
cerebras

ACCELERATING TRAINING IN THE CLOUD

Ardavan Pedram

William Lynch

Gary Lauterbach

RANGE OF APPLICATIONS

Image Classification

Object Detection

Semantic Segmentation

Computer Vision
CNNs

Speaker Diarization

Speech Recognition

Speech Recognition
RNNs, LSTMs

Translation

Sentiment Analysis

Natural Language Processing
Sequence to sequence

Recommender

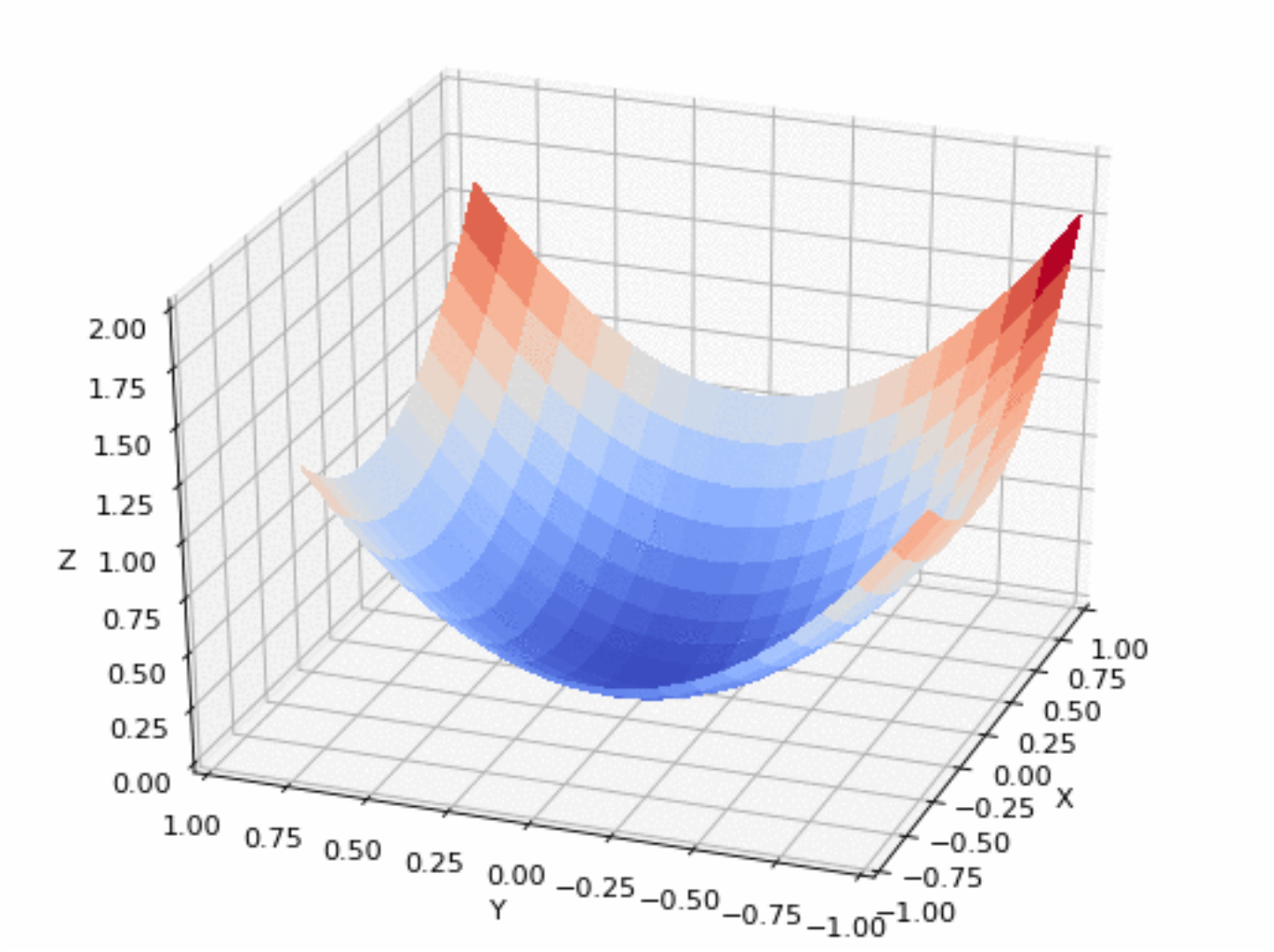
GamePlay

Others

AGENDA

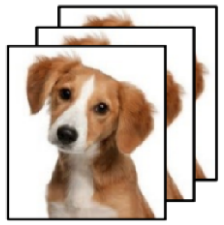
- Fundamentals of training
- Architecture features for training
- Scaling of training
- Benchmarking

BASICS OF GRADIENT DESCENT TRAINING

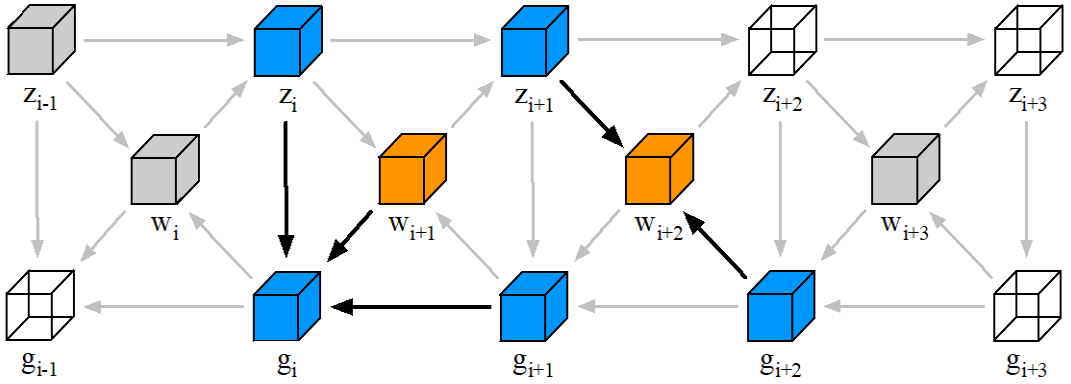


Sebastian Ruder, "An overview of gradient descent optimization algorithms" arXiv:1609.04747 15 Jun 2017
https://arxiv.org/pdf/1609.04747.pdf

BACKPROPAGATION

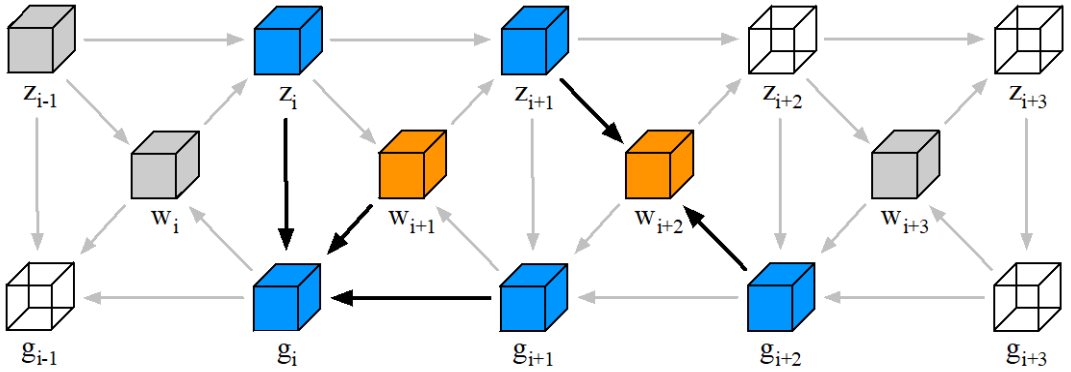


Input image batch



Loss Function

gradients



Loss Function

gradients

Simon Knowls, "Graphcore Intelligent Processing Unit (IPU)" Deep Learning at Supercomputer Scale Workshop, NIPS-17. https://www.matroid.com/scaledml/2018/simon.pdf

HYPERPARAMETERS & MACHINERY

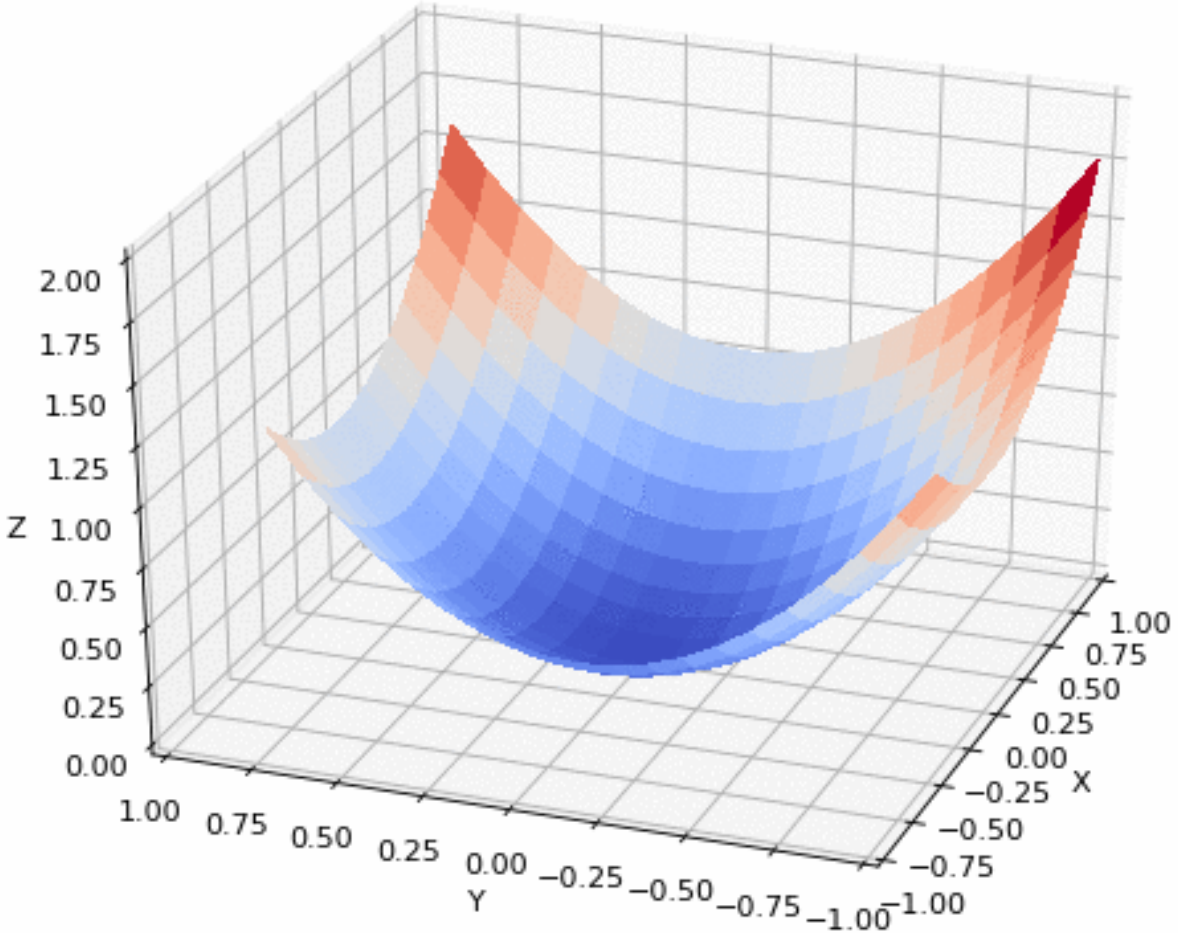
Machinery

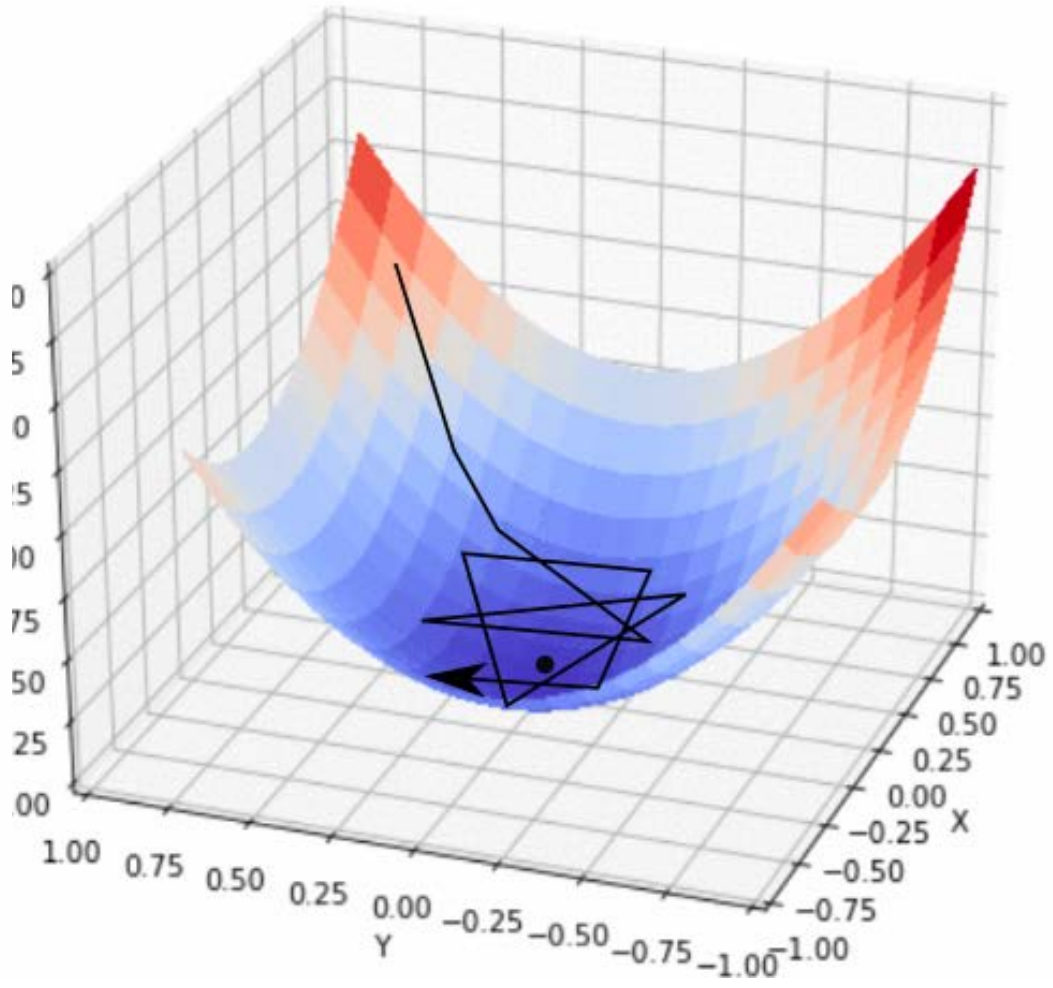
- Normalizers
- Loss functions
- Optimizers

Parameters into machinery

- Learning rate
  - Momentum
  - Decay
- Batch size

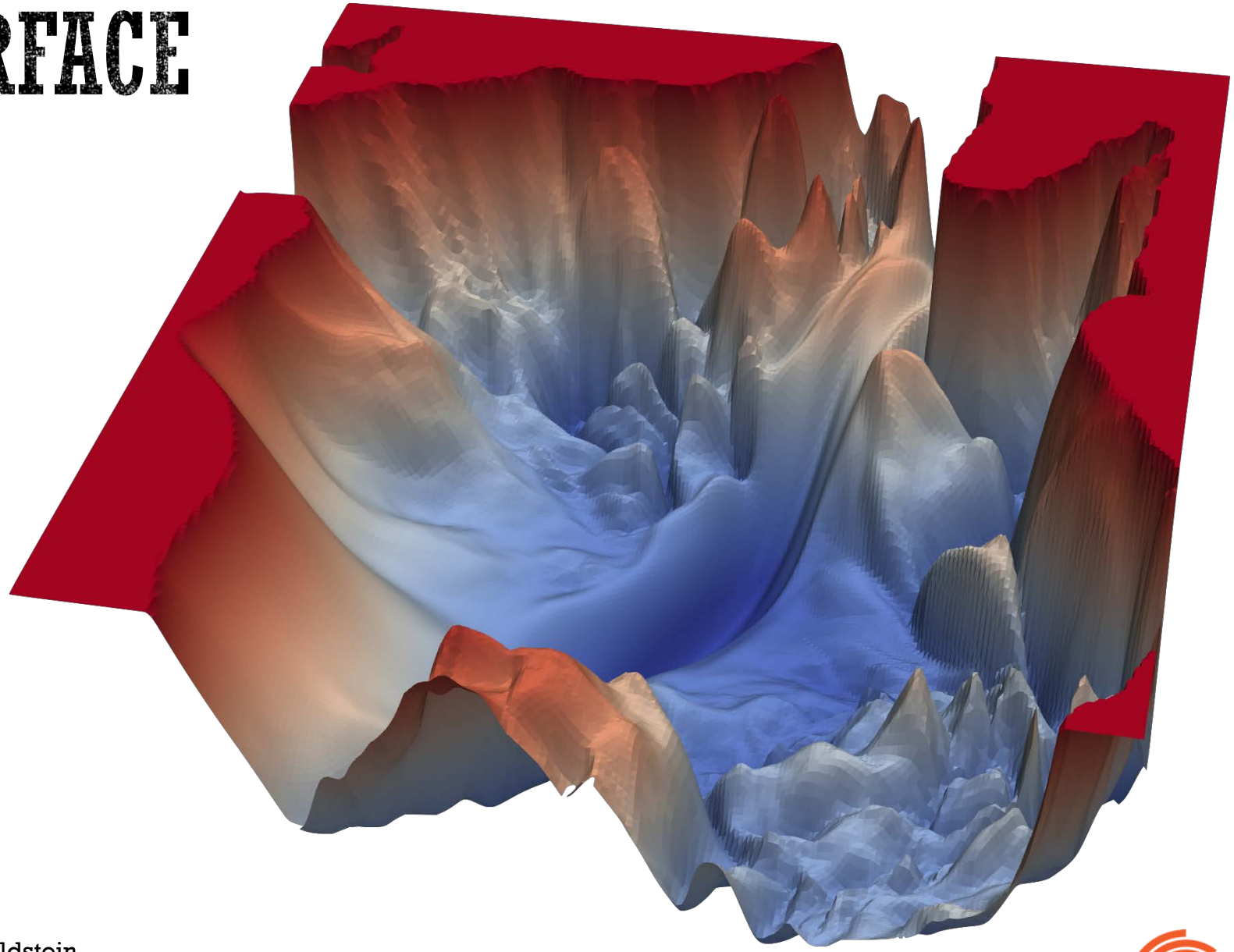
LEARNING RATE





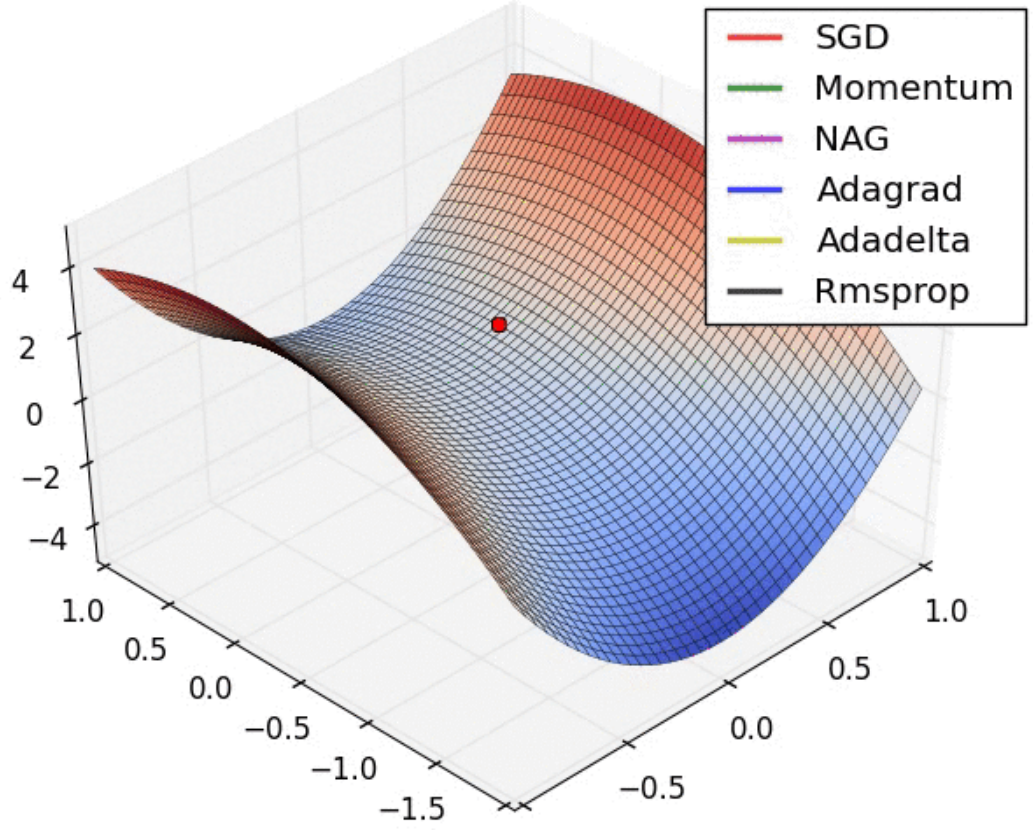
Sebastian Ruder, "An overview of gradient descent optimization algorithms" arXiv:1609.04747 15 Jun 2017
https://arxiv.org/pdf/1609.04747.pdf

