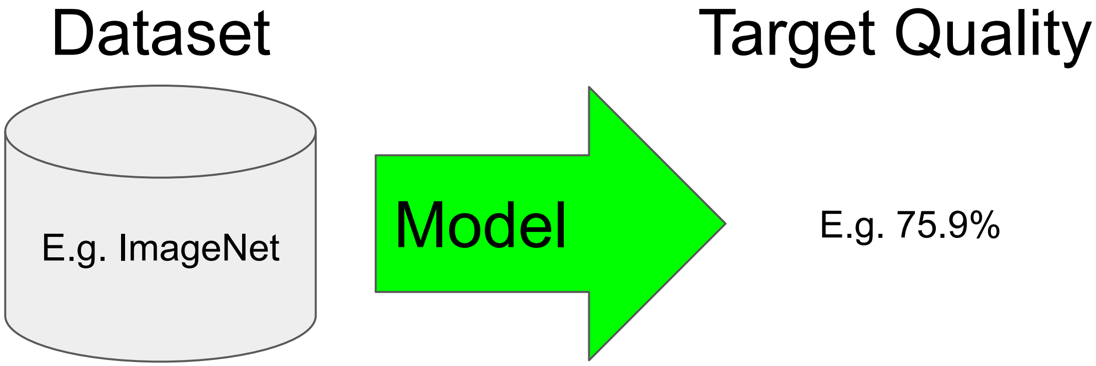
https://arxiv.org/abs/1910.01500

ML
C

MLPerf Training benchmark definition



Two divisions with different model restrictions



Closed division: specific model e.g. ResNet v1.5 → direct comparisons

Open division: any model → innovation

MLPerf Training 1.0 and 1.1 Suite

Task	Dataset	Model	Quality Target
Recommendation	Criteo 1TB	DLRM	0.8025 AUC
Speech recognition (*new*)	LibreSpeech	RNN-T	0.058 Word Error Rate
NLP (*improved*)	Wikipedia 2020-01-01	BERT-large	0.712 Mask-LM
Image Classification	ImageNet 2012	ResNet-50 v1.5	75.9% top-1
Object Detection (light)	COCO 2017	SSD-ResNet-34	0.23 mAP
Object Detection (heavy)	COCO 2017	Mask R-CNN	0.377 Box min AP and 0.339 Mask min AP
3D segmentation (*new*)	2019 KiTS Challenge	3D U-Net	0.908 Mean DICE score
Reinforcement learning	N/A	Mini-Go (19x19)	50% win rate

Metric: time-to-train

Alternative is throughput

Easy / cheap to measure

But can increase throughput at cost of total time to train!

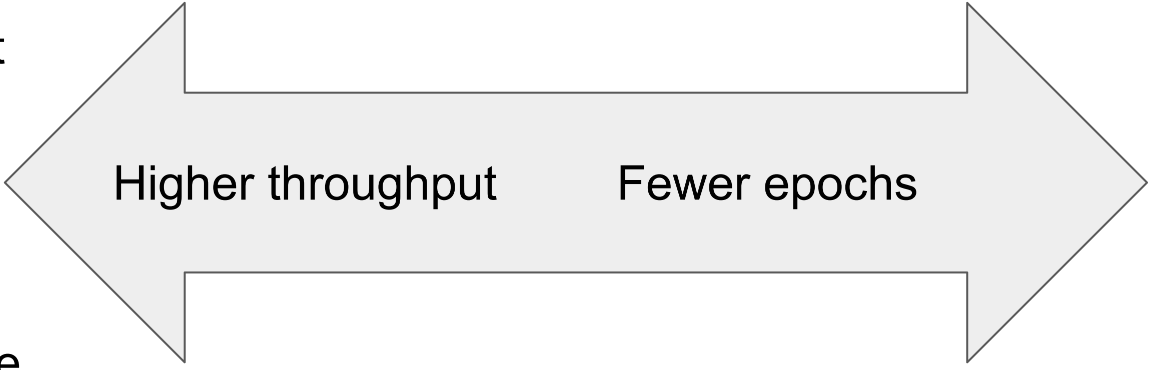
Time-to-train (end-to-end)

Time to solution!

