Architecture is not everything

Andrzej Sułecki, 2018-06-06

NVIDIA

How to decrease training time

1. Increase FLOPS/second
   a. Use more hardware
   b. Use better hardware
   c. Use hardware better
2. Compute less things
   a. Less epochs
   b. Smaller/shorter training examples

Increase FLOP/s

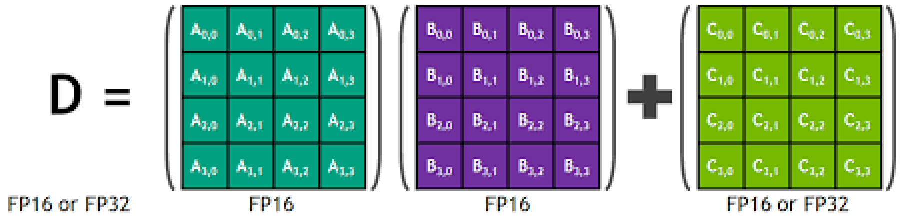
1. Use more hardware
    a. More GPUs = more speed
2. Use better hardware
    a. Better GPUs = more speed
3. Efficient hardware use
    a. FP16
    b. Bigger batch size

FP16

FP16

Overview

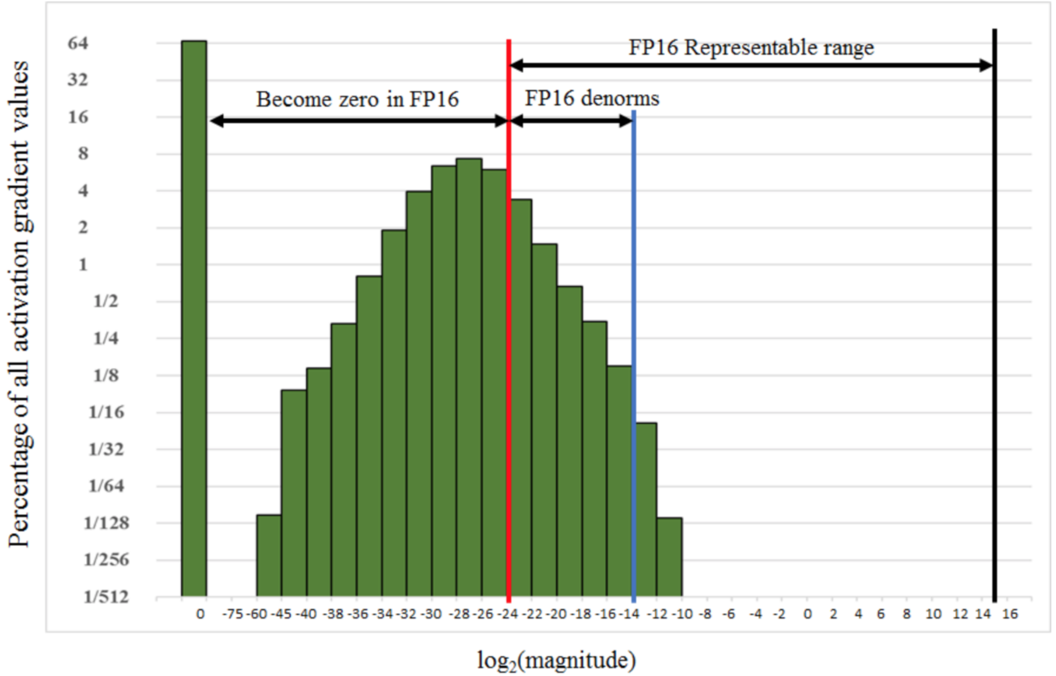
- Much faster
  - Theoretical speedup 8x
  - In practice can be 2-3x
- Lower memory usage = larger batch size
- Convergence problems
- Static loss scaling
- Dynamic loss scaling