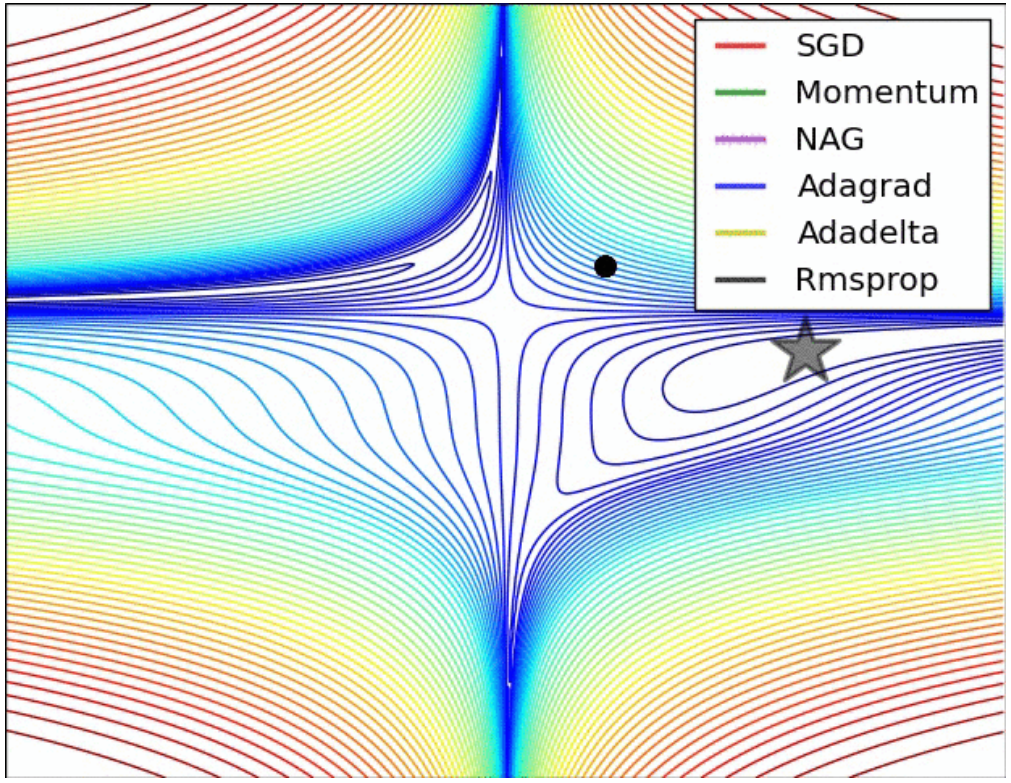
GRADIENT SURFACE



Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, Tom Goldstein,
"VISUALIZING THE LOSS LANDSCAPE OF NEURAL NETS" arXiv:1712.09913v2 [cs.LG] 5 Mar 2018
https://www.cs.umd.edu/~tomg/projects/landscapes/

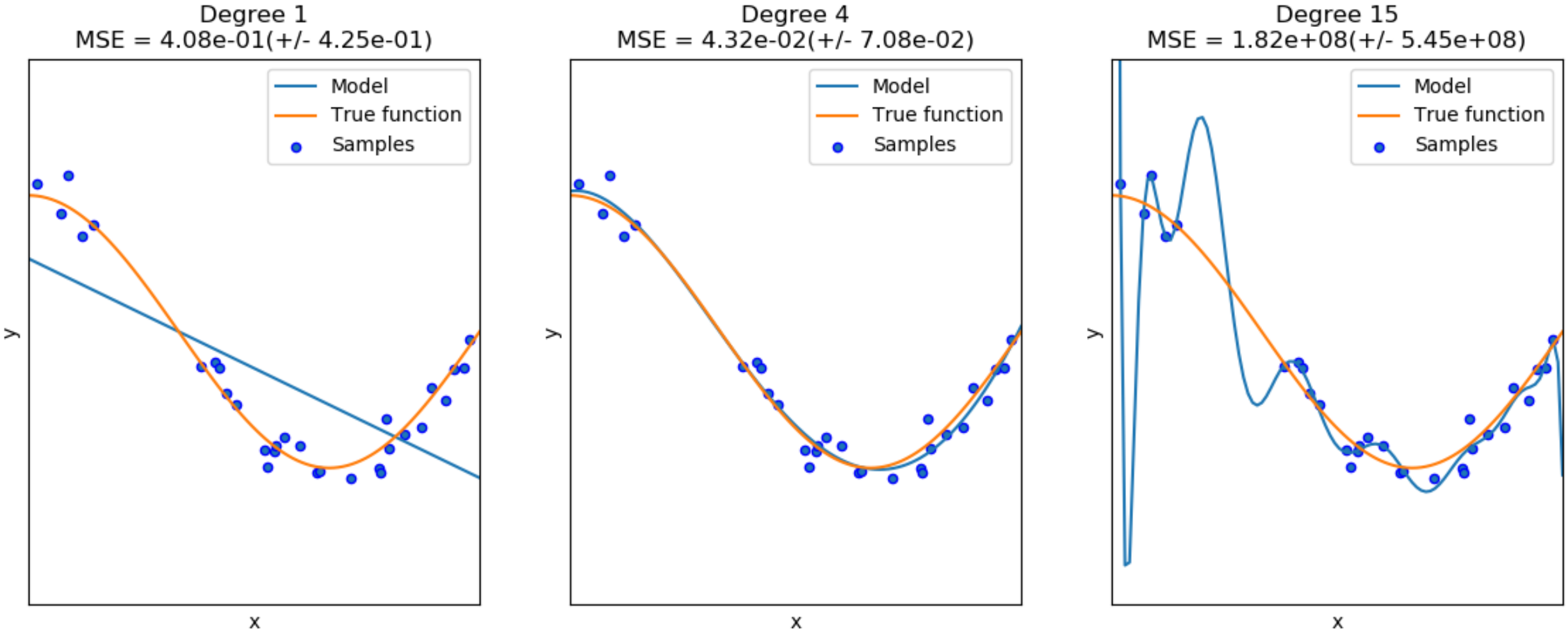
SGD OPTIMIZERS





Sebastian Ruder, "An overview of gradient descent optimization algorithms" arXiv:1609.04747 15 Jun 2017
https://arxiv.org/pdf/1609.04747.pdf

GENERALIZATION AND OVERFITTING



http://scikit-learn.org/stable/auto_examples/model_selection/plot_underfitting_overfitting.html

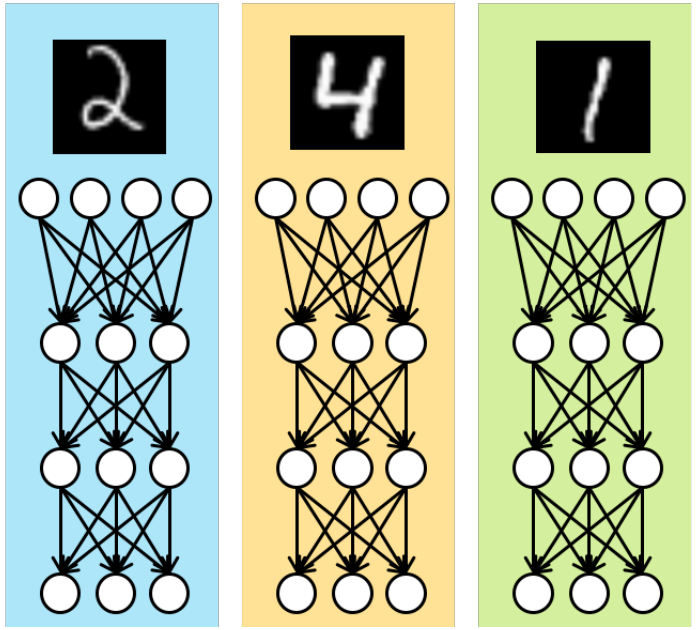
ARCHITECTURAL ATTRIBUTES

- Parallelism
- Precision, Quantization
- Sparsity

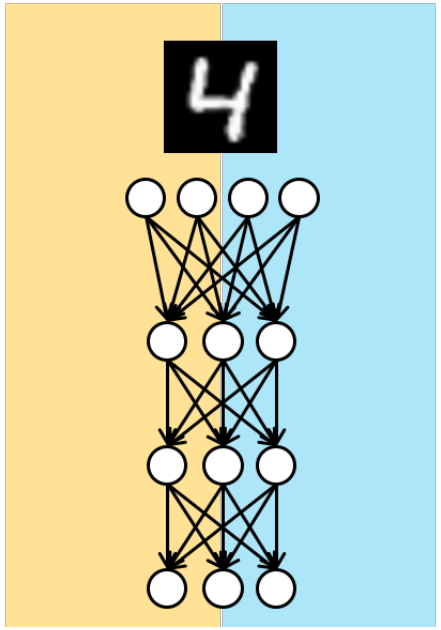
- **Parallelism**
  - Data parallelism
    - Coarse grain
      - Mini-batch size
      - Amortizing the cost of communication latency
    - Fine grain
      - SIMD
  - Model parallelism
    - Granularity of network chunks

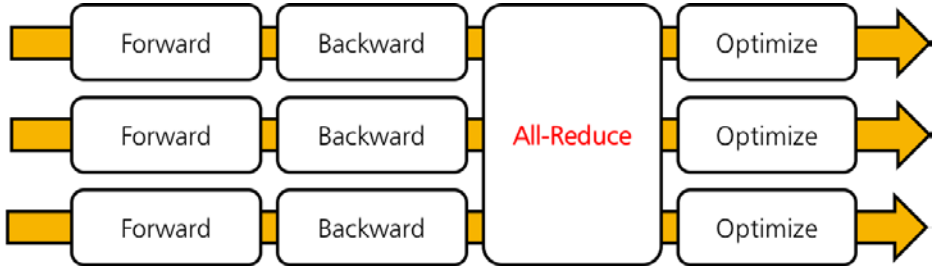
DATA VS. MODEL PARALLELISM

Data Parallel



Model Parallel



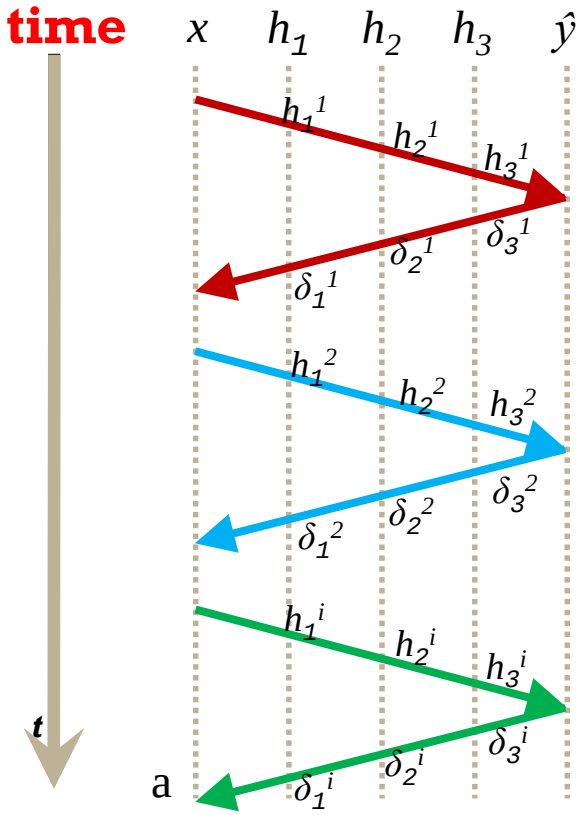


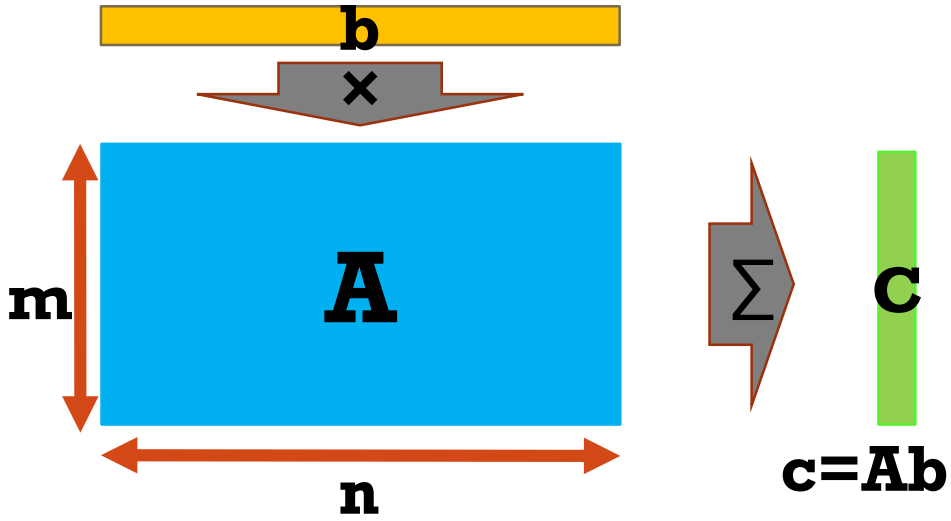
Data Parallel

http://chainermn.readthedocs.io/en/v1.0.0b2_a/tutorial/overview.html

STOCHASTIC GRADIENT DESCENT

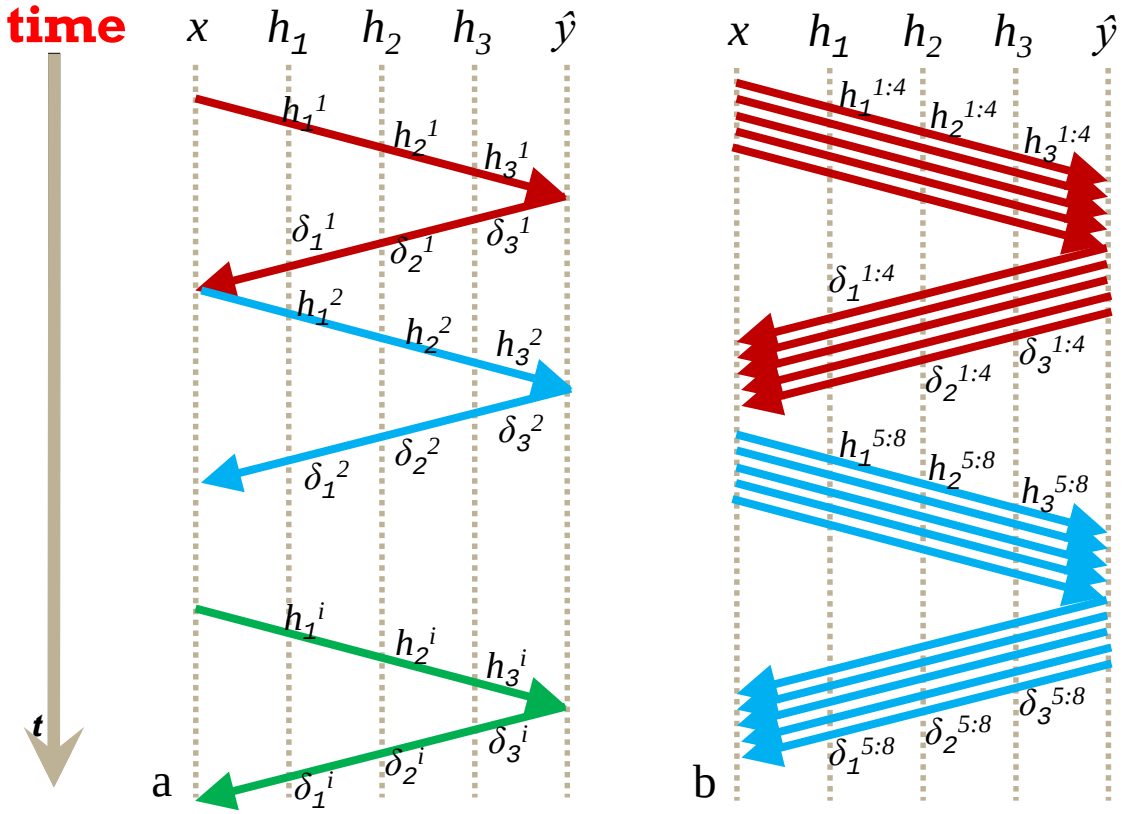
- GEMV
  - Inherently inefficient
  - Requirements
    - Broadcast (systolic /non-systolic)
    - Reduction





Yuanfang Li, Ardavan Pedram,"CATERPILLAR: Coarse Grain Reconfigurable Architecture for Accelerating the Training of Deep Neural Networks," IEEE-ASAP2017. https://arxiv.org/abs/1706.00517

BATCHED GRADIENT DESCENT

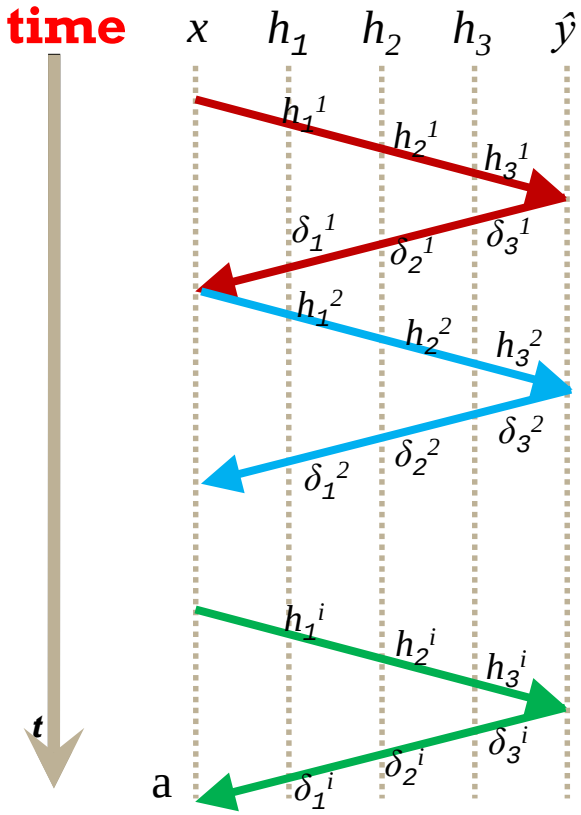


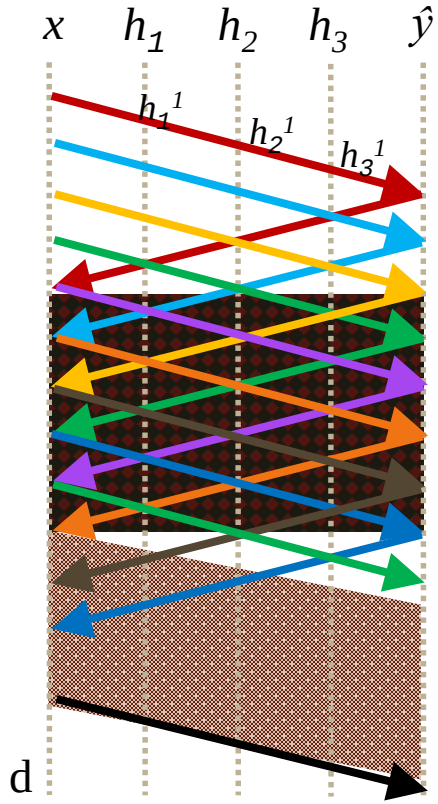
- Data parallelism
  - GEMV → GEMM
  - GEMM: Memory efficient kernel
  - # of weight updates / batch size

Yuanfang Li, Ardavan Pedram,"CATERPILLAR: Coarse Grain Reconfigurable Architecture for Accelerating the Training of Deep Neural Networks," IEEE-ASAP2017. https://arxiv.org/abs/1706.00517

PIPELINED BACKPROPAGATION

- Pipeline parallelization
  - Pipelining inputs
  - Layer locality
  - More efficient GEMVs
    - **Smaller reduction tree**
  - Weight temporal locality
    - Update and consume immediately





Yuanfang Li, Ardavan Pedram,"CATERPILLAR: Coarse Grain Reconfigurable Architecture for Accelerating the Training of Deep Neural Networks," The 28th IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP2017). https://arxiv.org/abs/1706.00517

WHAT DO WE NEED TO SUPPORT?