Computationally expensive

High variance

Least bad choice



Lower precision
Higher batch size

Higher precision
Lower batch size

Time-to-train excludes

System initialization

Depends on cluster configuration and state

Model initialization

Disproportionate for big systems with small benchmarking datasets

Data reformatting

Mandating format would give advantage to some systems

Challenges and Contributions

ML Training benchmarking challenges

Diverse software stacks and hardware systems	• Can't use the same executable • Can't use the same *code*

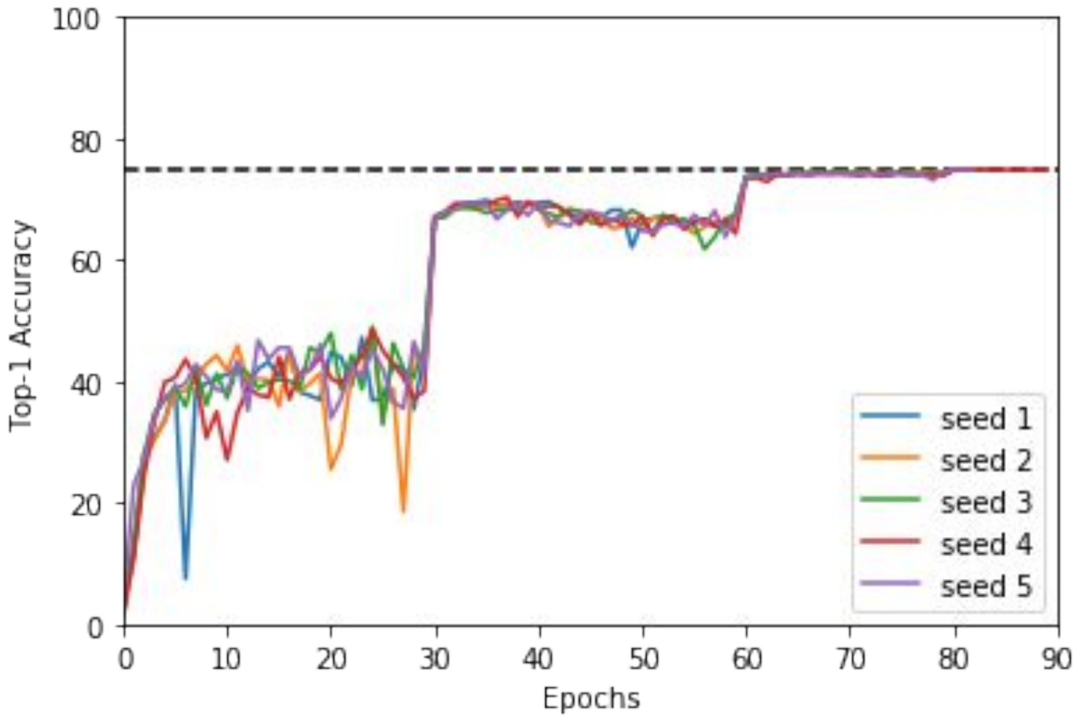
ML Training benchmarking challenges

Diverse software stacks and hardware systems	• E.g.: larger systems → larger SGD mini batches → different optimizer hyperparams • Hyperparameter tuning is computationally expensive, can be unfair
Different scales and/or numerics require tuning	

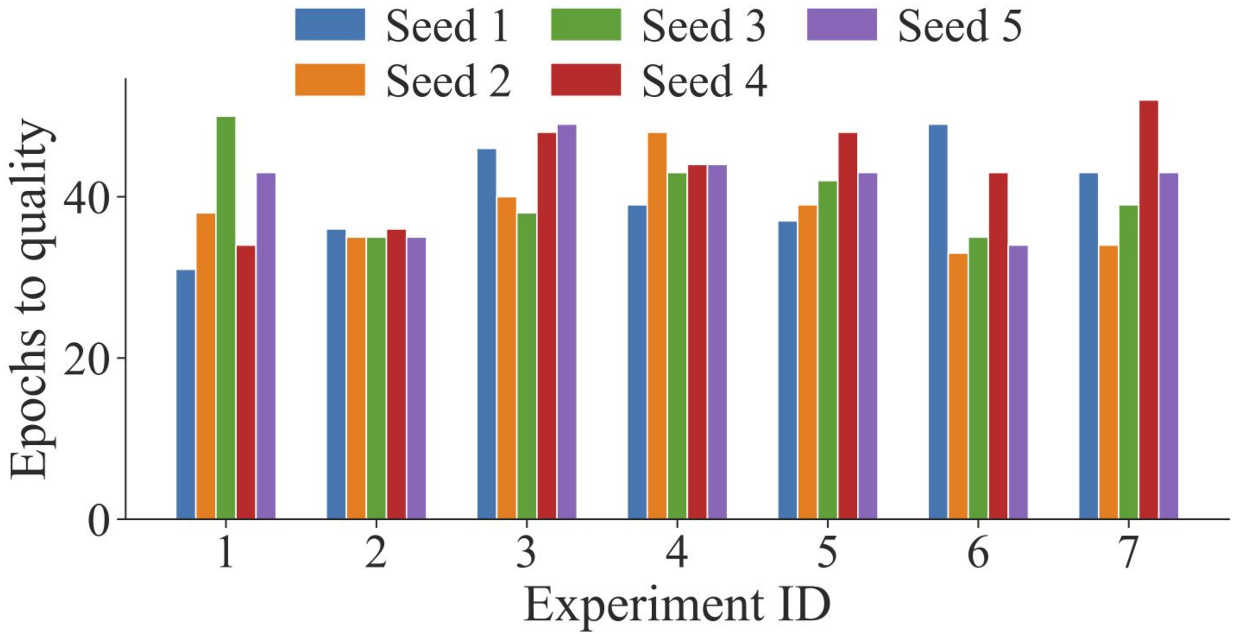
ML Training benchmarking challenges

Diverse software stacks and hardware systems	• Random weight initialization • Non-deterministic floating point effects
Different scales and/or numerics require tuning	
Convergence is stochastic	