FP16

Problems

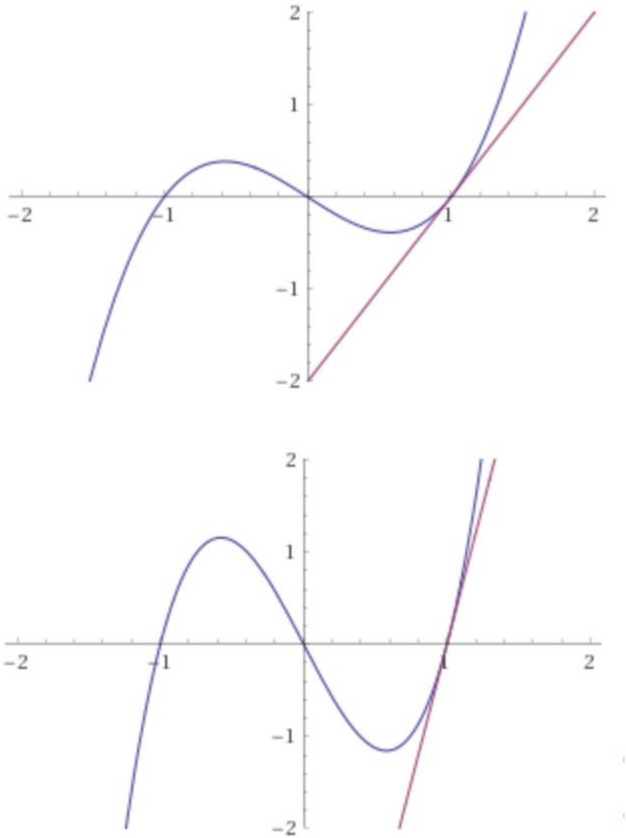
- Gradients too small to be represented in FP16
- Updates too small to move the weights



FP16

Static Loss Scaling

- Keep copy of the network weights in FP32
- Multiply the loss by scale_factor
- Gradients are scale_factor times larger
- This is works because *[a*f(x)]’* = *a*f’(x)*
- scale_factor from 1024 to 8192
- Works on ImageNet with no accuracy loss