- **GEMM:** General Matrix Matrix multiplication
- **GEMV:** General Matrix Vector multiplication
- **Collective communications**
  - Gather
  - Reduce
  - All gather
  - All reduce
  - Broadcast
  - All-to-All

PRECISION AND SPARSITY

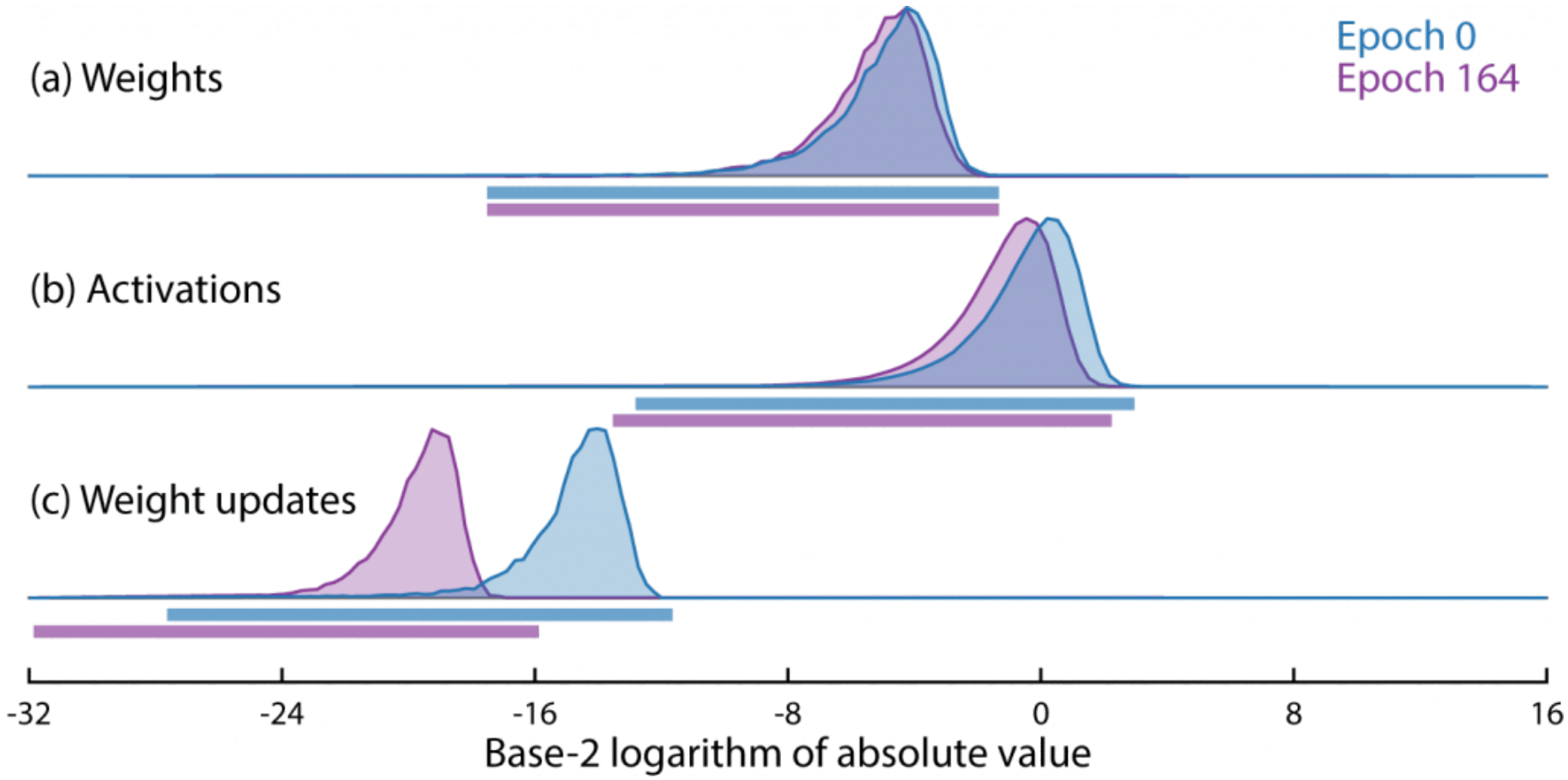
Precision for FPUs

- Distribution of scales
- Loss scaling

Sparsity

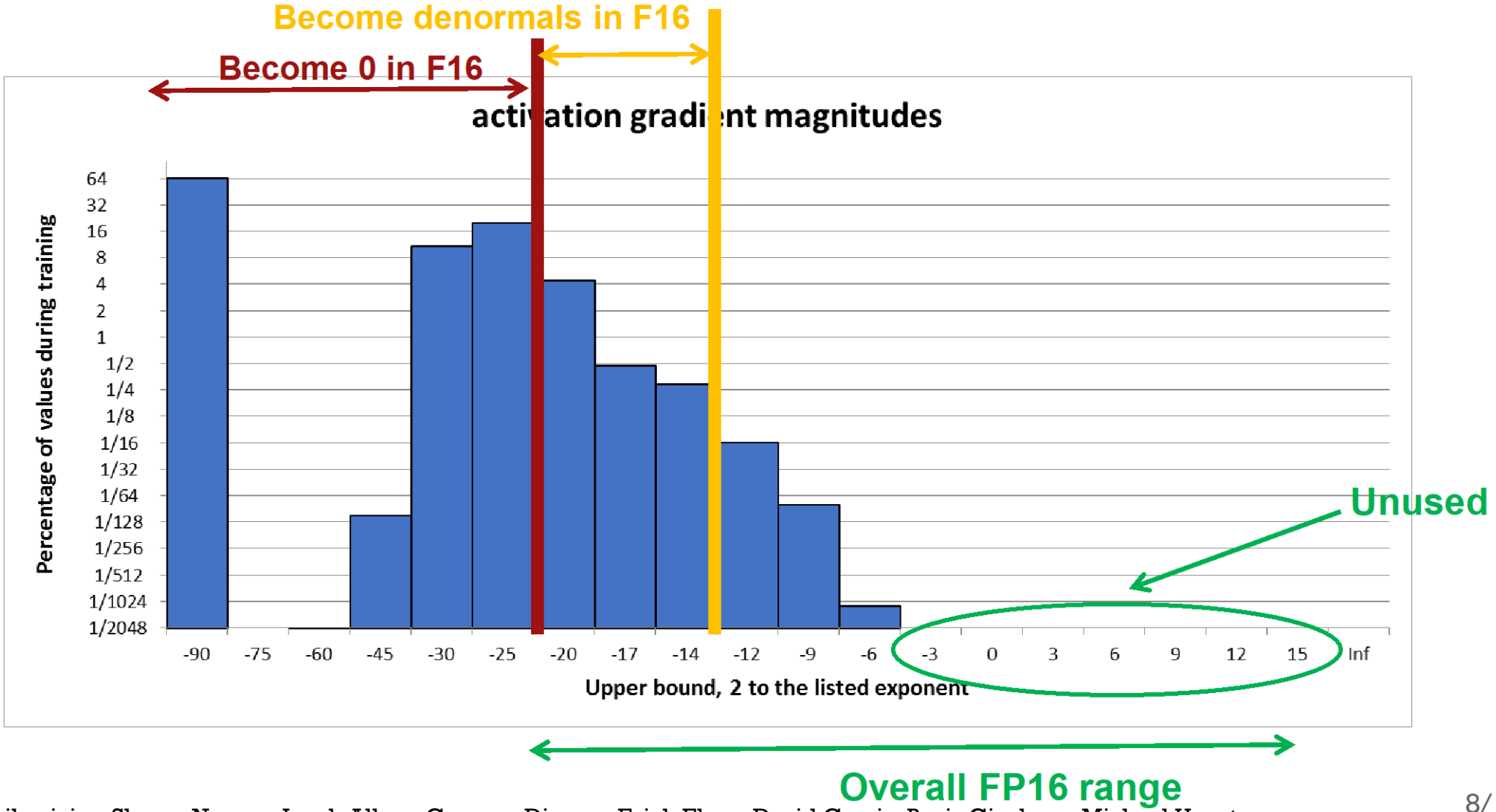
- Activation sparsity
- Weight sparsity

SCALES OF VALUES VARY WIDELY



Urs Köster, Tristan J. Webb, Xin Wang, Marcel Nassar, Arjun K. Bansal, William H. Constable, Oğuz H. Elibol, Scott Gray, Stewart Hall, Luke Hornof, Amir Khosrowshahi, Carey Kloss, Ruby J. Pai, Naveen Rao, "Flexpoint: An Adaptive Numerical Format for Efficient Training of Deep Neural Networks" Neural Information Processing Systems (NIPS) 2017. https://arxiv.org/abs/1711.02213

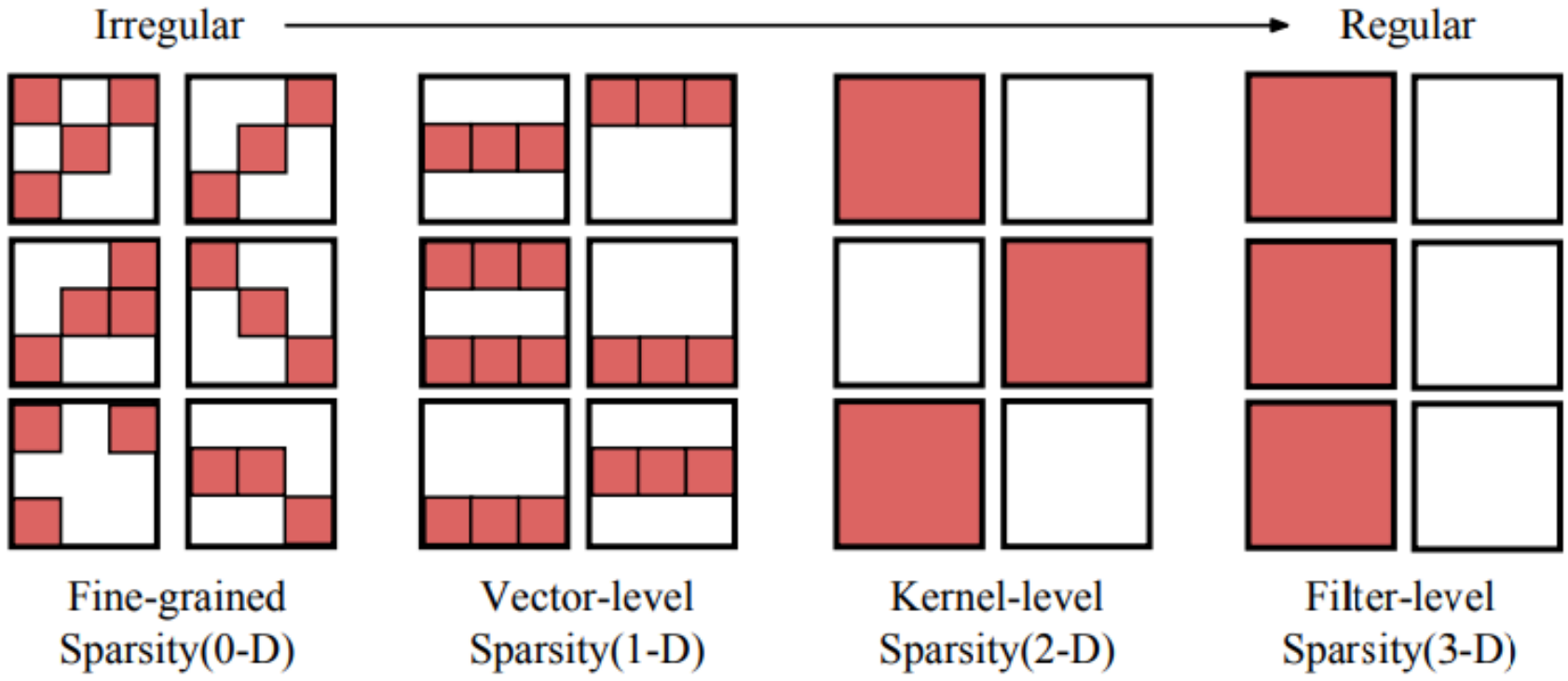
WE ARE MISSING THE RANGE WITH FP16



Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, Hao Wu, "Mixed Precision Training", ICLR 2018. https://arxiv.org/abs/1710.03740

SPARSITY HAS MANY FLAVORS

- Activation Sparsity
  - RELU / MAXPOOL (on back propagation)
- Weight Sparsity
  - Fine grain
  - Per row
  - Per column
  - Per kernel
  - Per channel
  - Per filter
  - Block sparsity



Huizi Mao, Song Han, Jeff Pool, Wenshuo Li, Xingyu Liu, Yu Wang, William J. Dally,“ Exploring the Granularity of Sparsity in Convolutional Neural Networks” CVPR’17 TMCV workshop. https://arxiv.org/abs/1705.08922

RNNS & CNNS BENEFIT FROM STRUCTURED SPARSITY

- RNNs [1]
  - 90% sparsity reduces relative accuracy by 10% to 20%
  - Solution: Make the sparse model larger
  - Large sparse model still have less parameters compared to the small dense baseline and achieves a slight increase in accuracy

- CNNs [2]
  - Pruning with large granularity will greatly hurt accuracy
  - Due to index savings, coarse-grain pruning can still achieve space savings even at a lower overall sparsity

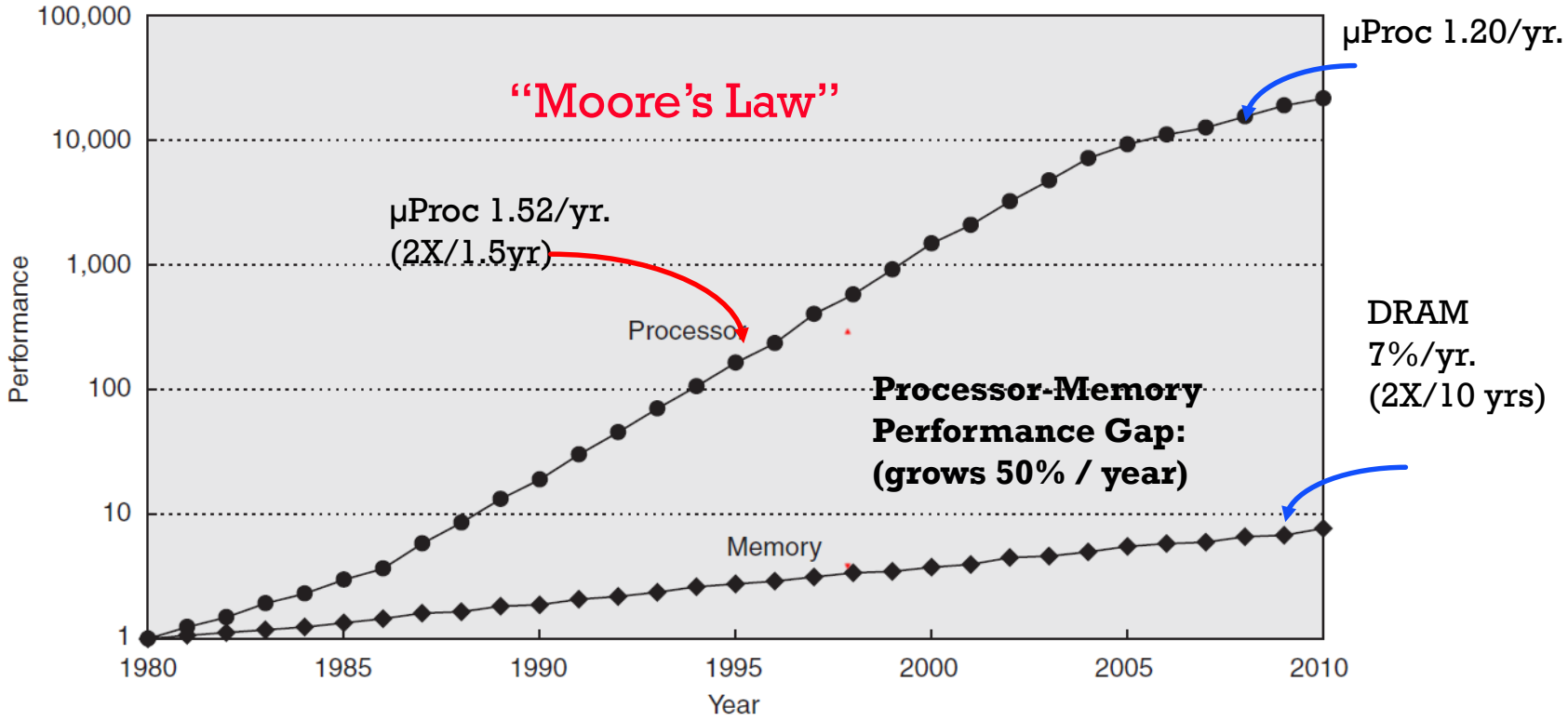
[1]- Sharan Narang, Erich Elsen, Gregory Diamos, Shubho Sengupta, "Exploring Sparsity In Recurrent Neural Network", ICLR 2017. https://arxiv.org/abs/1704.05119
[2]- Huizi Mao, Song Han, Jeff Pool, Wenshuo Li, Xingyu Liu, Yu Wang, William J. Dally," Exploring the Granularity of Sparsity in Convolutional Neural Networks" CVPR'17 TMCV workshop. https://arxiv.org/abs/1705.08922

SCALING OF TRAINING

- Scaling the problem: Same system, bigger network
  - Memory bottleneck
  - Cost of computation vs. communication
- Scaling the system: Bigger system
  - Synchronization bottleneck
  - Data communication on the cloud
  - Cloud scale synchronized SGD
  - Asynchronous SGD

OVERCOME MEMORY WALL WITH MEMORY HIERARCHY

Processor-DRAM Memory Gap



- 1980: no cache in micro-processor; 2010: 3-level cache on chip, 4-level cache off chip
- 1989 the first Intel processor with on-chip L1 cache was Intel 486, 8KB size
- 1995 the first Intel processor with on-chip L2 cache was Intel Pentium Pro, 256KB size
- 2003 the first Intel processor with on-chip L3 cache was Intel Itanium 2, 6MB size

John Hennessy, David Patterson, "Computer Architecture A Quantitative Approach", Morgan Kaufman. **ISBN-13:** 978-8178672663

MEMORY ACCESS IS >500× ARITHMETIC ENERGY

Operation	16 bit (integer)		64 bit (DP-FP)	
	E/op PJ	vs. Add	E/op PJ	vs. Add
ADD	0.18	1.0 ×	5	1.0 ×
Multiply	0.62	3.4 ×	20	4.0 ×
16-Word Register File	0.12	0.7 ×	0.34	0.07 ×
64-Word Register File	0.23	1.3 ×	0.42	0.08 ×
4 K-word SRAM	8	44 ×	26	5.2 ×
32 K-word SRAM	11	61 ×	47	9.4 ×
DRAM	**640**	**3556×**	**2560**	**512 ×**

Ardavan Pedram, Stephen Richardson, Sameh Galal, Shahar Kvatinsky, and Mark A. Horowitz, "Dark Memory and Accelerator-Rich System Optimization in the Dark Silicon Era," IEEE Design and Test Magazine Special Issue on Dark Silicon, April 2017. https://arxiv.org/pdf/1602.04183.pdf

GEMM ACCELERATORS 10× MORE EFFICIENT

	GFLOPS	W/mm²	GFLOPS/mm²	GFLOPS/W	Utilization
Cell BE (SP)	200	0.3	1.5	5	88%
NVidia GTX480 SM (SP)	**780**	0.2	0.9	**5.2**	70%
NVidia GTX480 SM **(DP)**	**390**	0.2	0.4	**2.6**	70%
Intel Core-i7 960 (SP)	96	0.4	0.5	1.2	95%
Intel Core-i7 960 (DP)	48	0.4	0.25	0.6	95%
Altera Stratix IV (DP)	100	0.02	0.05	3.5	90+%
ClearSpeed CSX700 (DP)	75	0.02	0.2	12.5	78%
Linear Algebra Processor (SP)	**1200**	**0.2**	**6-11**	**55**	**90+%**
Linear Algebra Processor (DP)	**600**	**0.2**	**3-5**	**25**	**90+%**

45nm scaled power / performance @ 1.4GHz for equivalent throughput

Ardavan Pedram, Robert van de Geijn , Andreas Gerstlauer "Codesign Tradeoffs for High-Performance Low-Power Linear Algebra Architectures," IEEE Transactions on Computers, Special Issue On Energy Efficient Computing, August 2012.