Convergence variance: ResNet



Convergence variance: MiniGo



MLPerf contributions

Diverse software stacks and hardware systems	**Reference implementations** **Rules for reimplementation**
Different scales and/or numerics require tuning	
Convergence is stochastic	

MLPerf contributions

Diverse software stacks and hardware systems	Reference implementations Rules for reimplementation
Different scales and/or numerics require tuning	**Tunable hyperparameters; limited range of values**
Convergence is stochastic	

MLPerf contributions

Diverse software stacks and hardware systems	Reference implementations Rules for reimplementation
Different scales and/or numerics require tuning	Tunable hyperparameters; limited range of values
Convergence is stochastic	**Require multiple runs** **Drop low and high, average**

Submission Process

MLPerf Training Categories and Divisions

- Two Divisions
  - Closed: Mathematically equivalent to the reference model, to enable optimization on many different systems with a level playing field
    - Limited set of hyperparameters can vary, e.g., batch size, numerics, padding
    - Cannot change: Random data sort order, # of layers
  - Open Model: not mathematically equivalent to the reference
    - Could be very different, or a small difference, submitters should describe

- Three Categories
  - Available: Commercially available at submission
  - Preview: Commercially available soon (~6 months from submission)
  - RDI: Not commercially available, e.g. research, prototype, or internal systems

Pre-submit

Download **reference implementation**, read rules,
join submitters working group

↓

Reimplement benchmark for system under test (SUT)

↓

Tune hyperparameters (allowed by list, to allowed values)

↓

Run benchmark required number of times

↓

Submit logs from all runs, code, metadata in Github by deadline

↓

Post-submit

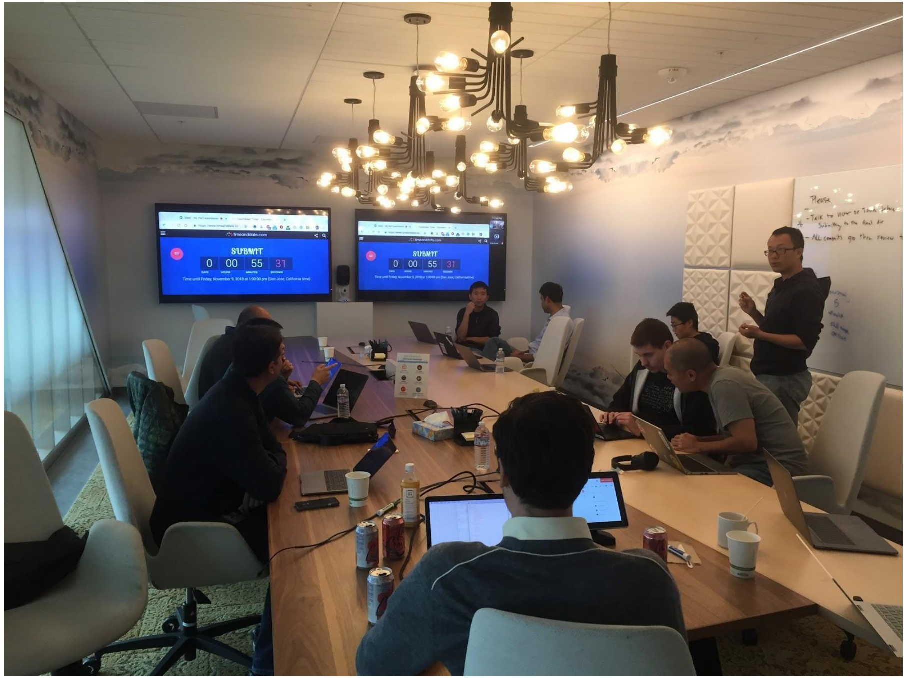
All submitters **peer review** all submissions, raise issues

Borrow hyperparameters from other submissions and resubmit if desired

MLPerf posts all results and makes logs, metadata, and code public under Apache-2

Celebrate!!!