FP16

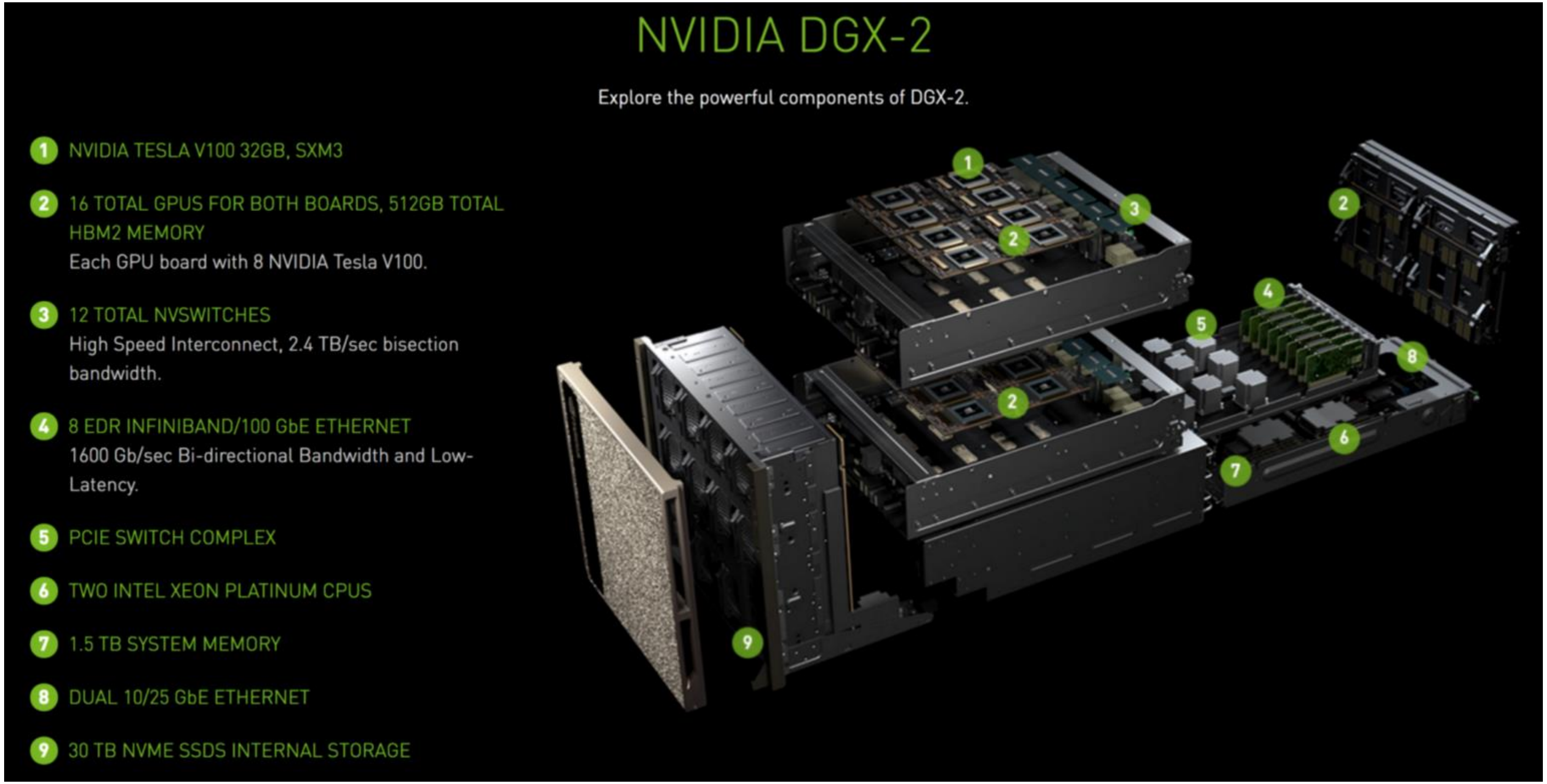
Dynamic loss scaling

- If NAN/Inf/NInf found in gradient:
  - Skip batch
  - decrease scale factor
- If gradient ok for a long time.
  - increase scale factor
- Not needed for ImageNet, useful for NLP/ASR/others

Big batch size

Big batch size

Motivation

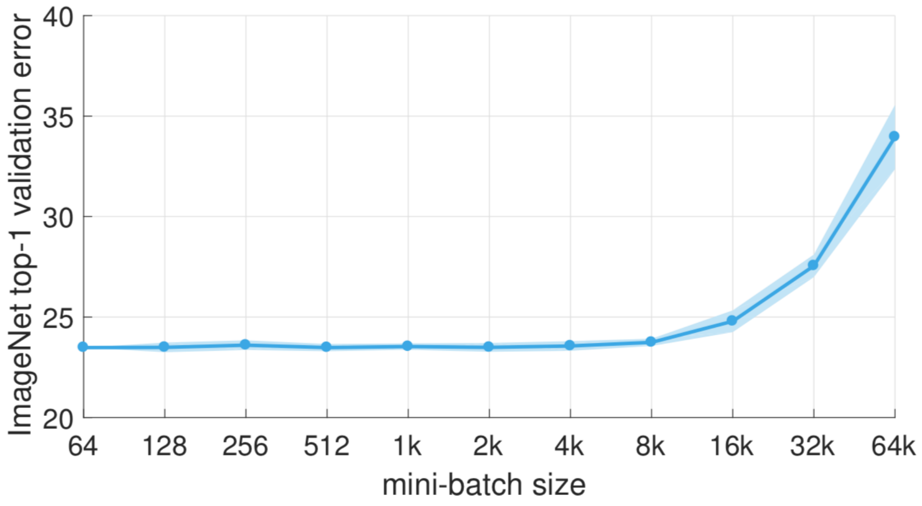


Accurate, Large Minibatch SGD: Training ImageNet in 1 hour

30 Apr 2018 - P. Goyal et al. - Facebook

https://arxiv.org/pdf/1706.02677.pdf

- Linear LR scaling is not enough for batches over 1k
- Gradual LR warmup
- < 1 hour
- 256 P100 GPUs
- BS = 8k



Layer-wise Adaptive Rate Scaling

Yang You (Berkley), Igor Gitman (Carnagie Mellon), Boris Ginsburg (NVIDIA)

https://arxiv.org/pdf/1708.03888.pdf

- AlexNet - batch size = 8k
- ResNet - batch size = 32k

Algorithm 1 SGD with LARS. Example with weight decay, momentum and polynomial LR decay.

Parameters: base LR γ_0, momentum m, weight decay β, LARS coefficient η, number of steps T
Init: $t = 0, v = 0$. Init weight w_0^l for each layer l
while $t < T$ for each layer l **do**
  $g_t^l \leftarrow \nabla L(w_t^l)$ (obtain a stochastic gradient for the current mini-batch)
  $\gamma_t \leftarrow \gamma_0 * \left(1 - \frac{t}{T}\right)^2$ (compute the global learning rate)
  $\boxed{\lambda^l \leftarrow \frac{||w_t^l||}{||g_t^l|| + \beta ||w_t^l||} \text{ (compute the local LR } \lambda^l\text{)}}$
  $v_{t+1}^l \leftarrow m v_t^l + \gamma_{t+1} * \lambda^l * (g_t^l + \beta w_t^l)$ (update the momentum)
  $w_{t+1}^l \leftarrow w_t^l - v_{t+1}^l$ (update the weights)
end while