VOLTA: GPU INFUSED WITH GEMM CORES

21B transistors
815 mm²

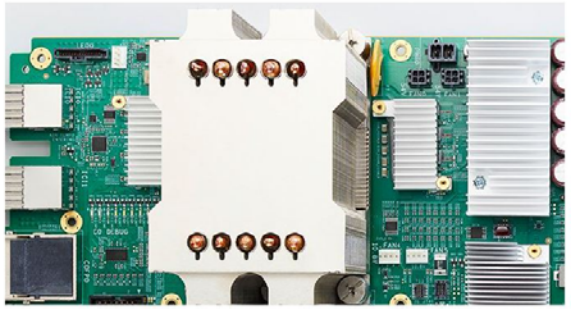
80 SM
5120 CUDA Cores
640 Tensor Cores

16 GB HBM2
900 GB/s HBM2
300 GB/s NVLink

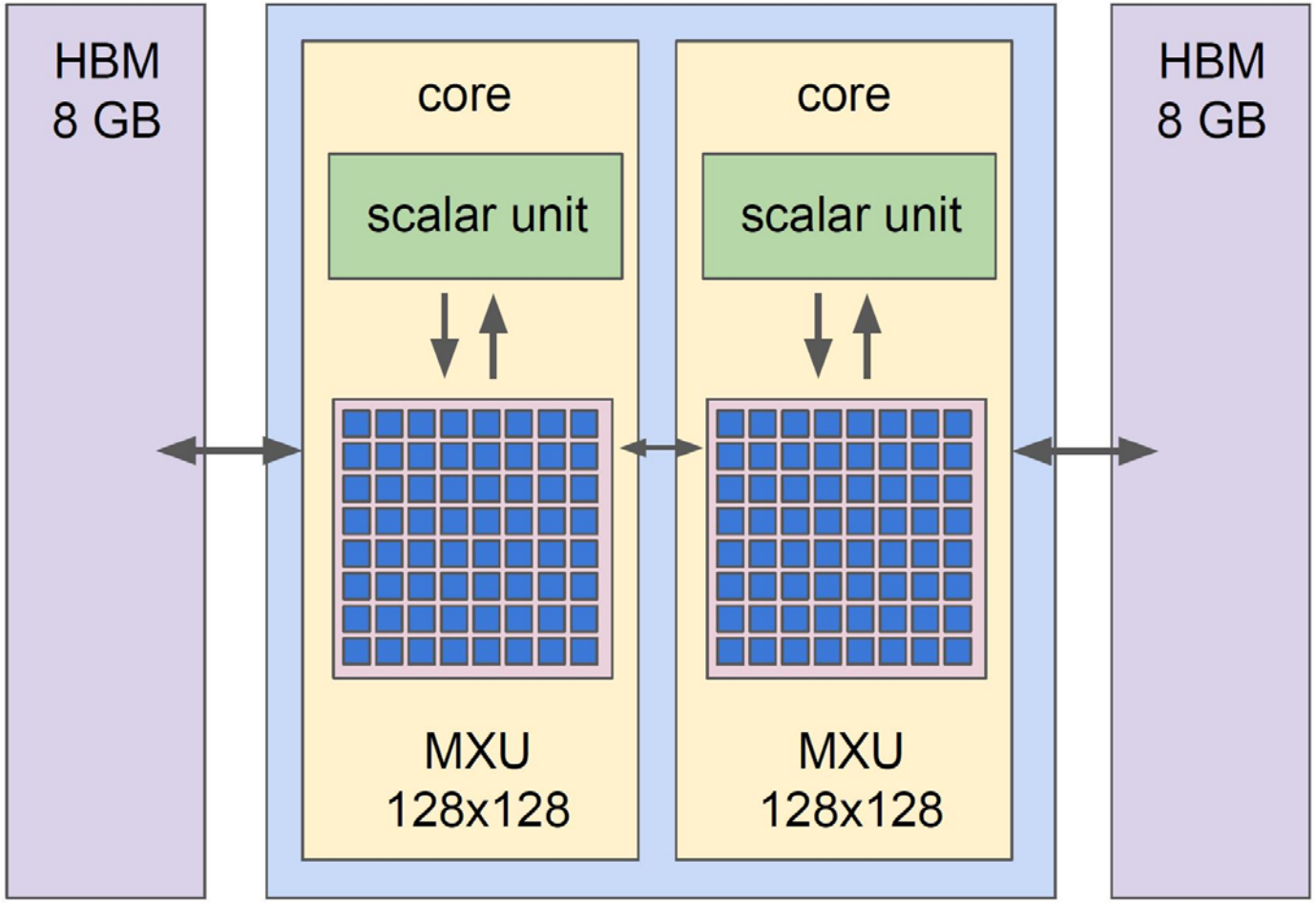
*full GV100 chip contains 84 SMs

https://www.servethehome.com/nvidia-v100-volta-update-hot-chips-2017/

TPU2: GEMM SYSTOLIC ARRAYS

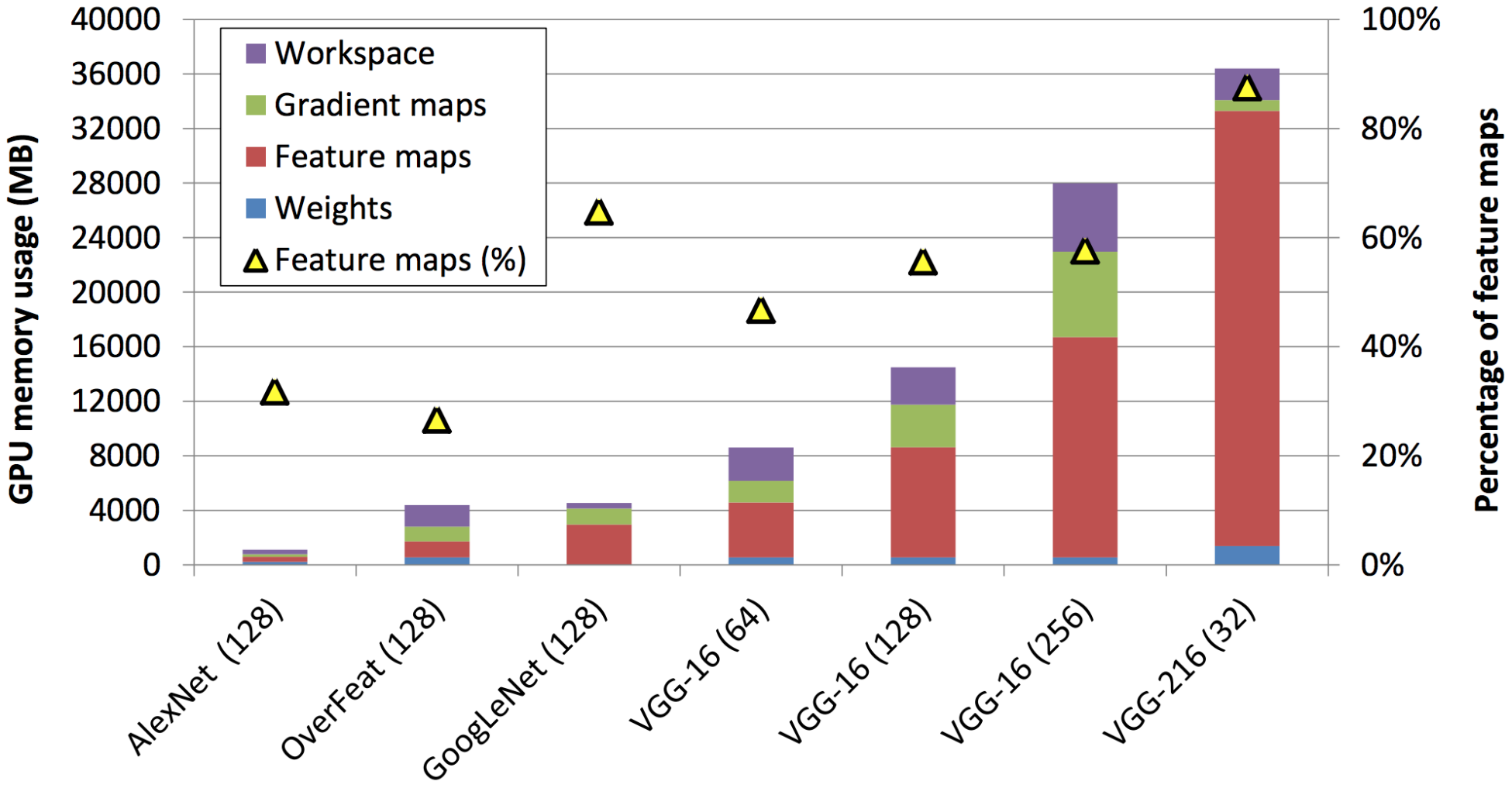


- 16 GB of HBM
- 600 GB/s mem BW
- Scalar unit: 32b float
- MXU: 32b float accumulation but reduced precision for multipliers
- 45 TFLOPS



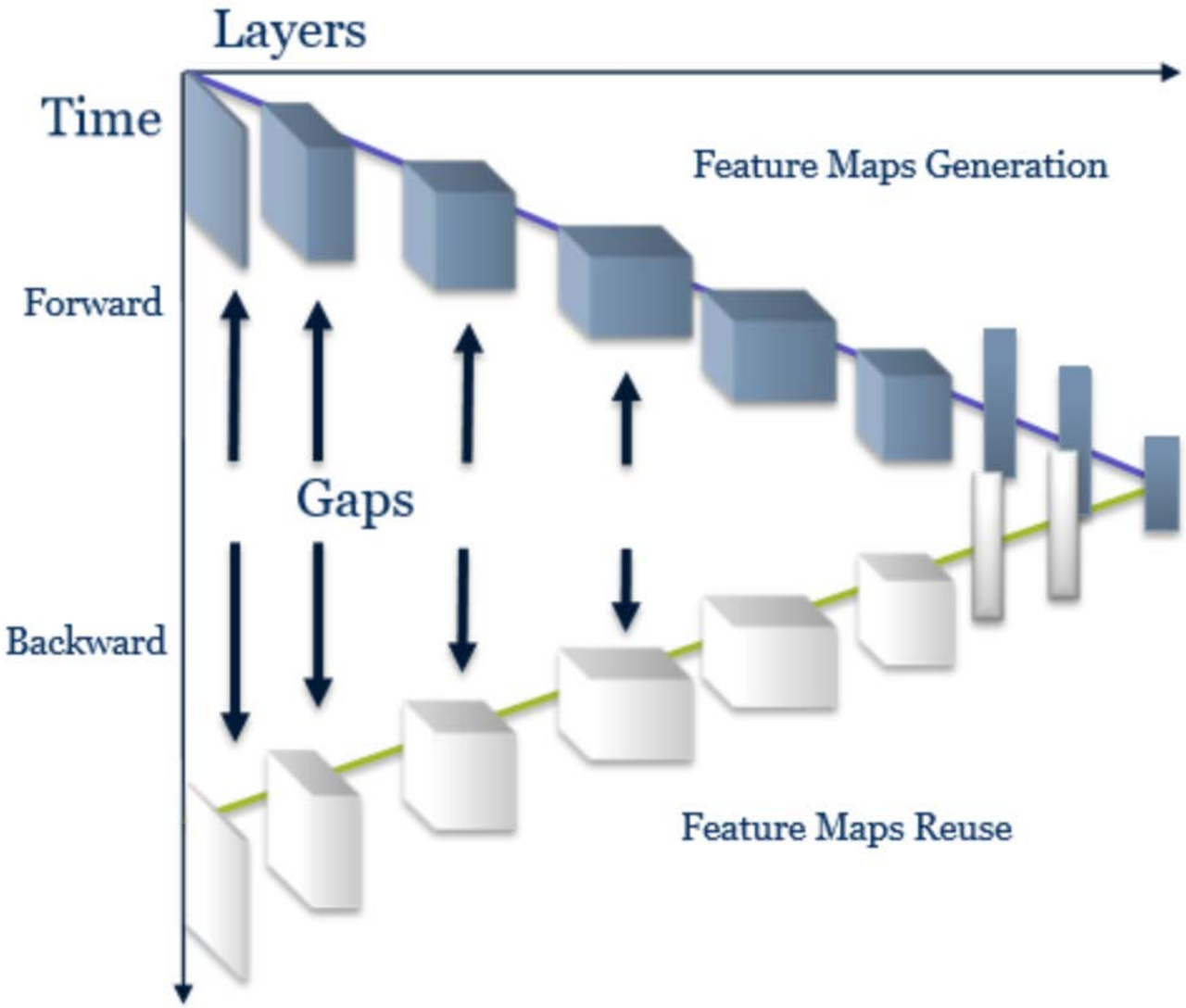
https://www.matroid.com/scaledml/2018/jeff.pdf

ACTIVATIONS TAKE MOST OF THE MEMORY



Minsoo Rhu, Natalia Gimelshein, Jason Clemons, Arslan Zulfiqar, Stephen W. Keckler, "vDNN: Virtualized Deep Neural Networks for Scalable, Memory-Efficient Neural Network Design" IEEE MICRO-49, 2016. https://arxiv.org/pdf/1602.08124.pdf

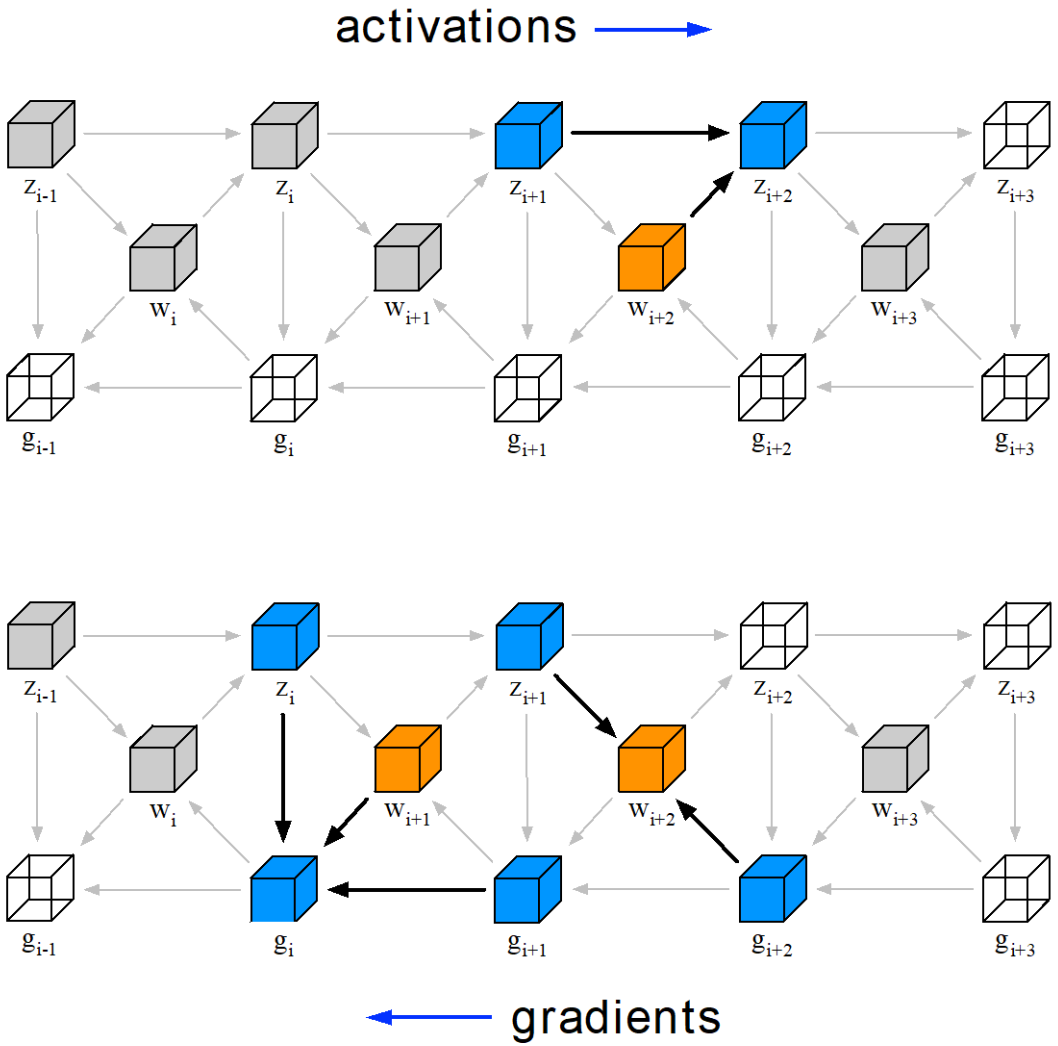
STORAGE IS QUADRATIC FUNCTION OF NETWORK DEPTH



Hongyu Zhu, "How to Train a Very Large and Deep Model on One GPU?", April 2017.
https://medium.com/syncedreview/how-to-train-a-very-large-and-deep-model-on-one-gpu-7b7edfe2d072

RECOMPUTE WHAT YOU CAN'T REMEMBER

- Recursive Checkpointing
- Recompute the Activations from sparse snapshots
- Trade most storage for one repeat of forwards pass compute



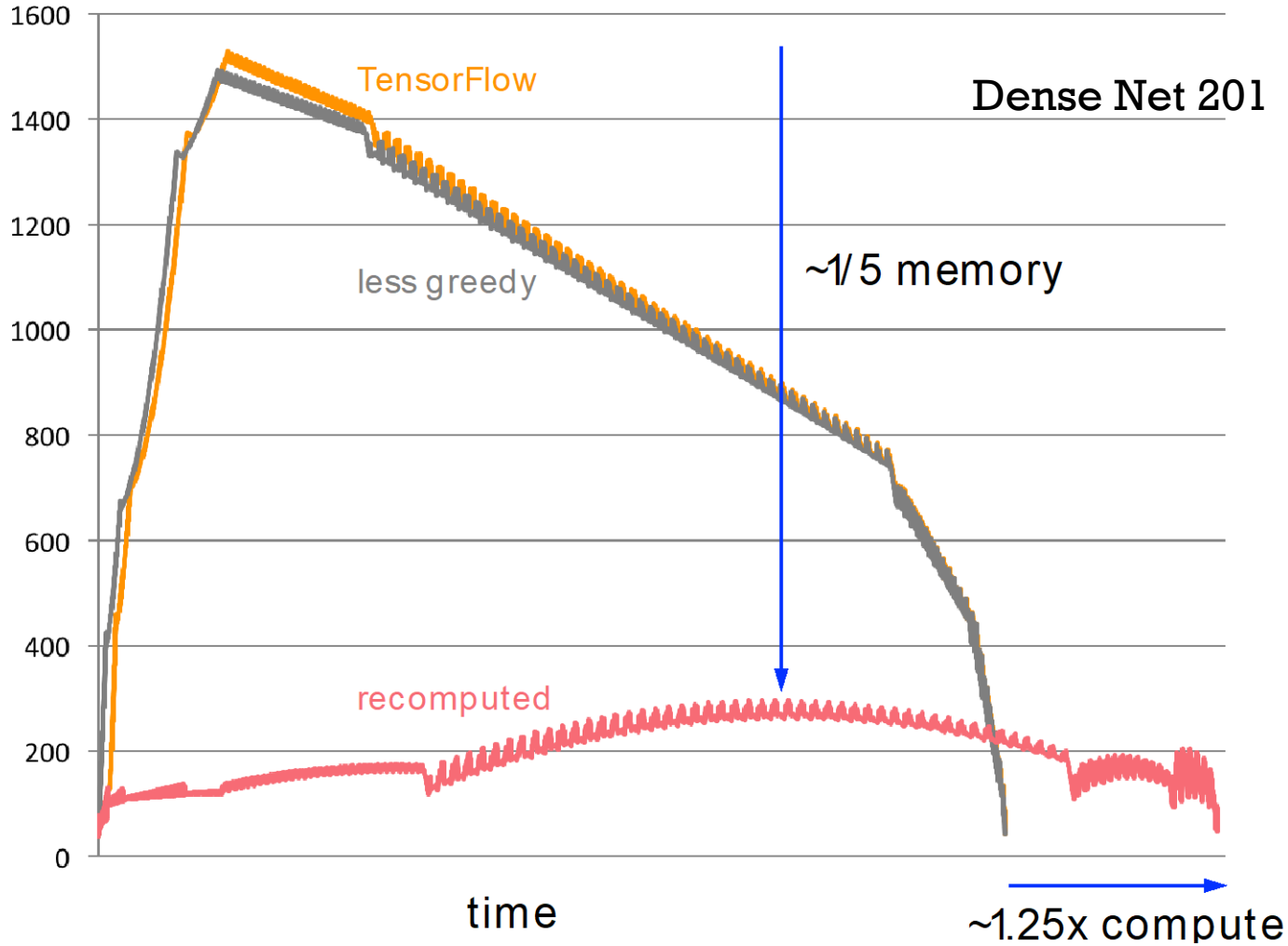
Simon Knowls, "Graphcore Intelligent Processing Unit (IPU)" Deep Learning at Supercomputer Scale Workshop, NIPS-17. https://www.matroid.com/scaledml/2018/simon.pdf

25% MORE COMPUTE SAVES 80% MEMORY

Naive strategy: memorize activations only at input of each residue block

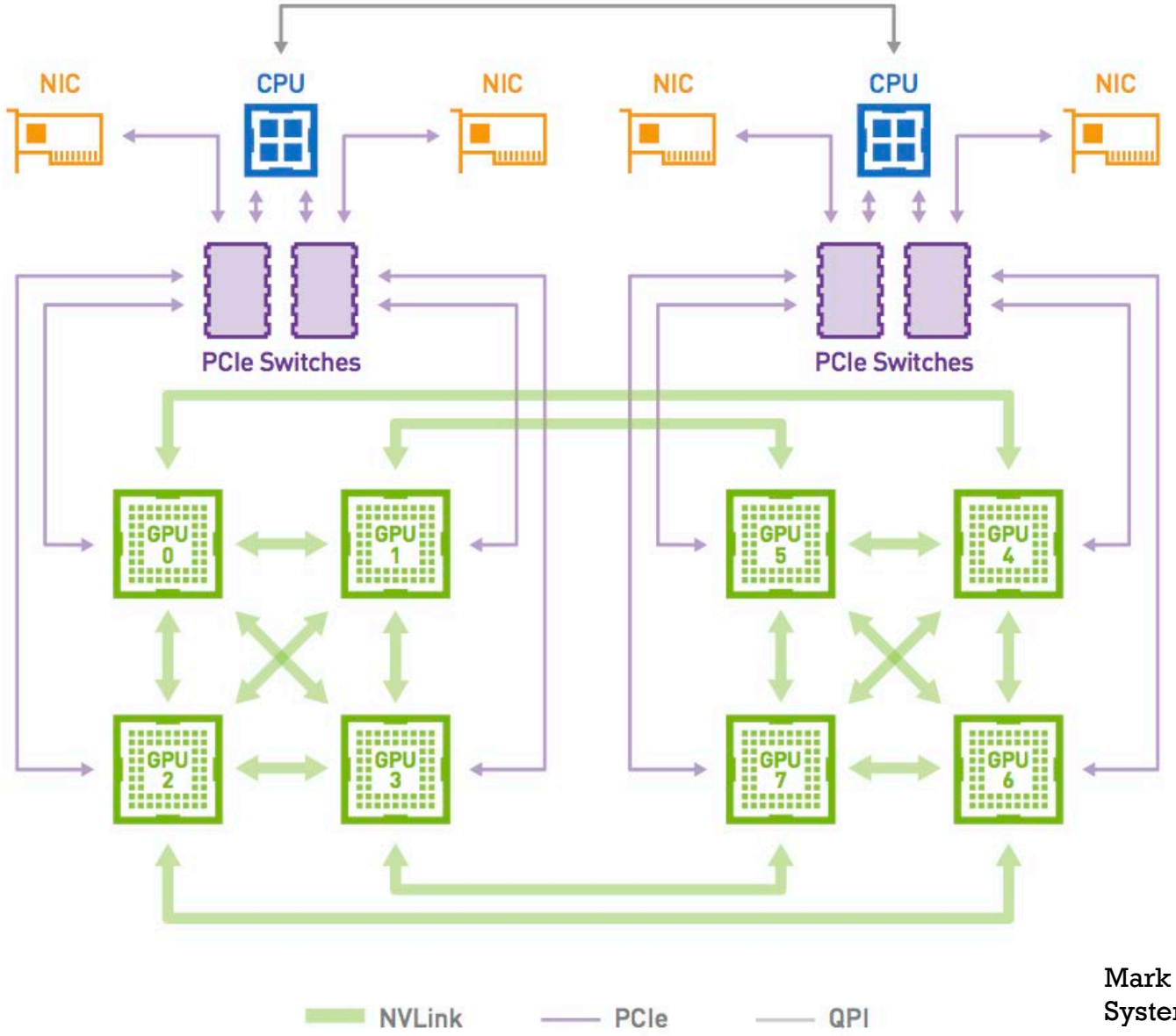
Batch=16 executing on CPU, recording total memory allocated for weights + activations.

f16 weights and activations, single weight copy.