ML
C

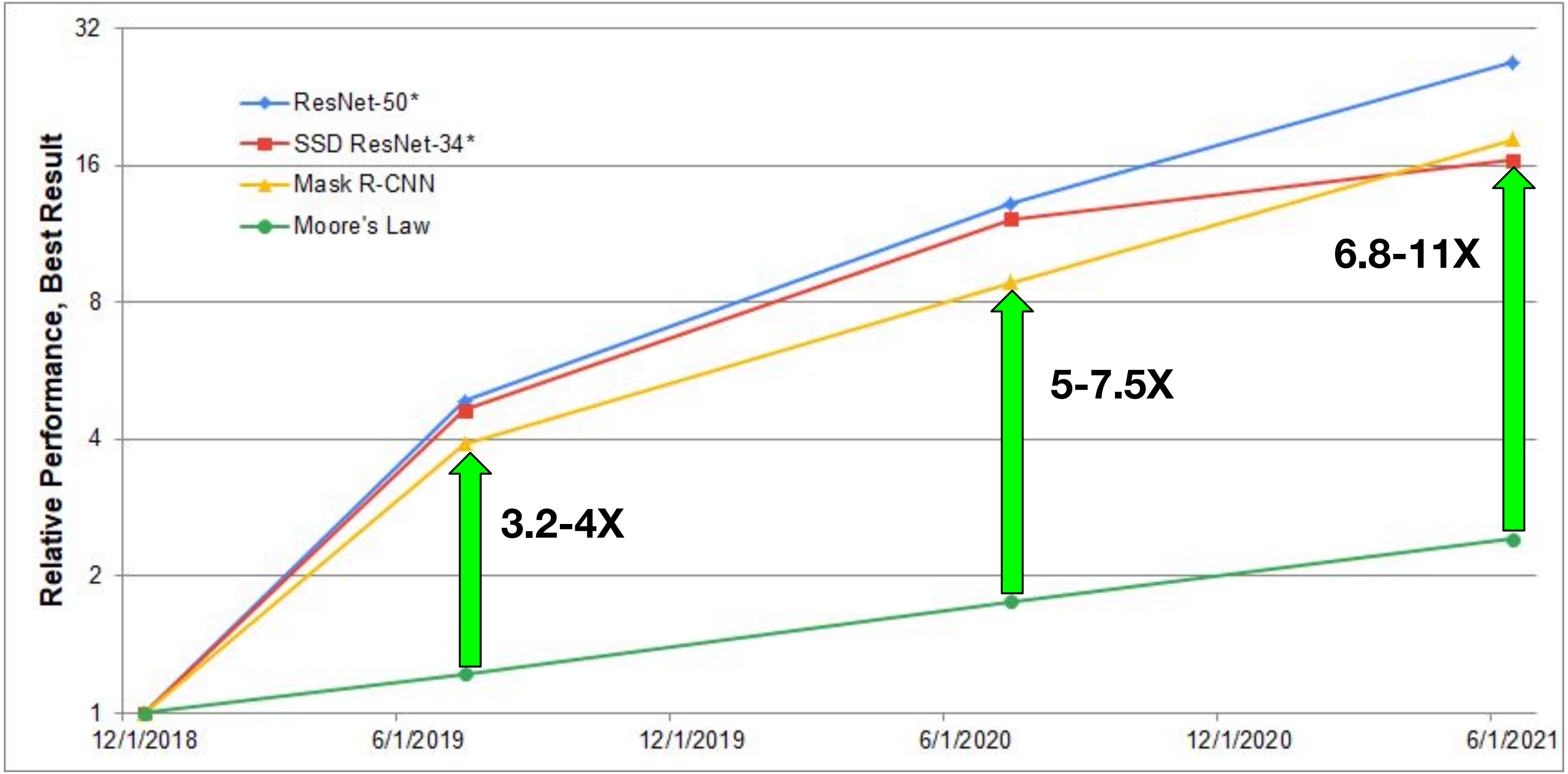

timeanddate.com
SUBMIT
0 00 55 31
Time until Friday, November 9, 2018 at 1:00:00 pm (San Jose, California time)

Results and Lessons Learned

Impact of good benchmarks

Benchmarks	Competition	Better Software / HW
• Defined set of problems • Clear metrics	• Competing engineering teams try different approaches • Results show what works best	• Improved understanding of performance • Faster, more scalable software stacks • Future hardware designs driven by best-of-breed ideas