Part 1 - outtakes

What to do with the info you just heard

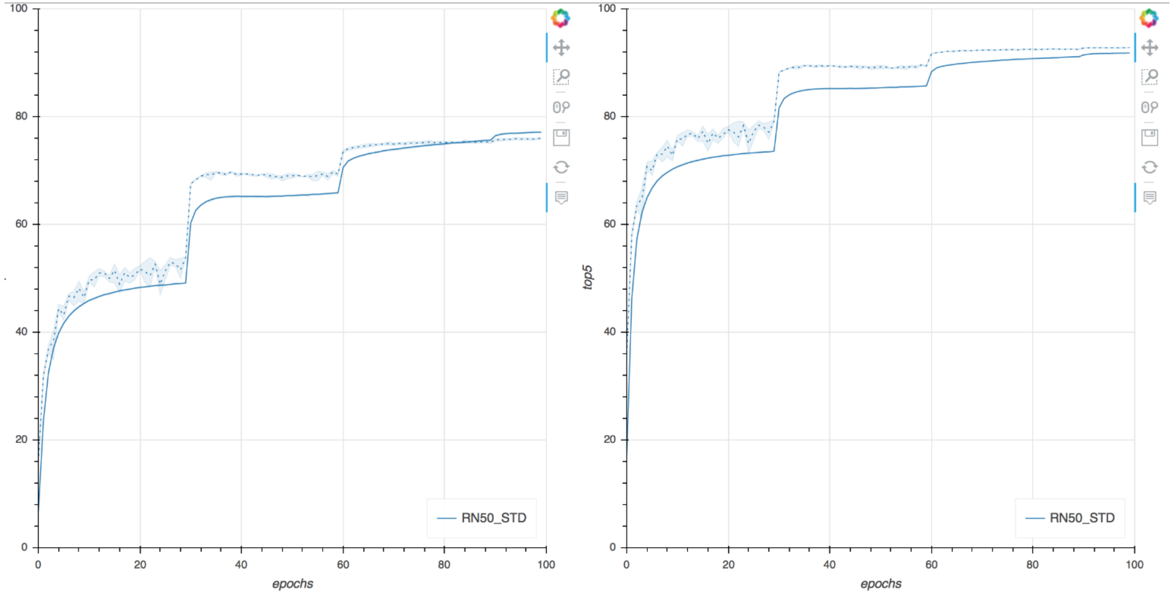
- Use FP16
  - Don’t be lazy, couple of lines of code can give a big speedup
  - If it doesn’t work, contact NVIDIA!
- Use large batches
  - Don’t be lazy, couple of lines of code can give a big speed-up

Compute less stuff

The “golden” standard

How does training look most of the time.

- 90 epochs
- Batch size = 256
- LR = 0.1
- Every 30 epochs decrease LR 10x



Why is it “bad”

Why bother changing the method if it works?

- Takes a lot of time
- Hurts this guys (more time = higher energy consumption):

- Higher cost of research/development.

DAWNBench

An End-to-End Deep Learning Benchmark and Competition

- Any architecture
- Must reach 93% top-5 accuracy.
- Best time-to-solution wins.