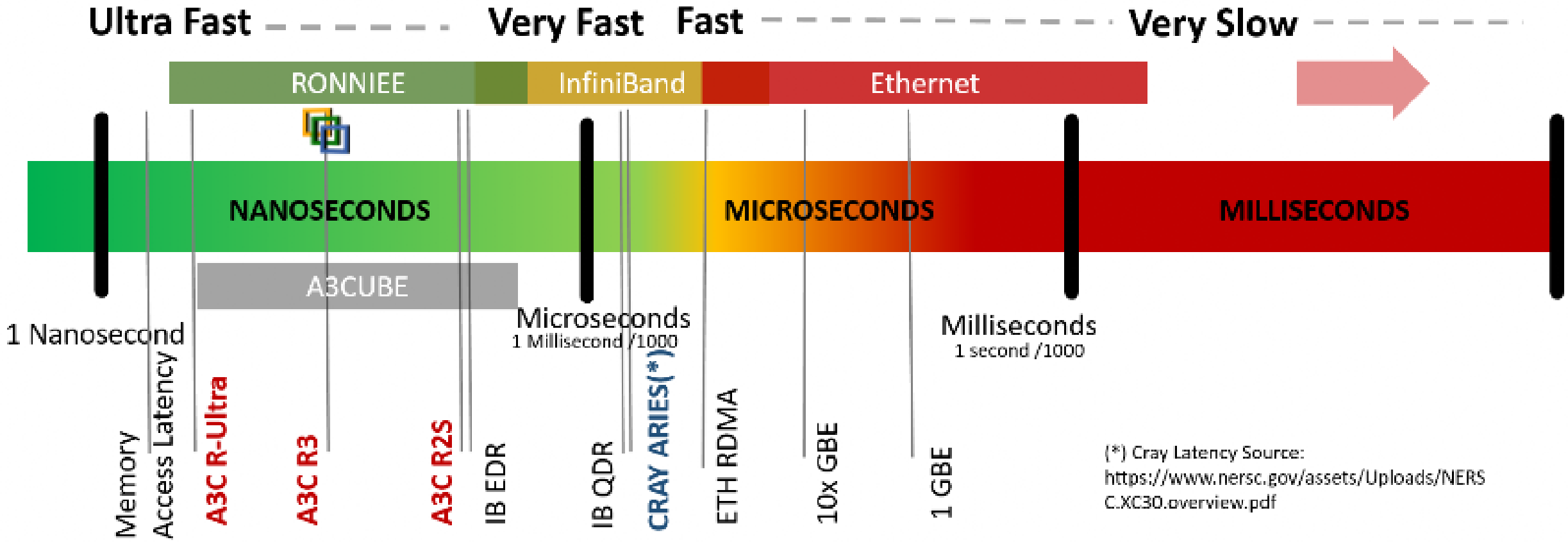
Simon Knowls, "Graphcore Intelligent Processing Unit (IPU)" Deep Learning at Supercomputer Scale Workshop, NIPS-17. https://www.matroid.com/scaledml/2018/simon.pdf

SCALING THE SYSTEM



Mark Harris, "NVIDIA DGX-1: The Fastest Deep Learning System", April 2017.

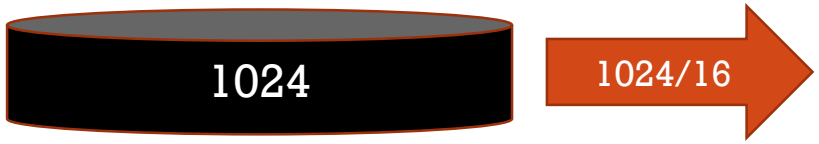
LATENCY IS A BOTTLENECK

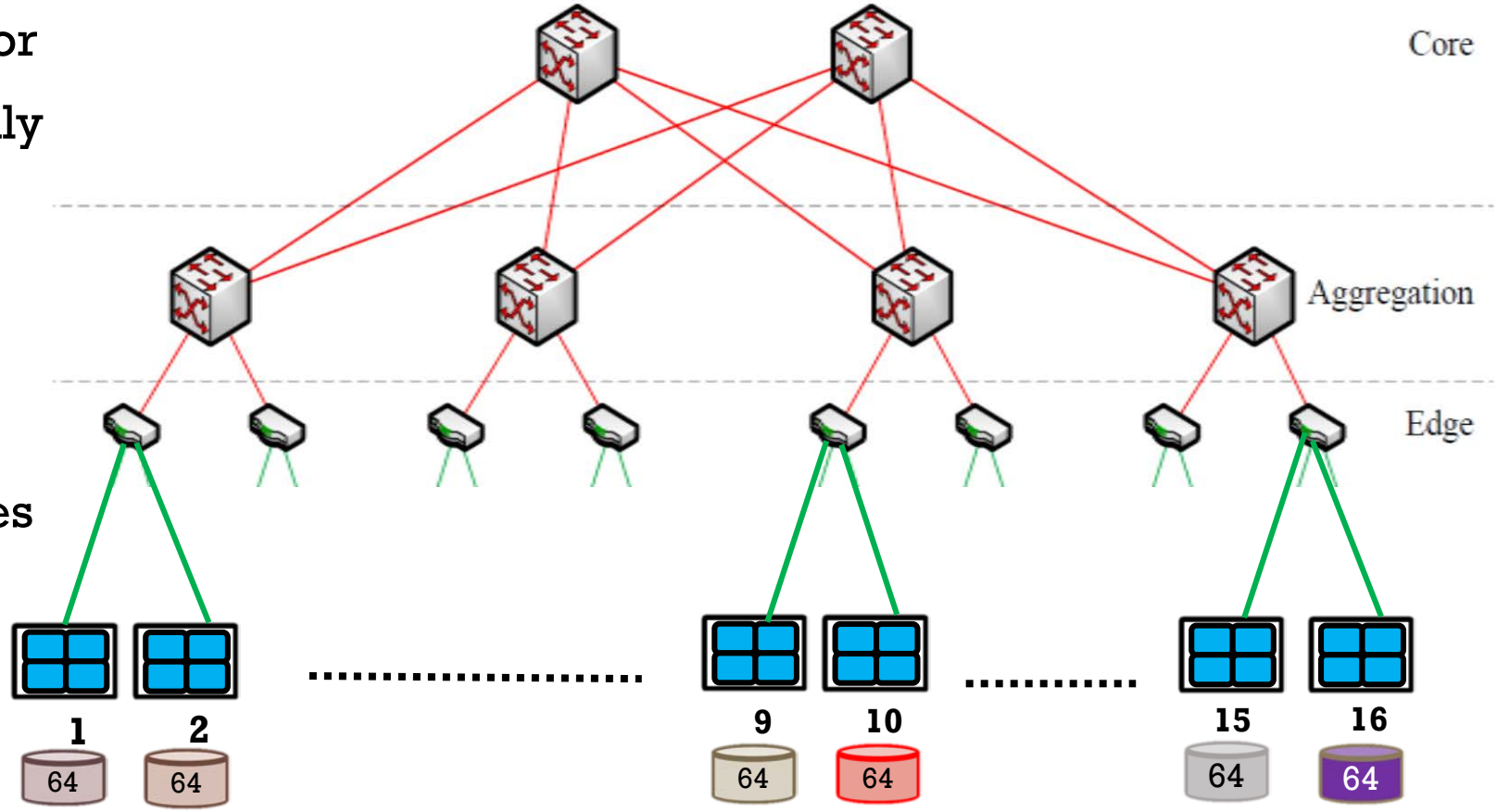


A3CUBE: "Latency Matters", © A3CUBE INC
http://www.a3cube-inc.com/-latency-matters.html

DISTRIBUTED SGD EXPLOITS DATA PARALLELISM

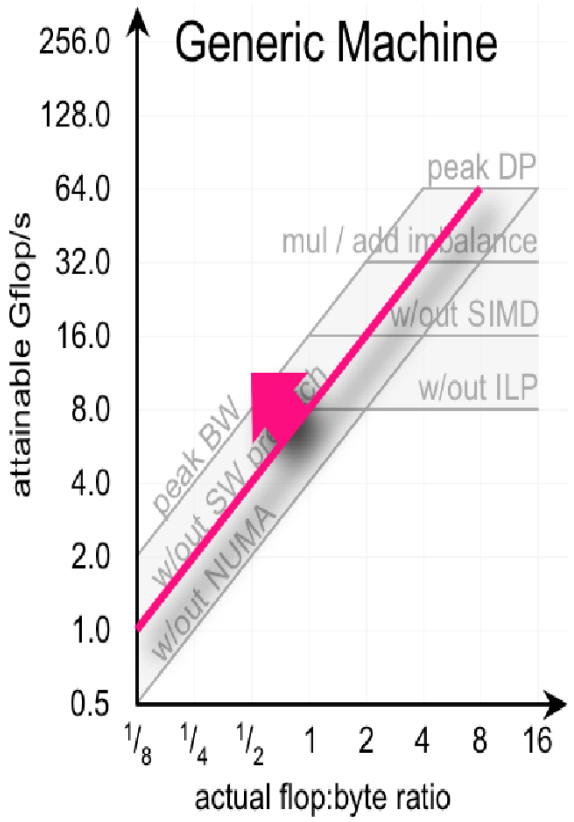
- Entire model on each processor
- Distribute the SGD batch evenly across each processor
  (aka per-processor batch):
  - 1024 batch distributed over 16 PEs
  - Batch of 64/PE
- Communicate gradient updates all-to-all





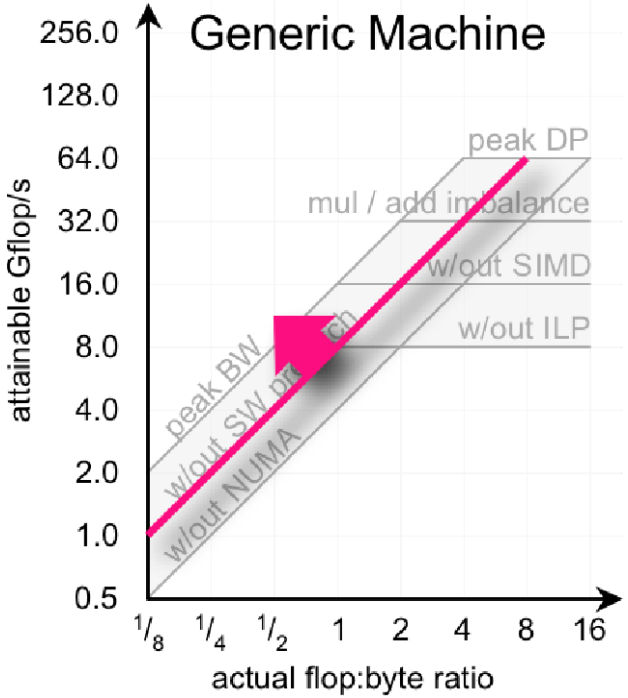
(Data center picture from)
Mohammad Al-Fares, Alexander Loukissas, Amin Vahdat,
"A scalable, commodity data center network architecture" ACM SIGCOMM 2008 conference on Data communication.

ROOFLINE MODEL FOR DISTRIBUTED TRAINING



(b) maximizing bandwidth

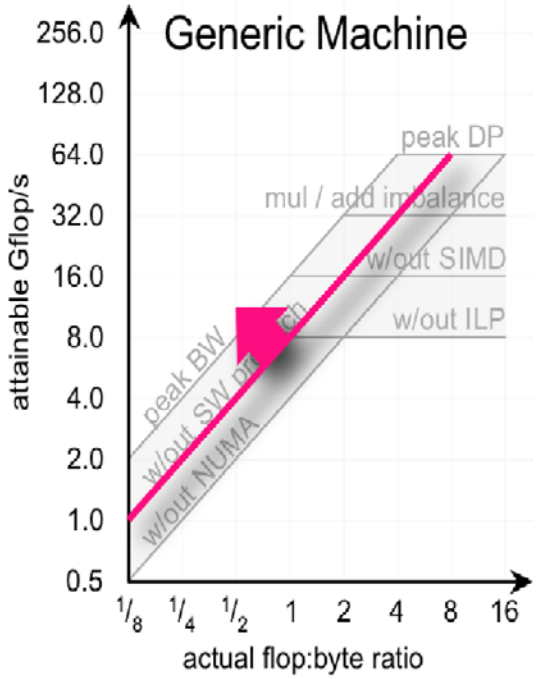
Use accelerators



(b) maximizing bandwidth

Interconnect network BW

DRAM BW



(b) maximizing bandwidth

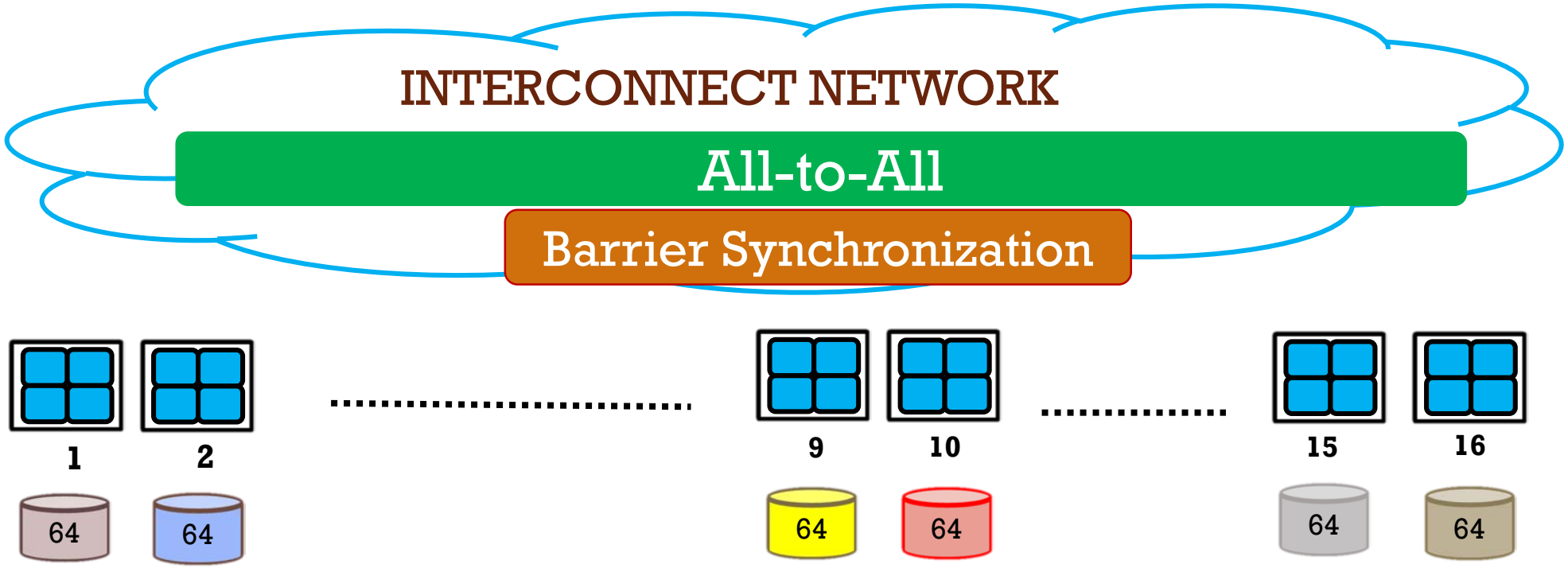
Node locality

Exploit sparsity

Less comm / synch

Samuel Williams, Andrew Waterman, and David Patterson, "Roofline: An Insightful Visual Performance Model for Floating-Point Programs and Multicore Architectures". Communications of the ACM April 2008.

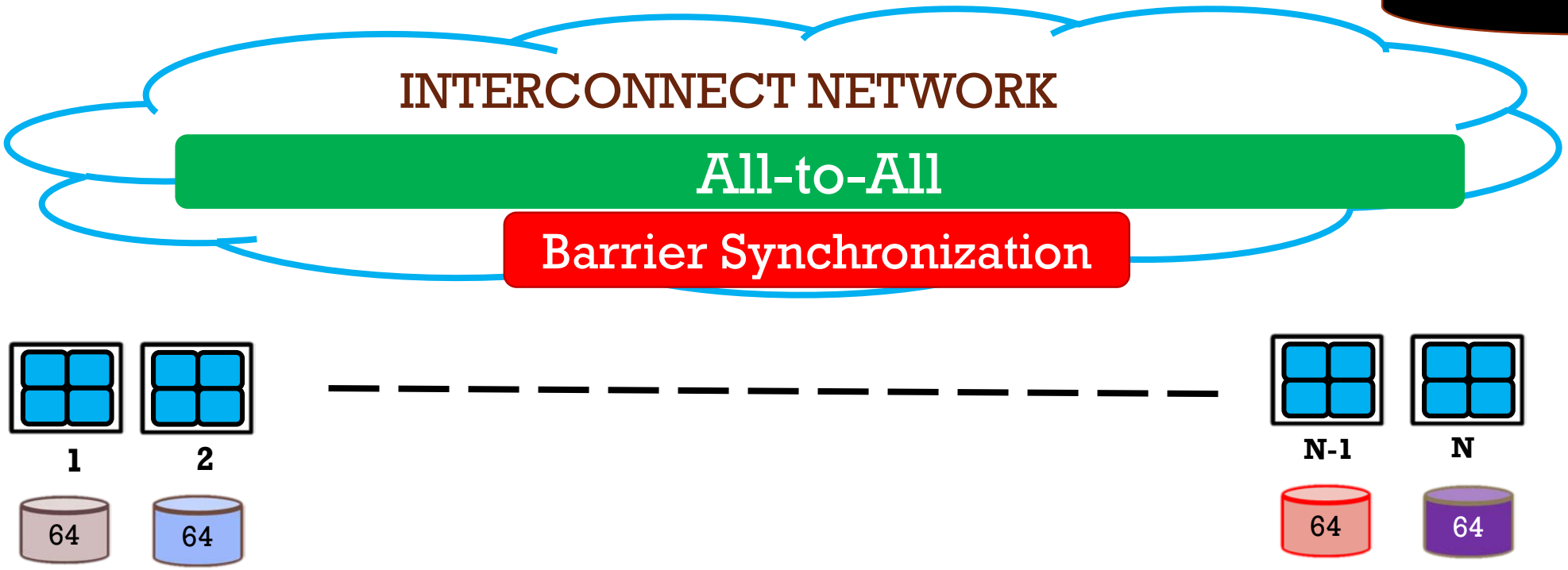
BULK SYNCHRONOUS SGD



- Synchronization bottleneck
- Various approach to ameliorate this but the problem is inherent

Rogers, R. O., and David B. Skillicorn. "Using the BSP cost model to optimise parallel neural network training." *International Parallel Processing Symposium*. Springer Berlin Heidelberg, 1998.