MLPerf™ Training Outstrips Moore's Law



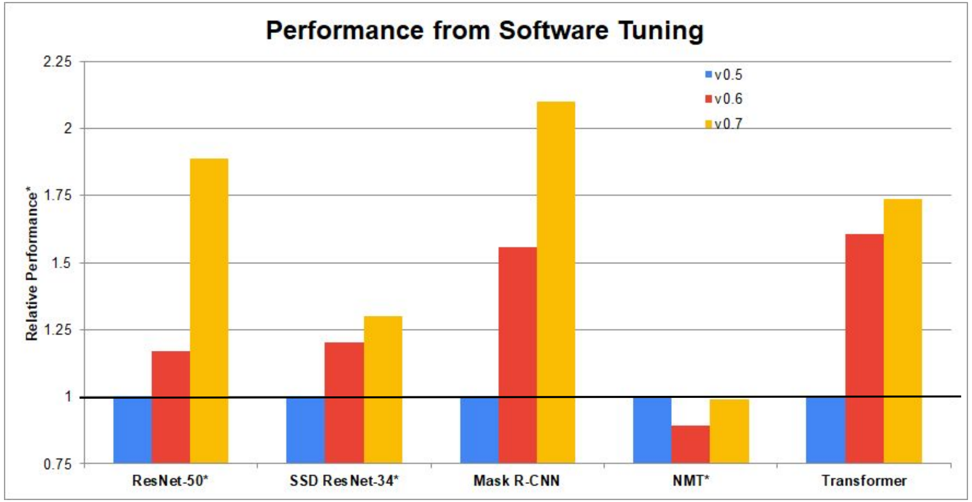
MLPerf Drives Better Software

- Closed, available submissions
- Single-node, same hardware, **new software versions**

- Many benchmarks **increased accuracy requirements in v0.6**

- Upto **2.1X better performance** on identical hardware

- Comparing against a highly optimized baseline

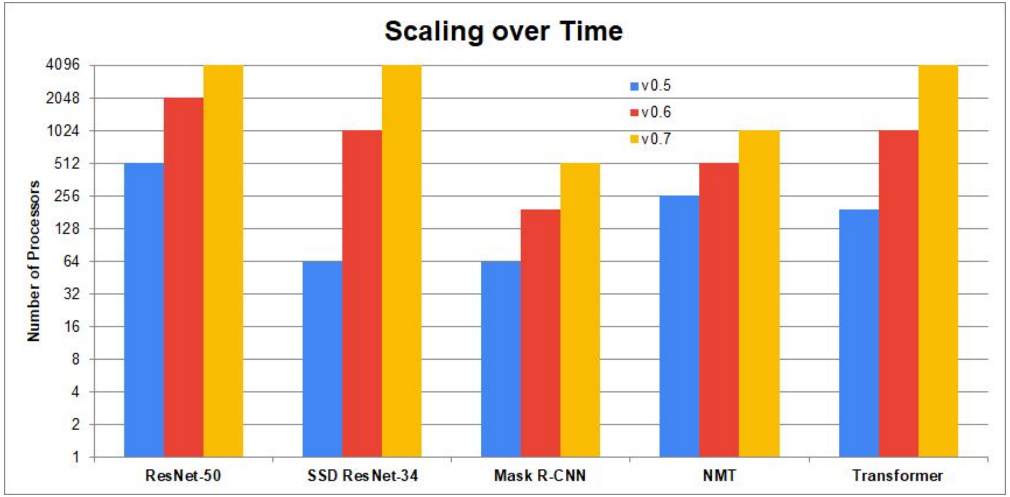


* ResNet-50, SSD, NMT accuracy targets increased
Sources: 0.5-12, 13; 0.6-8, 9, 0.7-39, 40;

ML
•C

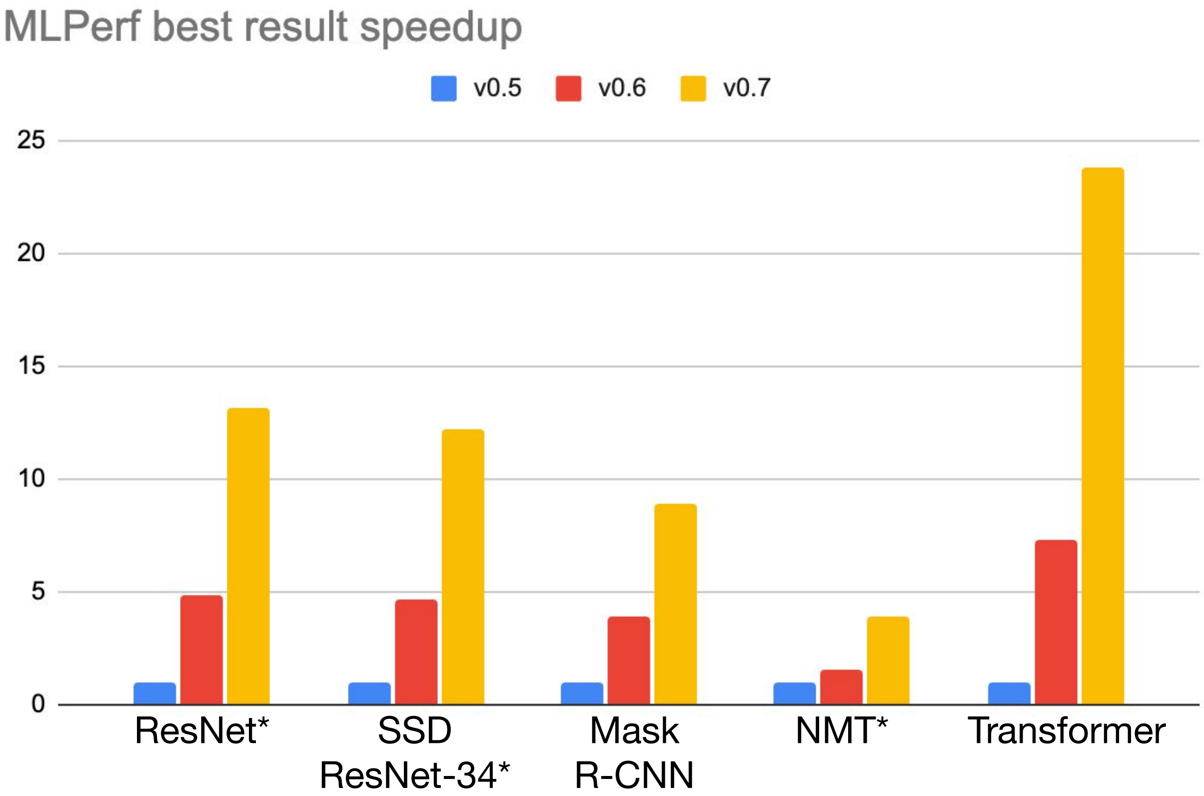
MLPerf Drives Scalability

- Largest system submitted, any division/category in Training v0.5-0.7
- All benchmarks scale differently!
- **4-64X** more parallelism in less than 2 years
- Lots of progress in software and tuning



Sources: 0.5-11, 14, 15, 16, 25; 0.6-5, 6, 11, 23, 30, 33; 07-34, 36, 66, 67

MLPerf Drives Performance



Some Initial Thoughts

- Performance is reported as time-to-train, smaller is better

- MLPerf Training is a full system benchmark and tests many aspects
  - Model / training algorithm (e.g., hyperparameters, optimizer, model parallelism)
  - Software (e.g., framework, numerics, compilers, math libraries)
  - Hardware (e.g., CPU, accelerators, interconnect, networking, server configuration)

- Scale matters, running on 8 processors is different than 64, 512, or 4K
  - Interconnect matters for larger systems
  - Model partitioning matters (impacts communication patterns, load balancing)
  - Like most scale-up problems, efficiency drops at larger system size
  - Larger batch size (for more nodes) requires more compute to converge
  - Don’t compare per-chip performance for 8 processors and 4K, very different

Listening to the Results

- Every result says something interesting, but it may not be obvious
    - Look at submissions that are similar across some dimensions, e.g., same vendor, same scale, same processor, best performance...but different in other dimensions

- Scaling system size
- Scaling over time
- Tuning software over time
- New software stacks
- Systems progress from RDI/Preview to Available
- New processors

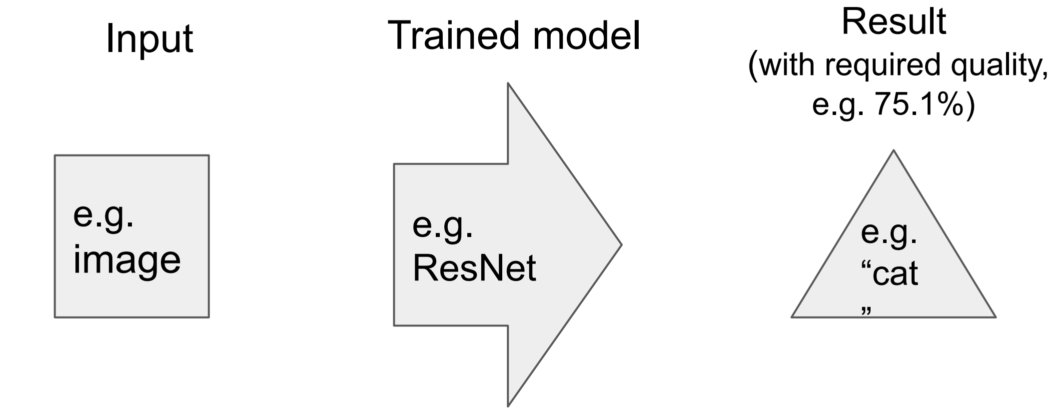
MLPerf **Inference** Benchmark

Vijay Janapa Reddi, Christine Cheng, **David Kanter**, Peter Mattson, Guenther Schmuelling, Carole-Jean Wu, Brian Anderson, Maximilien Breughe, Mark Charlebois, William Chou, Ramesh Chukka, Cody Coleman, Sam Davis, Pan Deng, Greg Diamos, Jared Duke, Dave Fick, J. Scott Gardner, Itay Hubara, Sachin Idgunji, Thomas B. Jablin, Jeff Jiao, Tom St. John, Pankaj Kanwar, David Lee, Jeffery Liao, Anton Lokhmotov, Francisco Massa, Peng Meng, Paulius Micikevicius, Colin Osborne, Gennady Pekhimenko, Arun Tejusve Raghunath Rajan, Dilip Sequeira, Ashish Sirasao, Fei Sun, Hanlin Tang, Michael Thomson, Frank Wei, Ephrem Wu, Lingjie Xu, Koichi Yamada, Bing Yu, George Yuan, Aaron Zhong, Peizhao Zhang, Yuchen Zhou