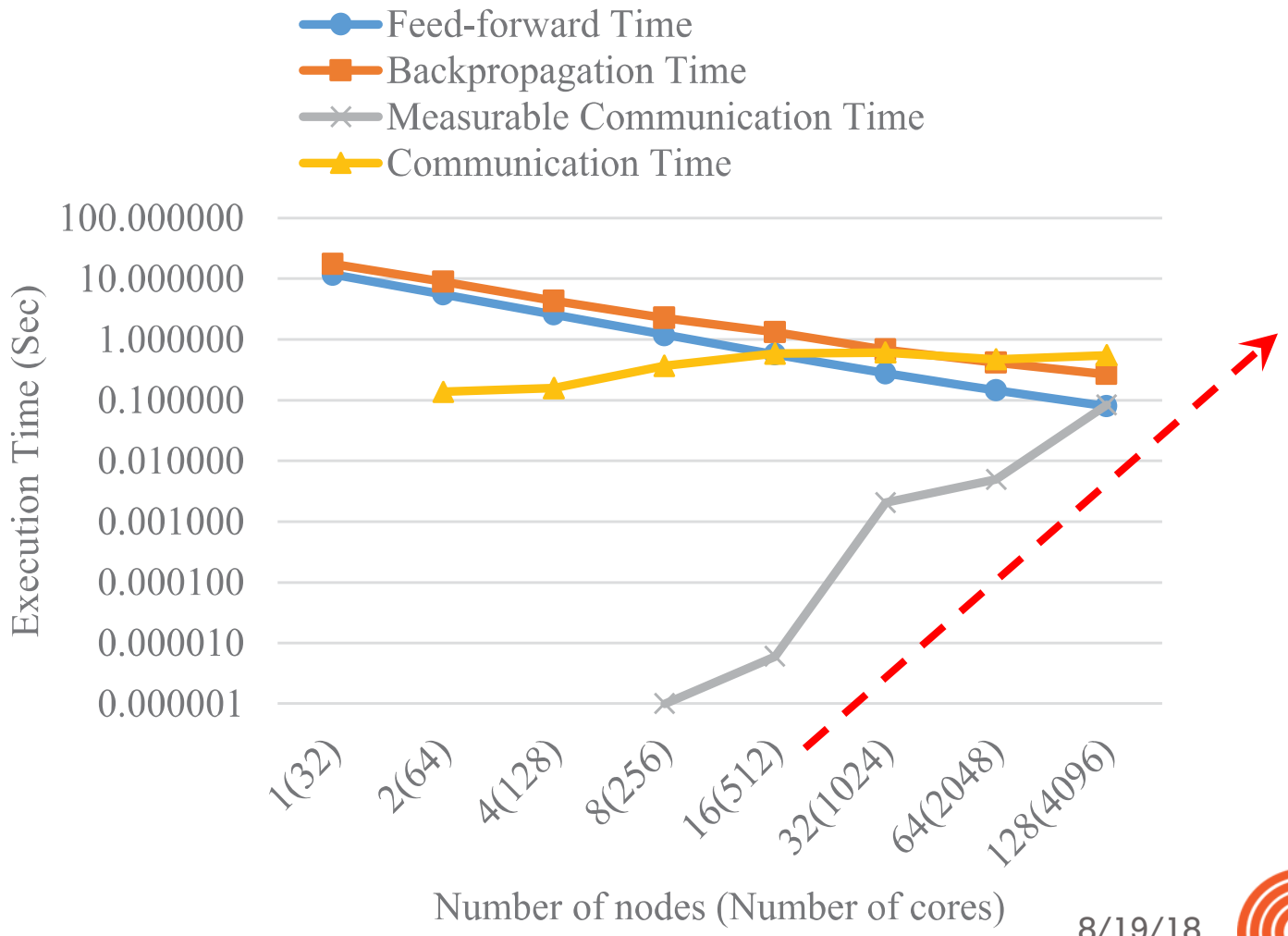
WE MUST INCREASE THE BATCH SIZE



- If we want to keep scaling synchronous SGD then we have to keep increasing the batch size
- N=256 -> Batch Size=**16K**

OVERLAP COMMUNICATION AND COMPUTE TO HIDE NETWORK LATENCY

- Breakdown for VGG
- Minibatch 256

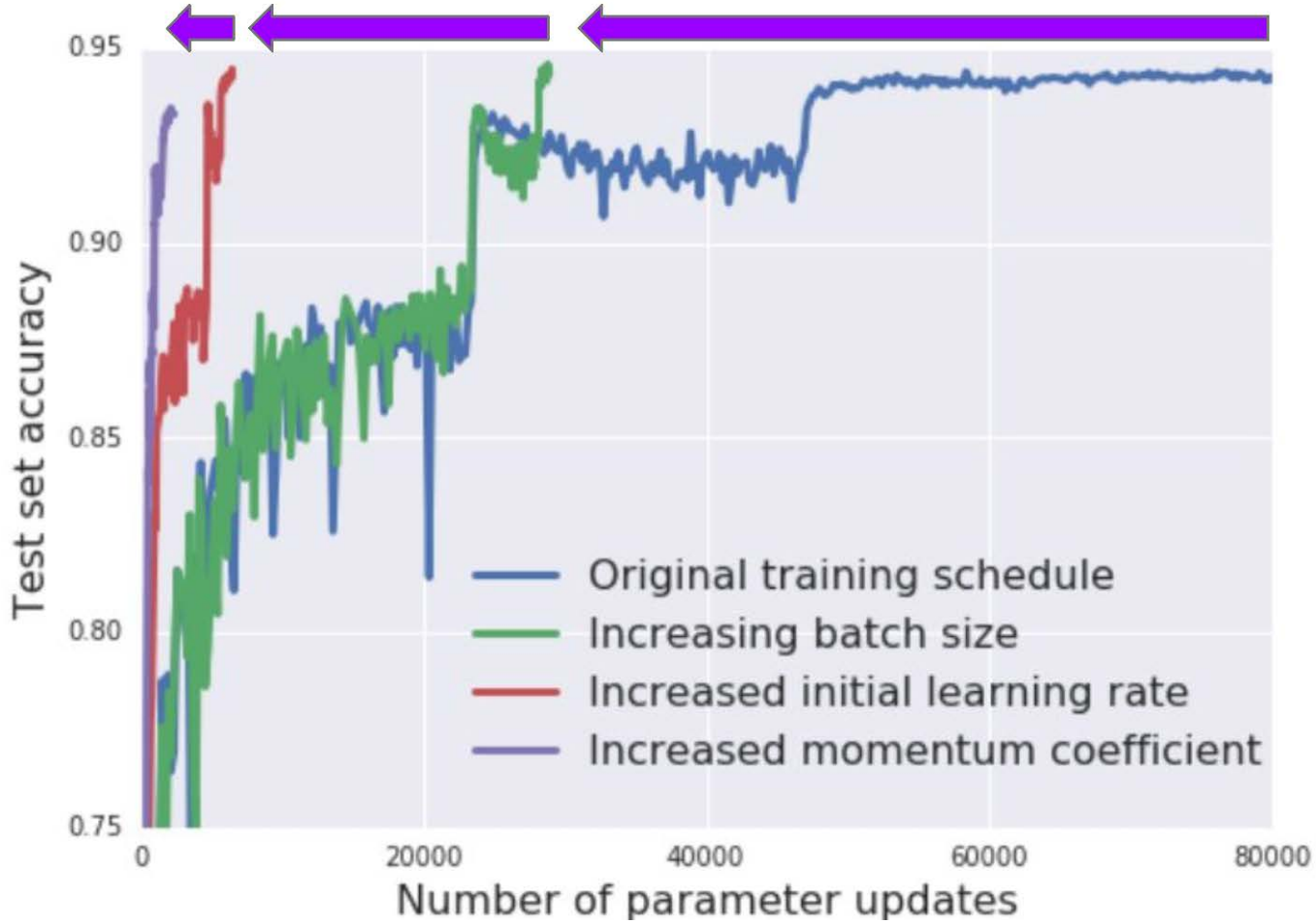


Sunwoo Lee, Dipendra Jha, Ankit Agrawal, Alok Choudhary, and Wei-keng Liao, “Parallel Deep Convolutional Neural Network Training by Exploiting the Overlapping of Computation and Communication “, IEEE 24th International Conference on High Performance Computing, 2017.

Das, Dipankar, Sasikanth Avancha, Dheevatsa Mudigere, Karthikeyan Vaidynathan, Srinivas Sridharan, Dhiraj Kalamkar, Bharat Kaul, and Pradeep Dubey. "Distributed deep learning using synchronous stochastic gradient descent." *arXiv preprint arXiv:1602.06709* (2016).

DON'T DECAY THE LEARNING RATE, INCREASE THE BATCH SIZE

- The key difficulty
  - Numerical optimization
- Decrease # of parameter updates
- Batch Size vs. learning rate
- CIFAR-10 & Imagenet
- Not general enough

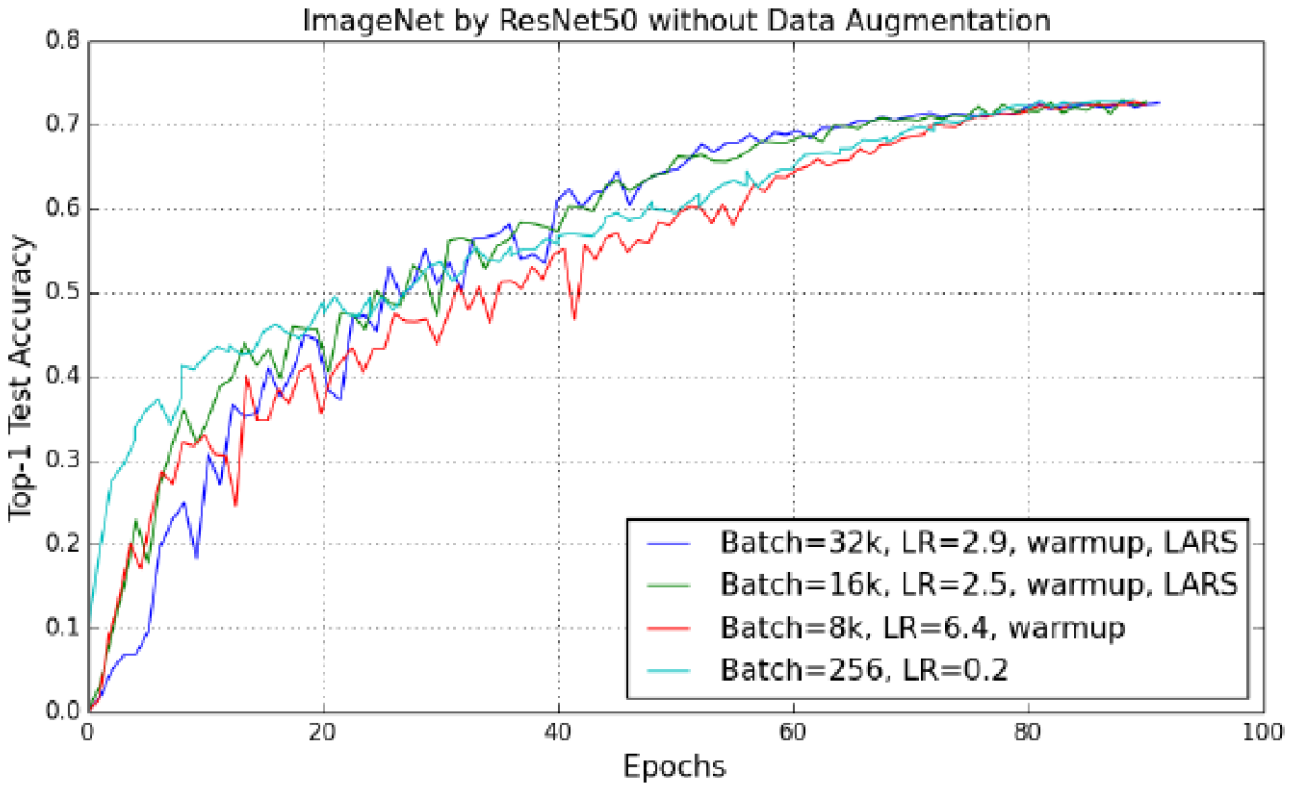


Samuel L. Smith, Pieter-Jan Kindermans, Chris Ying, Quoc V. Le," Don't Decay the Learning Rate, Increase the Batch Size," ICLR 2018. https://arxiv.org/abs/1711.00489

LAYER-WISE ADAPTIVE LEARNING RATE

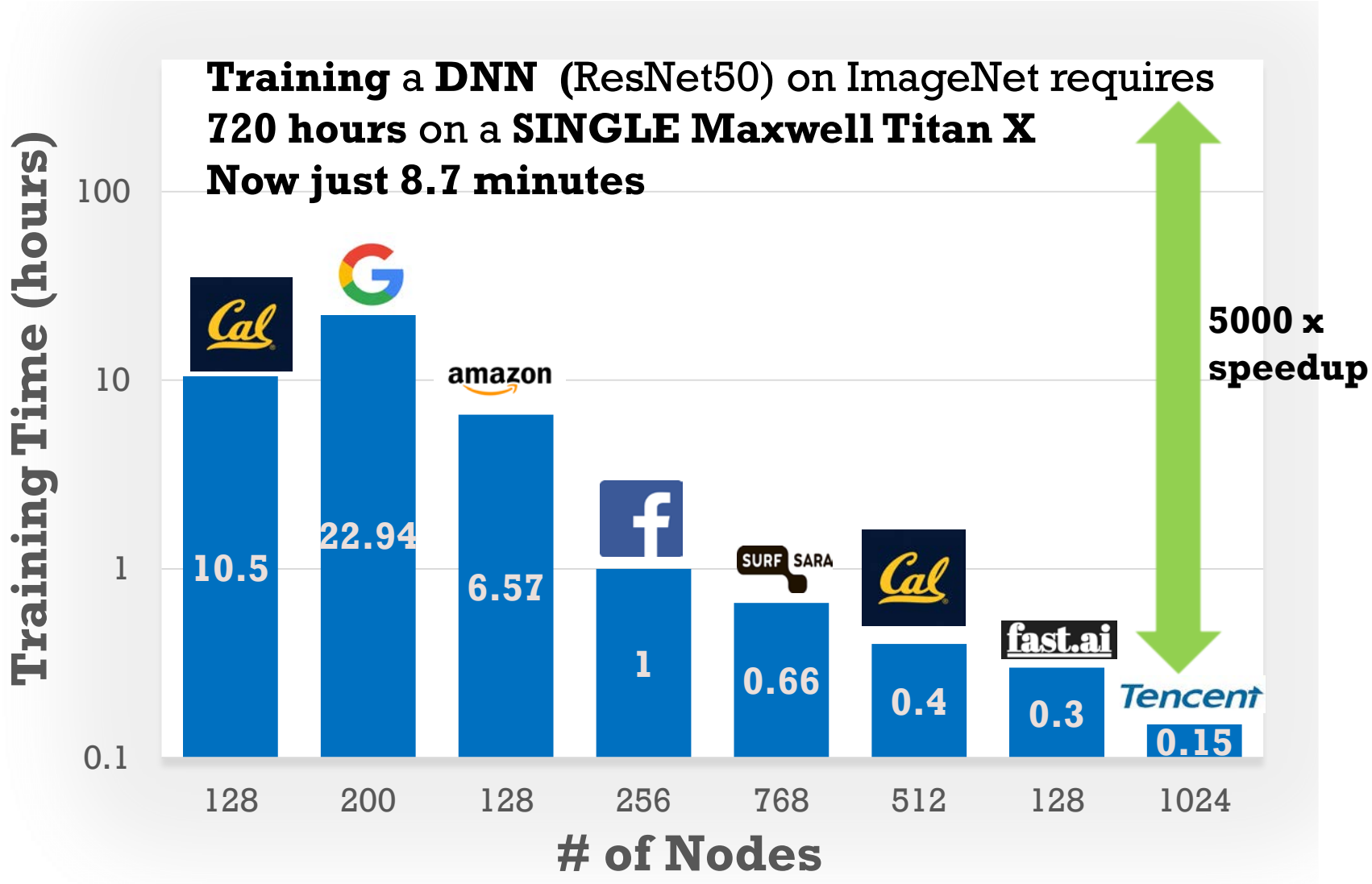
RESNET-50 WITH LARS: B→ 32K

- LARS:
  - Adapts the learning rate for each layer
  - Scaling to
    B=8K for Alexnet
    B=32K for Resnet-50.
- Above 32K without accuracy loss is still open problem

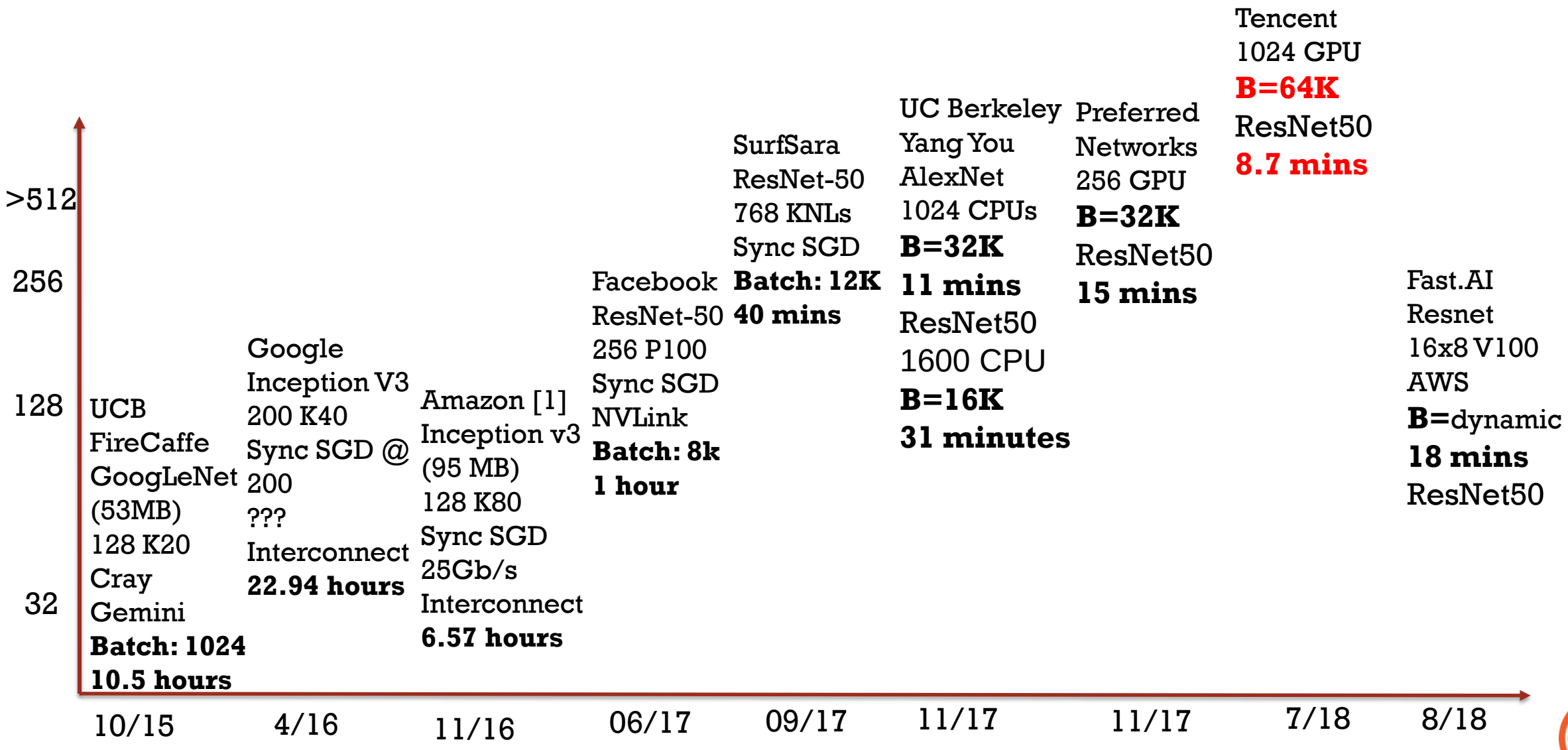


- You, Yang, Igor Gitman, and Boris Ginsburg. "Scaling SGD Batch Size to 32K for ImageNet Training." *arXiv preprint arXiv:1708.03888* (2017).
- You, Yang, Zhao Zhang, C. Hsieh, James Demmel, and Kurt Keutzer. "ImageNet training in minutes." ICPP 2018. https://arxiv.org/abs/1709.05011