https://arxiv.org/abs/1911.02549

ML
•C

MLPerf Inference Definition



Submission division	**Closed**	**Open**
Inference	Strict rules Apples-to-apples ML system comparison	Permissive rules Better models than reference

MLPerf benchmarking scope: ML systems (HW + SW)

MLPerf Inference v1.0 Workloads

Datacenter / Edge Inference

Use Case	Reference Network
Image Classifier	ResNet-50 v1.5
Object detector (large)	SSD ResNet-34
Object detector(small)	SSD MobileNet v1 (edge only)
3D medical imaging	3D UNET
Speech-to-text	RNN-T
NLP / Q&A	BERT Large
Recommendation	DLRM (datacenter only)

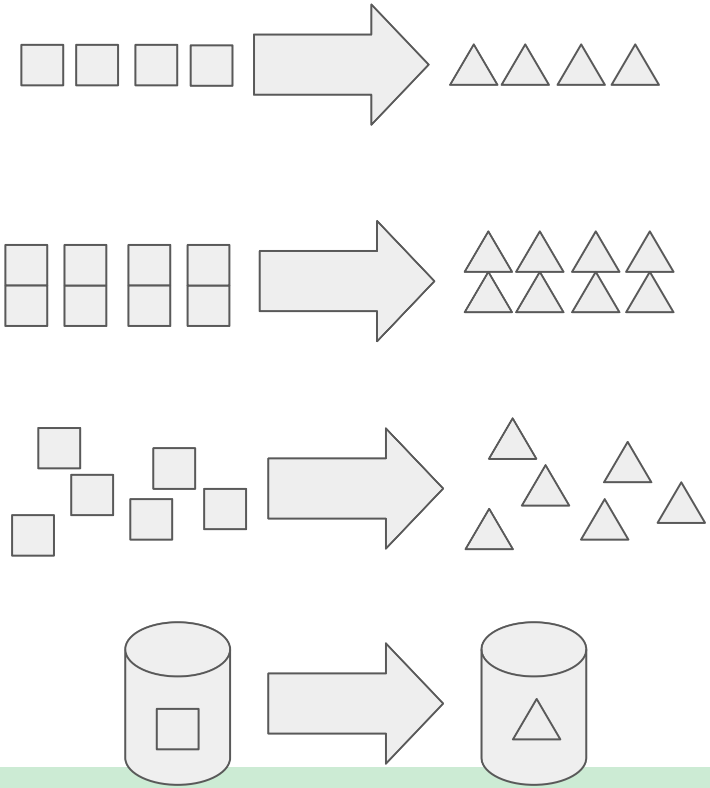
Data Center: Offline and Server scenario
Edge: Single Stream, Offline, (deprecating Multi-Stream)

Mobile Inference

Use Case	Reference Network
Image Classifier	MobileNetEdge
Object Detector	MobileDet
Image Segmentation	DeepLab v3
NLP / Q&A	Mobile-BERT

Mobile: Single Stream, and Offline scenario

Four **scenarios** to handle different use cases



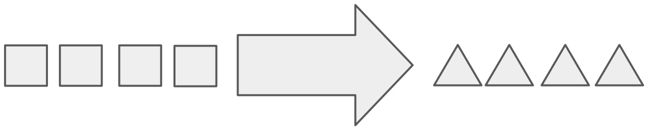
Single stream
(e.g. cell phone augmented vision)

Multiple stream
(e.g. multiple camera driving assistance)

Server
(e.g. translation app)

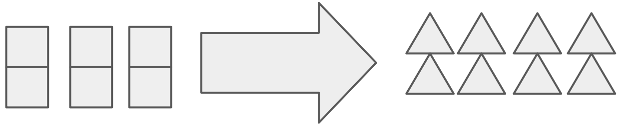
Offline
(e.g. photo sorting app)

Different metric for each scenario



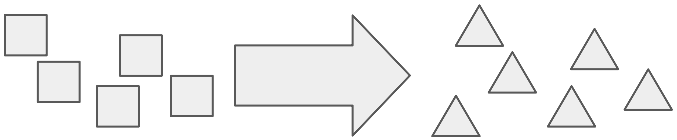
Single stream
e.g. cell phone augmented vision