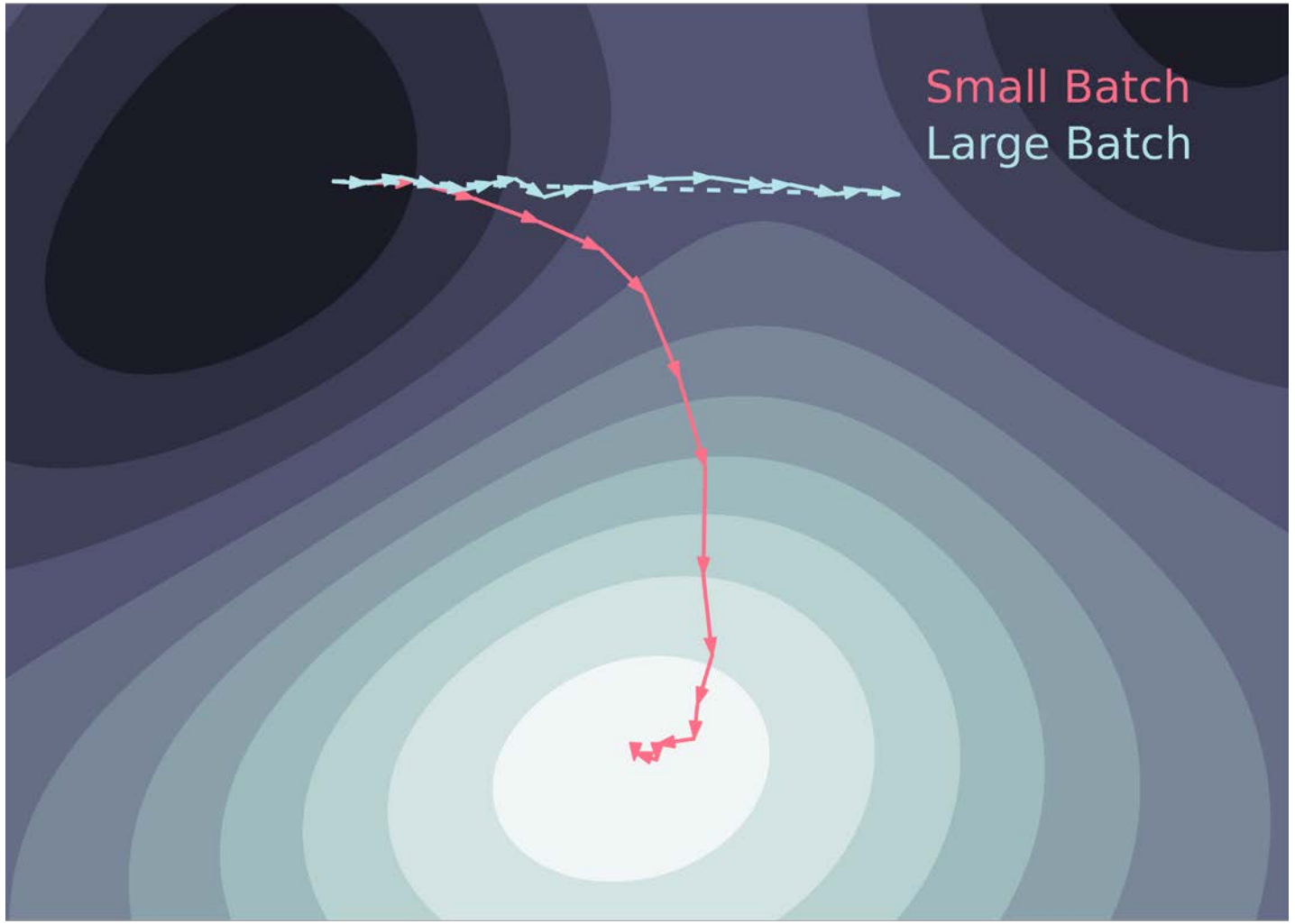
RECENT PROGRESS



PUSHING SYNCHRONOUS SGD FARTHER

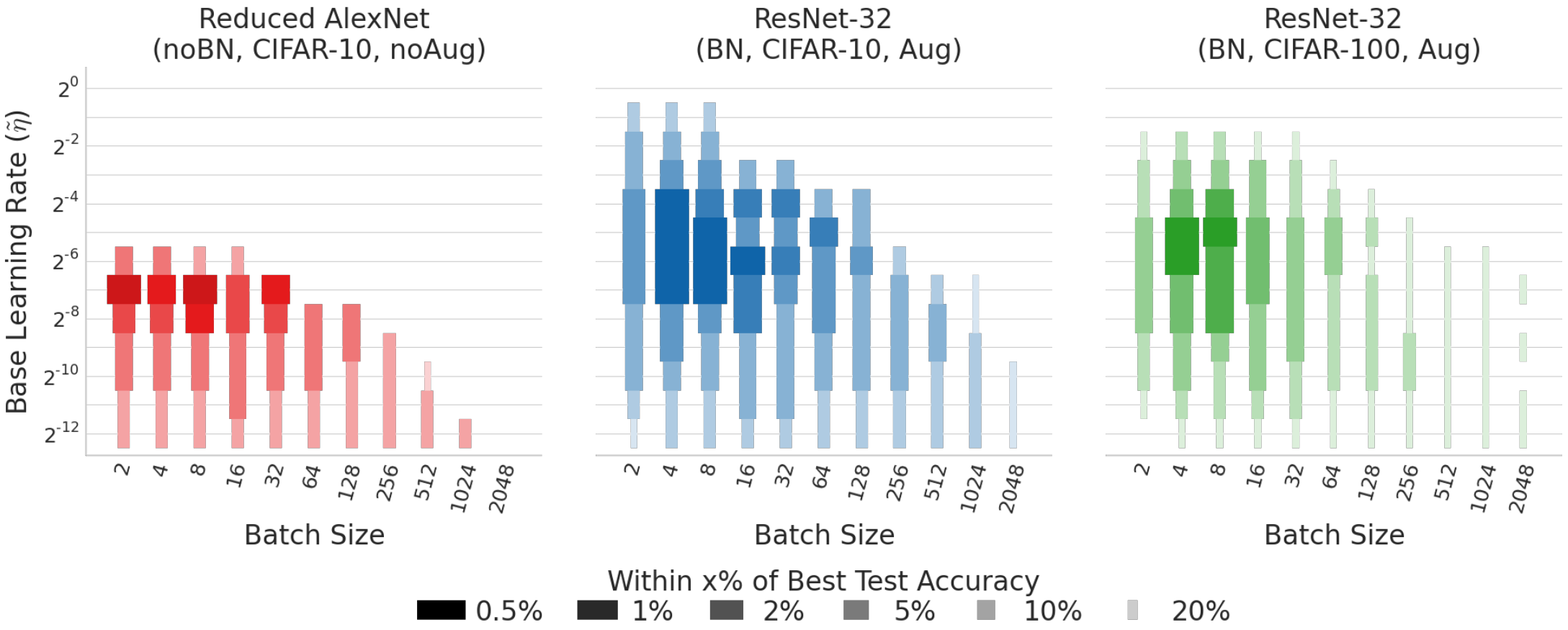


LARGE BATCHES MISS THE MINIMUM



Dominic Masters, Carlo Luschi, "Revisiting Small Batch Training for Deep Neural Networks" arXiv:1804.07612 2017.
https://www.graphcore.ai/posts/revisiting-small-batch-training-for-deep-neural-networks

LARGE BATCHS MISS TEST ACCURACY



Dominic Masters, Carlo Luschi, "Revisiting Small Batch Training for Deep Neural Networks" arXiv:1804.07612 2017.
https://arxiv.org/abs/1804.07612

SCALING TRAINING WITH LARGE BATCHES

Cons

- Constrained approach: Need to employ large batches to capture efficiency
- May not achieve target accuracy
- Only demonstrated on CNNs
- More prone to adversarial attacks[1]

Pros

- Robust: Relatively less hyperparameter tuning
- Sequential consistency
- Good infrastructural support from HPC frameworks (MPI)
- Fault tolerance is practically handled by snapshots and rollback otherwise

[1] Zhewei Yao, Amir Gholami, Qi Lei , Kurt Keutzer, Michael Mahoney,
"Hessian-based Analysis of Large Batch Training and Robustness to Adversaries", arXiv:1802.08241v.
https://arxiv.org/pdf/1802.08241.pdf

ASYNCHRONOUS SGD

- Hogwild![1]
  - Asynchronous on shared memory
- Distributed asynchronous SGD
  - Googles training (Dist belief)[2]
  - Accuracy has degraded at large scaling >32 [3,4]
- Deep gradient compression[5]

[1]- Feng Niu, Benjamin Recht, Christopher R´e and Stephen J. Wright, "Hogwild!: A Lock-Free Approach to Parallelizing Stochastic Gradient Descent", NIPS 2011. https://people.eecs.berkeley.edu/~brecht/papers/hogwildTR.pdf

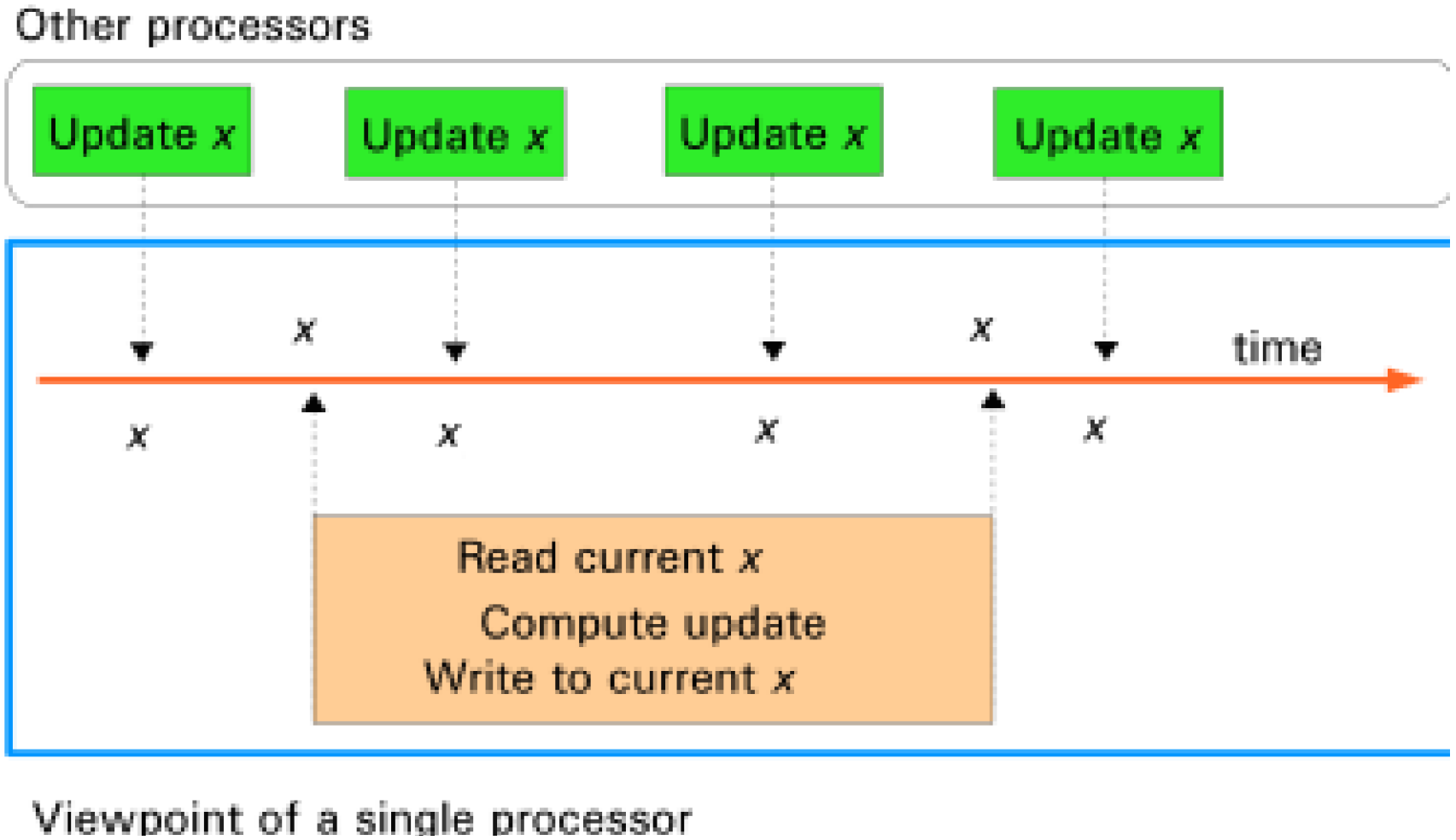
[2] - Jeffrey Dean, Greg S. Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Quoc V. Le, Mark Z. Mao, Marc'Aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, Andrew Y. Ng , "Large Scale Distributed Deep Networks". https://ai.google/research/pubs/pub40565

[3]- Zhang, Sixin, Anna E. Choromanska, and Yann LeCun. "Deep learning with elastic averaging SGD." In Advances in Neural Information Processing Systems, pp. 685-693. 2015.

[4]- Jin, Peter H., Qiaochu Yuan, Forrest Iandola, and Kurt Keutzer. "How to scale distributed deep learning?." arXiv preprint arXiv:1611.04581 (2016). NIPS MLSys 2017.

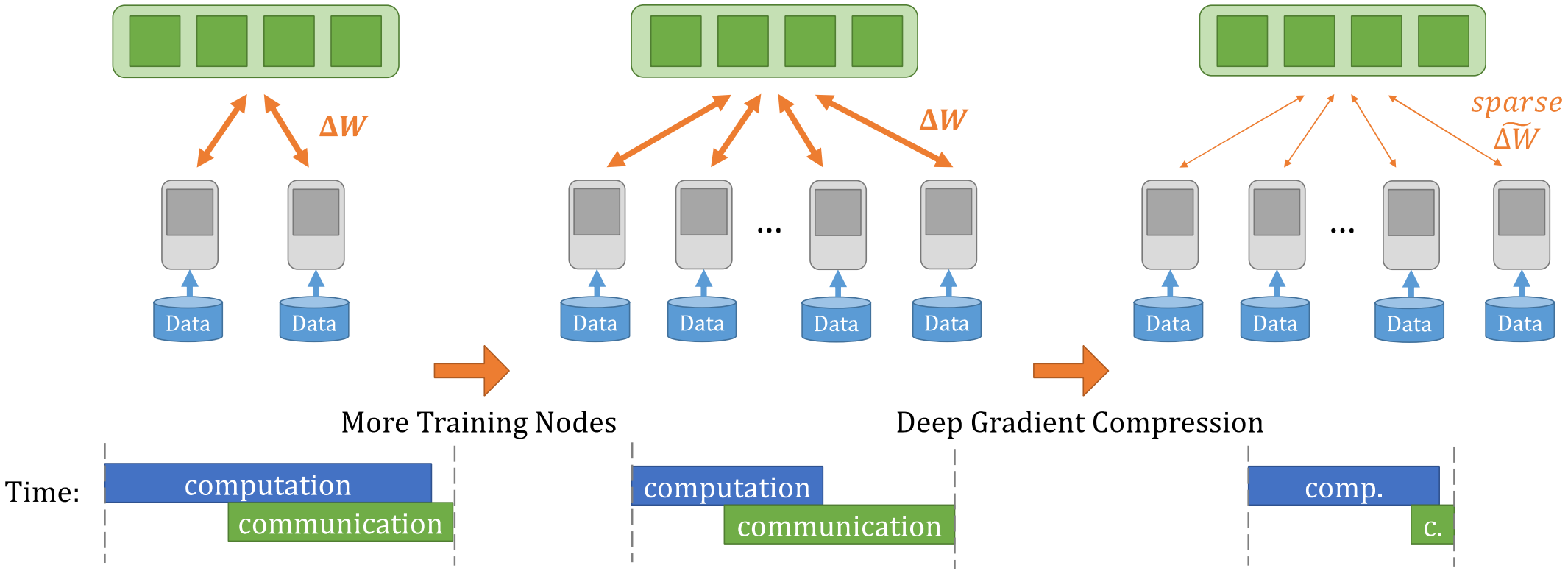
[5]- Yujun Lin, Song Han, Huizi Mao, Yu Wang, William J. Dally, "Deep Gradient Compression: Reducing the Communication Bandwidth for Distributed Training" ICLR 2018. https://arxiv.org/abs/1712.01887

HOGWILD!



Feng Niu, Benjamin Recht, Christopher R´e and Stephen J. Wright, "Hogwild!: A Lock-Free Approach to Parallelizing Stochastic Gradient Descent", NIPS 2011. https://people.eecs.berkeley.edu/~brecht/papers/hogwildTR.pdf

DEEP GRADIENT COMPRESSION

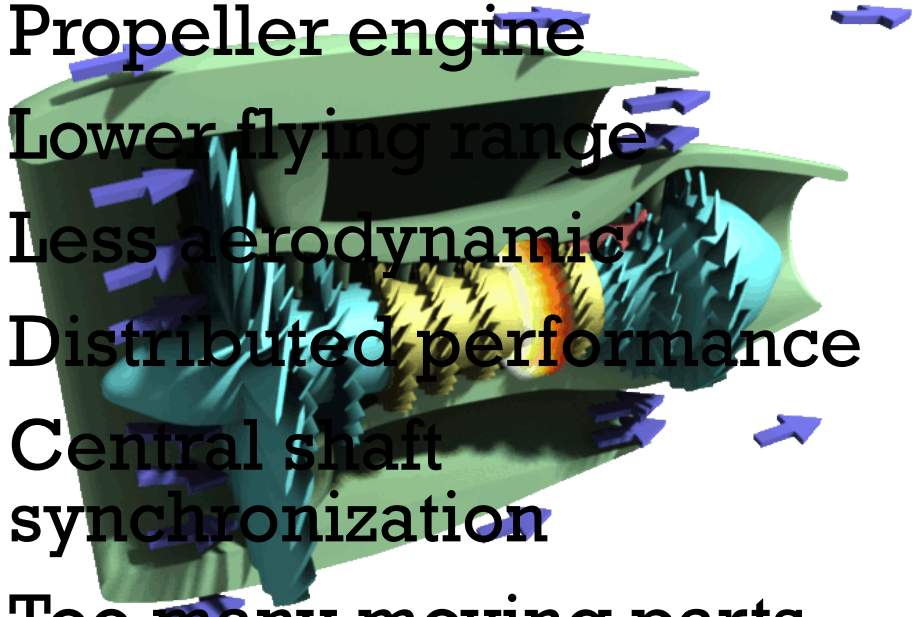


Yujun Lin, Song Han, Huizi Mao, Yu Wang, William J. Dally, "Deep Gradient Compression: Reducing the Communication Bandwidth for Distributed Training" ICLR 2018. https://arxiv.org/abs/1712.01887

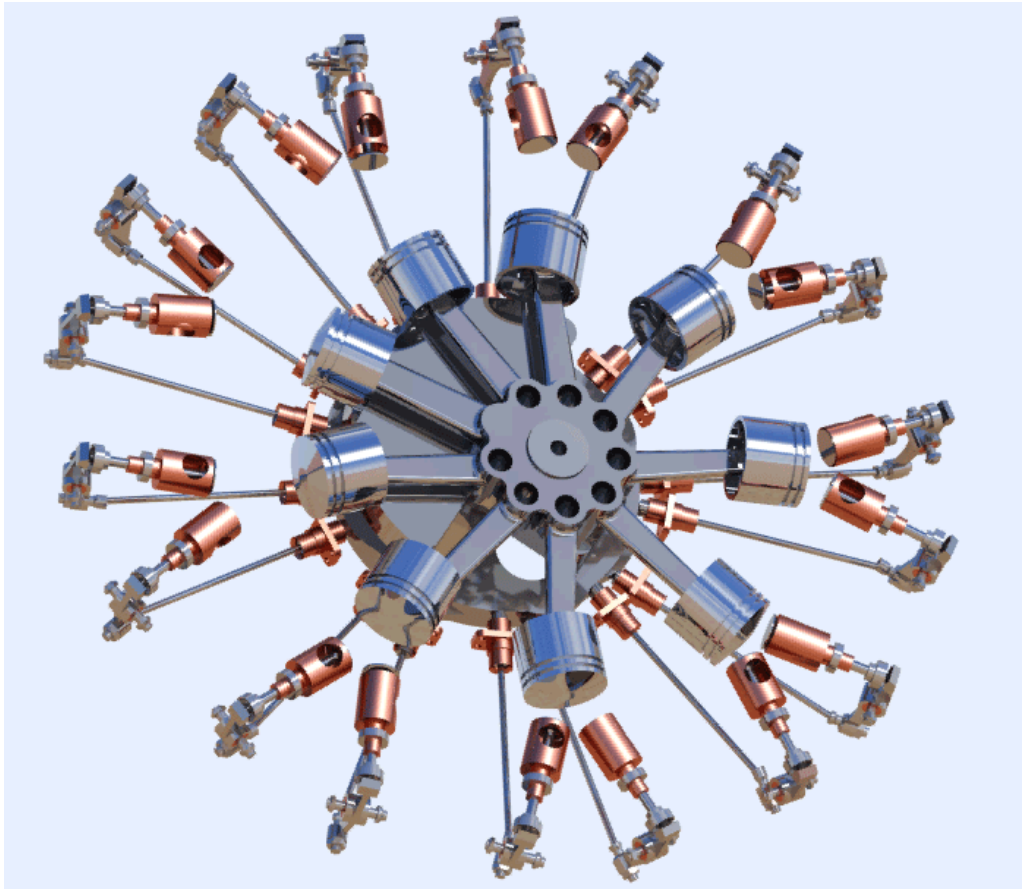
“If we all worked on the assumption that what is accepted as true is really true, there would be little hope of advance.”