Latency



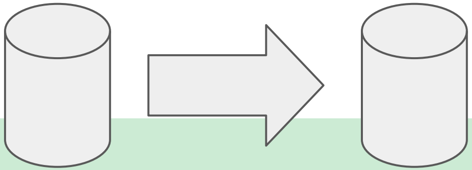
Multiple stream
e.g. multiple camera driving assistance

Number streams
subject to latency bound



Server
e.g. translation site

QPS
subject to latency bound

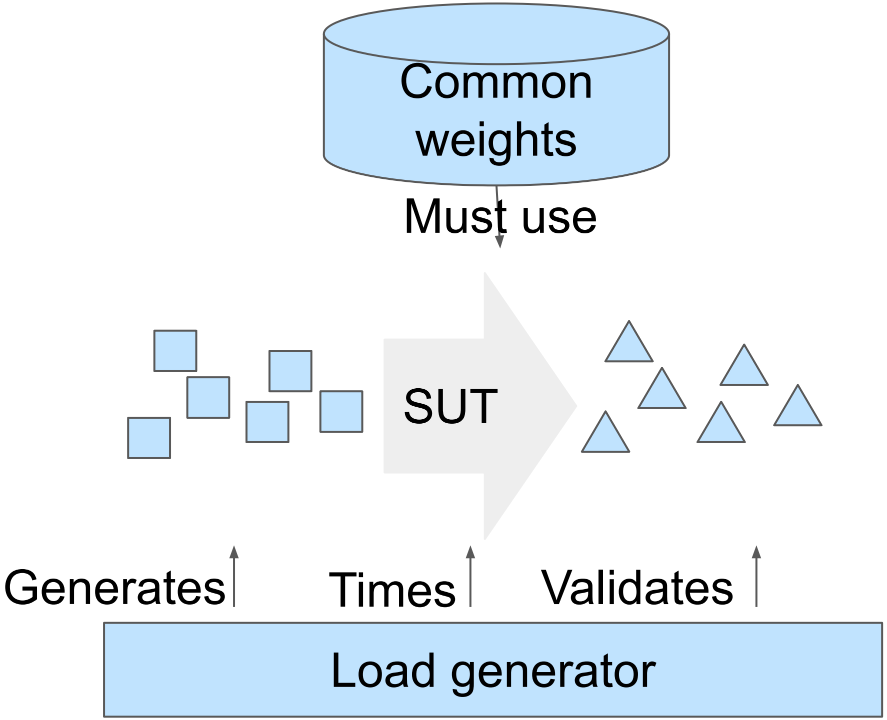


Offline
e.g. photo sorting

Throughput

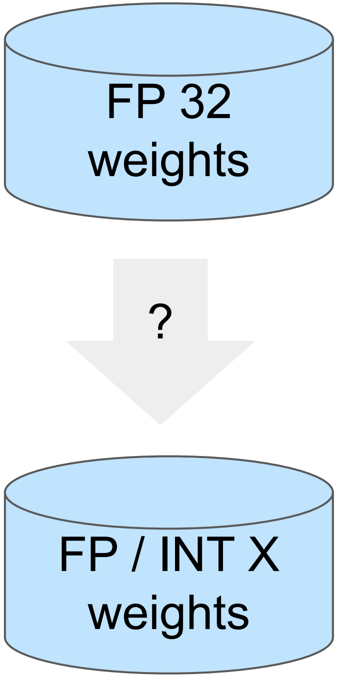
Inference Submitters’ Implementations

- Even greater range of software and hardware solutions
- So, allow submitters to reimplement subject to inference rules
  - Use standard set of **pre-trained weights for Closed Division**
  - Use **standard C++ “load generator”** that handles scenarios and metrics



Not a quantization contest!

- Quantization is key to efficient inference, but do not want a quantization contest
- Can the Closed division **quantize**?
  - **Yes**, but must be principled: describe reproducible method
- Can the Closed division **calibrate**?
  - **Yes**, but must use a fixed set of calibration data
- Can the Closed division **retrain**?
  - **No**, not a retraining contest. But, provide retrained 8 bit models..



Questions?

Executive Director: David Kanter

David Kanter is a Founder and the Executive Director of MLCommons where he helps lead the MLPerf benchmarks and other initiatives. He previously led the MLPerf Inference, Mobile, and Power working groups. He has 16+ years of experience in semiconductors, computing, and machine learning. He founded a microprocessor and compiler startup, was an early employee at Aster Data Systems, and has consulted for industry leaders such as Intel, Nvidia, KLA, Applied Materials, Qualcomm, Microsoft and many others. David holds a Bachelor of Science degree with honors in Mathematics with a specialization in Computer Science, and a Bachelor of Arts with honors in Economics from the University of Chicago.