Orville Wright

Radial Engine



- Propeller engine
- Lower flying range
- Less aerodynamic
- Distributed performance
- Central shaft synchronization
- Too many moving parts



1. https://imgur.com/gallery/79Qo0/comment/12698133
2. By RichardWheeler from wikipedia

FUTURE OF ACCELERATED TRAINING

- Increase physical scale to support model parallelism
- Architectures to improve communication and memory bandwidth
- Dedicated silicon area to neural network compute
- Exploit sparsity

Keep an eye out for companies that will contribute to cloud training




BENCHMARKING ML

TYPES OF ML BENCHMARKS: QUALITY

- Quality Benchmarks: measure the accuracy of networks
  - **ImageNet**: Fei-Fei Li 2012, 1.2M image standard dataset for measuring classification accuracy as well as a yearly competition
    - Revolutionized image classification
  - **Mnist**: hand written digit dataset
  - **CIFAR**: small image classification
  - Many more: https://en.wikipedia.org/wiki/List_of_datasets_for_machine_learning_research#Object_detection_and_recognition
    - Image data
    - Text data
    - Sound data
    - Signal data
    - Physical data
    - Biological data
    - Anomaly data
    - Question Answering data
    - Multivariate data

TYPES OF ML BENCHMARKS: PERFORMANCE

- Performance Benchmarks: measure the speed of network execution
  - Frameworks and hardware are both measured
    - Inference: throughput, latency
    - Training: throughput, time to accuracy

Benchmark	Breadth of Types	Accuracy requirement	Support	Submission rules and publication of results
Conv bench	1	No	Individual	No
DeepBench	Kernels	No	Corporate	No
DAWN Bench	2	Yes	University	Minimal
Fathom	8	No	University	No
MLPerf	7	Yes	Industry	Extensive

LIES, DAMN LIES, ...

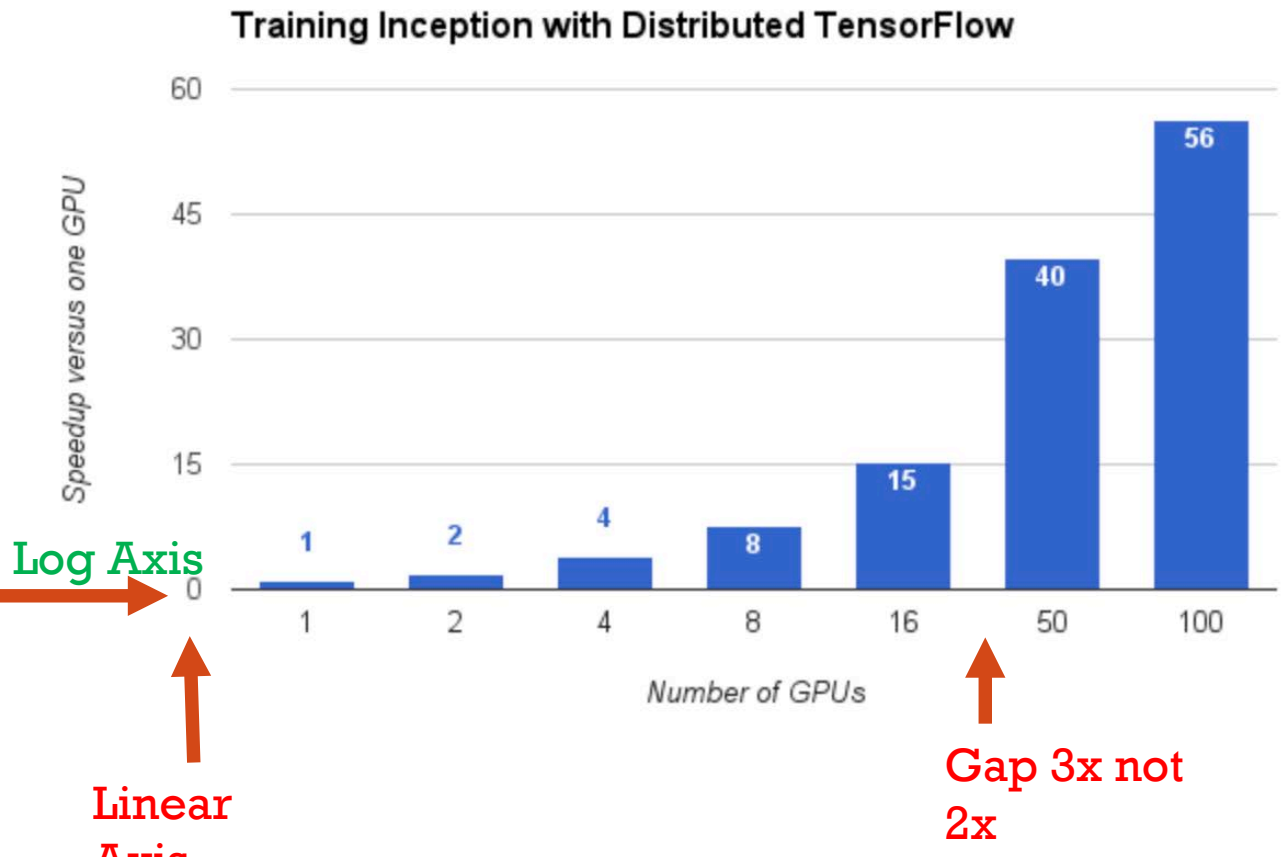
COMMON BENCHMARK CHEATS

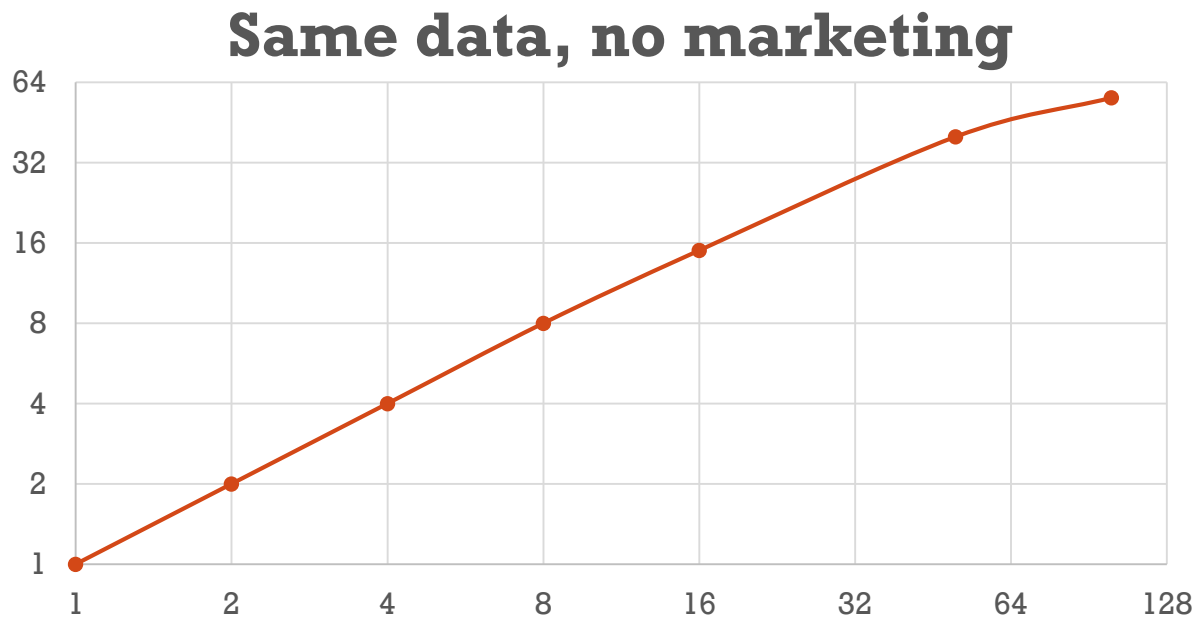
- Choosing thresholds that the comparison system cannot achieve
  - Latency cutoff of 7ms, comparison system has latency floor of 8 ms
- Cherry picking results
  - Run the benchmark 20 times and publish the best result
- Normalize to a metric of advantage even if scalability doesn't hold
  - If you don't win on performance compare performance/W, performance/$, performance/lb …

CHEATS UNIQUE TO ML

- ML is statistical compute, results are subject to variation
- Do massive search to find fastest training on benchmark data
  - Hyper-parameters: learning rate, batch size …
  - Fine grain verification to cherry pick first accuracy above threshold
  - Initialization seeds
- Techniques just ”game” for the benchmark data set and do not generalize

MISLEADING PRESENTATION





Derek Murray, "Announcing TensorFlow 0.8 – now with distributed computing support!", April 2016. https://ai.googleblog.com/2016/04/announcing-tensorflow-08-now-with.html

CONCLUSION: TODAY

- Scaling training is a matter of processor performance & communication latency
- Current accelerators:
  - GEMM centric to overcome memory wall
  - DNN frequently does not fit
  - Require large batch size to achieve high utilization/performance
- Scaling synchronous SGD requires large batch methods
  - Accuracy may require extensive hyperparameter tuning
  - Very large batch sizes only demonstrated on CNNs
- Minimize the impact of communication latency bottleneck
  - Use asynchronous approaches, but those have all negatively impacted accuracy
  - Hide communication latency by pipelining gradients
- Benchmarks emerging with some difficulty

CONCLUSION: FUTURE

- Massive multi-core engines that **enable model parallelism**
- Orders of magnitude **greater memory and communication BW**
- Unconstrained methods, e.g., large and **small mini-batch**
- Capture weight and activation **sparsity** for higher performance
- Support research and execution of **emergent model architectures** (not just those of today)

REFERENCES FOR NUMBERS (SLIDE 43)

2015/10: UC Berkeley: GoogleNet on 128 Nvidia K20's; Gemini Interconnect;

- Iandola FN, Ashraf K, Moskewicz MW, Keutzer K. **FireCaffe**: near-linear acceleration of deep neural network training on compute clusters. arXiv preprint arXiv:1511.00175. 2015 Oct 31. Also, In *Proceedings of CVPR 2016,* 2592-2060.

2016/02: Intel: VGG-A on 128 Intel Xeon E5; Aries Dragonfly Interconnect

- Das D, Avancha S, Mudigere D, Vaidynathan K, Sridharan S, Kalamkar D, Kaul B, Dubey P. Distributed deep learning using synchronous stochastic gradient descent. arXiv preprint arXiv:1602.06709. 2016 Feb 22.

2016/04: Google: Inception on 200 Nvidia K40's; ? Interconnect

- Chen J, Monga R, Bengio S, Jozefowicz R. Revisiting Distributed Synchronous SGD. arXiv preprint arXiv:1604.00981. 2016 Apr 4. Also, ICLR Workshop 2016.
- Dean, Jeff, Large-Scale Deep Learning With TensorFlow, presentation at ScaledML 2016, July 2016.

2016/11: Amazon: Resnet/Inception 3 on 128 K80's: 56GB Ethernet

- Mu Li, Alex Smola, MXNET,
- http://www.allthingsdistributed.com/2016/11/mxnet-default-framework-deep-learning-aws.html

2017/06: FaceBook: Resnet-50 on 256 P100s: NVLink, 1 hour

- https://arxiv.org/abs/1706.02677

2017/10-12: Yang You …

- You, Yang, Zhao Zhang, C. Hsieh, James Demmel, and Kurt Keutzer. "ImageNet training in minutes." Best Paper Award, ICPP 2018. Also, *arXiv preprint arXiv:* 1709.05011 (2017).

2017/11: Preferred Networks

- https://www.preferred-networks.jp/en/news/pr20171110

2018/07: Tencent

- Jia, Xianyan, et al. "Highly Scalable Deep Learning Training System with Mixed-Precision: Training ImageNet in Four Minutes." *arXiv preprint arXiv:1807.11205* (2018). Jia, Xianyan, et al. "Highly Scalable Deep Learning Training System with Mixed-Precision: Training ImageNet in Four Minutes." arXiv preprint arXiv:1807.11205 (2018).

2018/08: Fast AI

- Jeremy Howard. "Now anyone can train Imagenet in 18 minutes." http://www.fast.ai/2018/08/10/fastai-diu-imagenet.

CONCLUSION

Today

- Scaling training is a matter of
  - Achieving higher processor performance
  - Minimizing communication latency
- Current Accelerator Systems
  - GEMM centric
  - Often, the Deep Neural Network does not fit
  - Increase peak performance processors
  - Decrease memory traffic
  - Increase Utilization by increasing batch size
- The only way to scale Synchronous SGD is to increasing batch sizes
  - However retaining accuracy may require hyperparameter tuning
  - Very large batch sizes only demonstrated on CNNs
- Minimize the impact of communication latency
  - Use asynchronous approaches, but those have all negatively impacted accuracy
  - Hide communication latency by pipelining gradients

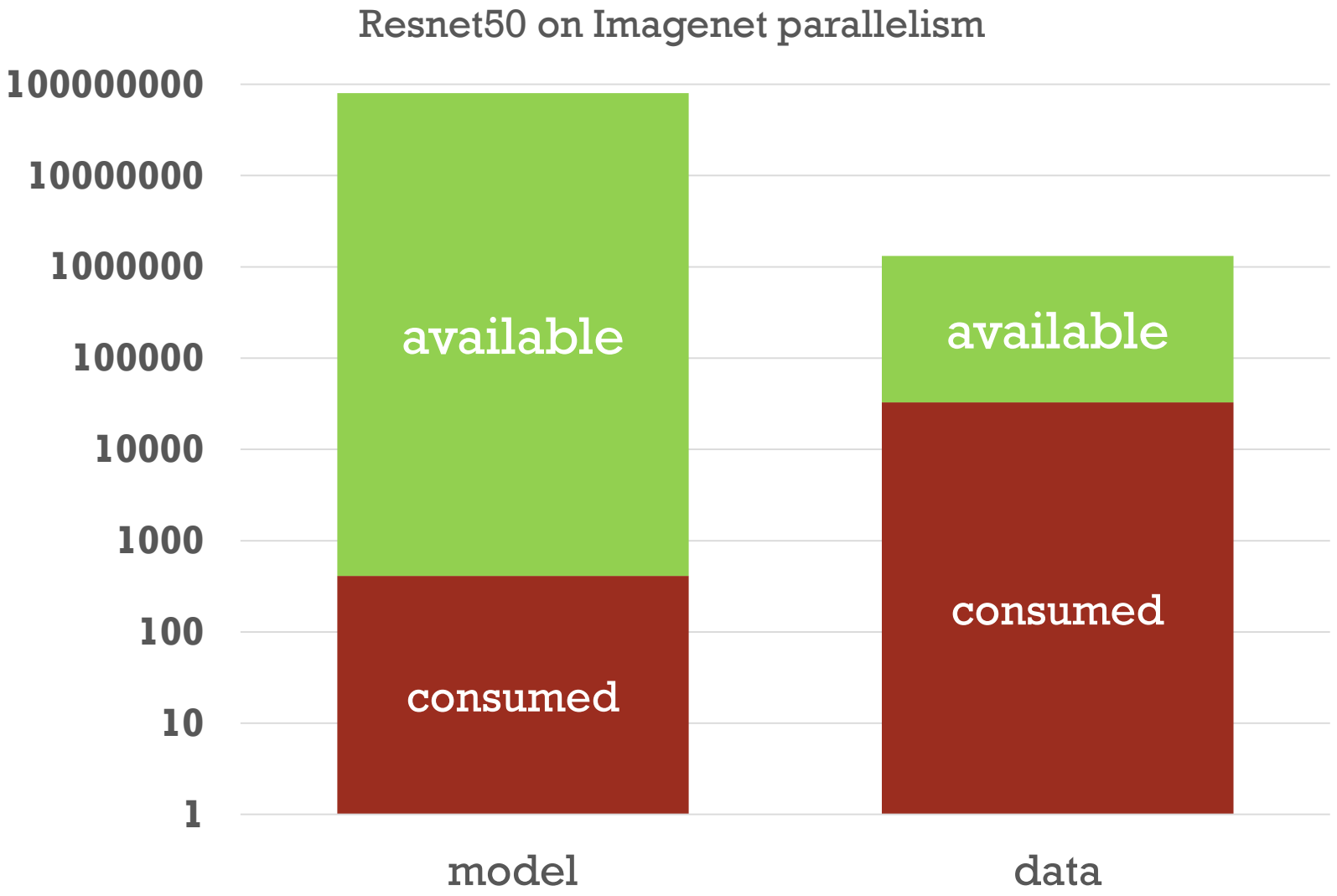
Tomorrow (Comments on the text below coming independently)

❑ A future machine can address:

- Large batch size requires hyper parameter tuning
  - Scale the performance but still keep batch size small
- Vast fine-grained-parallelism with huge compute
  - Scale fine grain parallelism
- Memory Wall
  - Large memory BW at large capacity
- Distributed Interconnect Network Wall
  - Take advantage of model parallelism
  - Exploit nearby communication at low latency
  - Sparse Communication
- Abundance of fine & coarse grain sparsity
  - Capture the sparsity available in a fine grain fashion

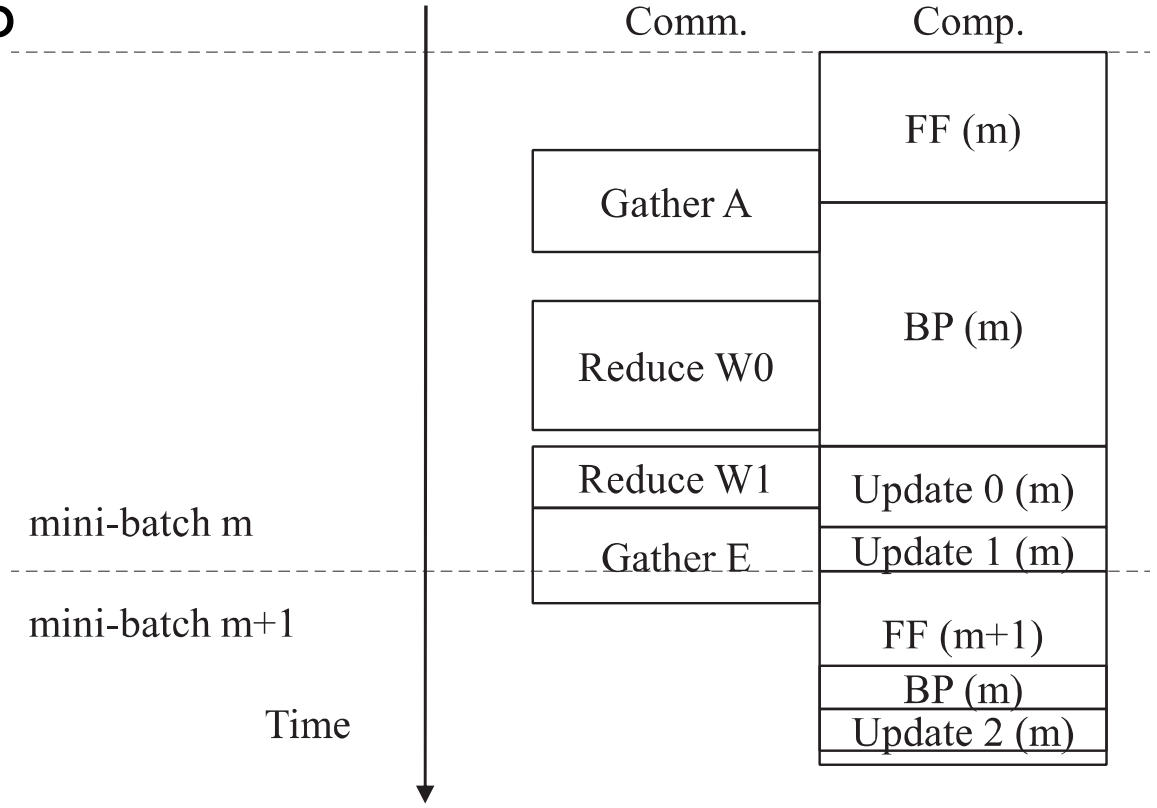
❑ Benchmarks emerging with some difficulty

TRAINING PARALLELISM OPPORTUNITIES



HIDING COMMUNICATION LATENCIES

- Time-flow chart with maximized overlap
- Linear speedup
  - All communications are hidden behind the computation
- Gradient computation and parameter update at the first fully-connected layers are delayed to the next mini-batch training



Sunwoo Lee, Dipendra Jha, Ankit Agrawal, Alok Choudhary, and Wei-keng Liao, "Parallel Deep Convolutional Neural Network Training by Exploiting the Overlapping of Computation and Communication ", IEEE 24th International Conference on High Performance Computing, 2017.