ImageNet Training

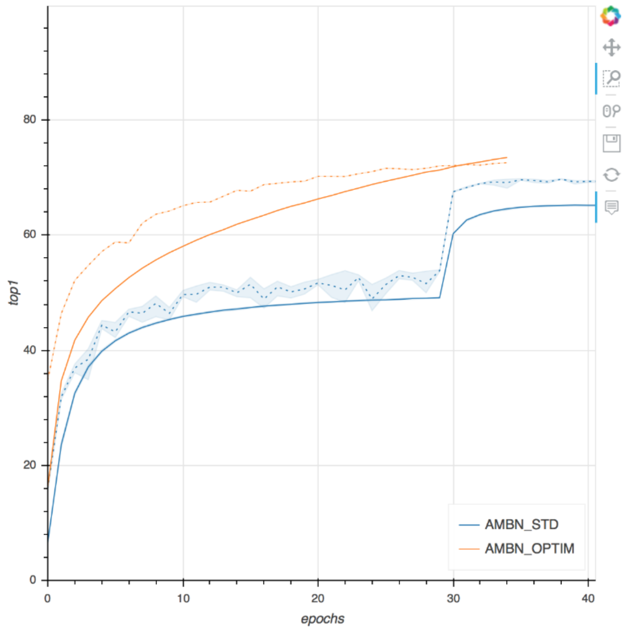
Submission Date	Model	Time to 93% Accuracy	Cost (USD)	Max Accuracy	Hardware	Framework
Apr 2018	ResNet50 *Google* source	0:30:43	N/A	93.03%	Half of a TPUv2 Pod	TensorFlow 1.8.0-rc1
Apr 2018	AmoebaNet-D N6F256 *Google* source	1:06:32	N/A	93.03%	1/4 of a TPUv2 Pod	TensorFlow 1.8.0-rc1
Apr 2018	AmoebaNet-D N6F256 *Google* source	1:58:24	N/A	93.17%	1/16 of a TPUv2 Pod	TensorFlow 1.8.0-rc1
Apr 2018	Resnet 50 *fast.ai + students team: Jeremy Howard, Andrew Shaw, Brett Koonce, Sylvain Gugger* source	2:57:28	$72.40	93.05%	8 * V100 (AWS p3.16xlarge)	fastai / pytorch
Apr 2018	ResNet50 *Intel(R) Corporation* source	3:25:55	N/A	93.02%	128 nodes with Xeon Platinum 8124M / 144 GB / 36 Cores (Amazon EC2 [c5.18xlarge])	Intel(R) Optimized Caffe
Apr 2018	ResNet56 *Intel(R) Corporation* source	3:31:47	N/A	93.11%	128 nodes with Xeon Platinum 8124M / 144 GB / 36 Cores (Amazon EC2 [c5.18xlarge])	Intel(R) Optimized Caffe
Apr 2018	ResNet50 *Intel(R) Corporation* source	6:09:50	N/A	93.05%	64 nodes with Xeon Platinum 8124M / 144 GB / 36 Cores (Amazon EC2 [c5.18xlarge])	Intel(R) Optimized Caffe
Apr 2018	AmoebaNet-D N6F256 *Google Cloud TPU* source	7:28:30	$49.30	93.11%	GCP n1-standard-2, Cloud TPU	TensorFlow 1.8.0-rc0

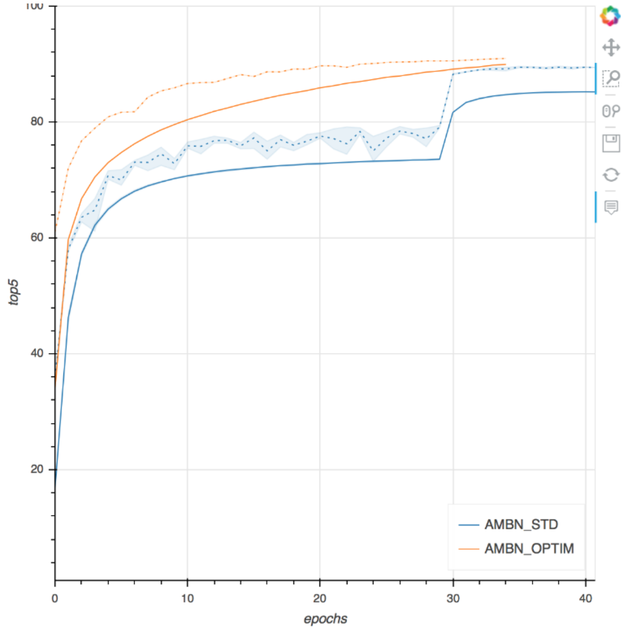
https://dawn.cs.stanford.edu/benchmark/

Google's entry

7.5h to 93% on ImageNET

- AmoebaNet
- 35 epochs
- RMSProp
- Exponential LR decay
- 1k batch size
- Exponential Moving Average of weights

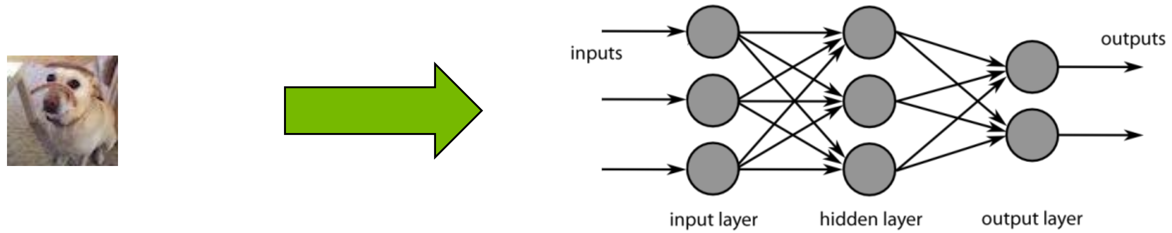


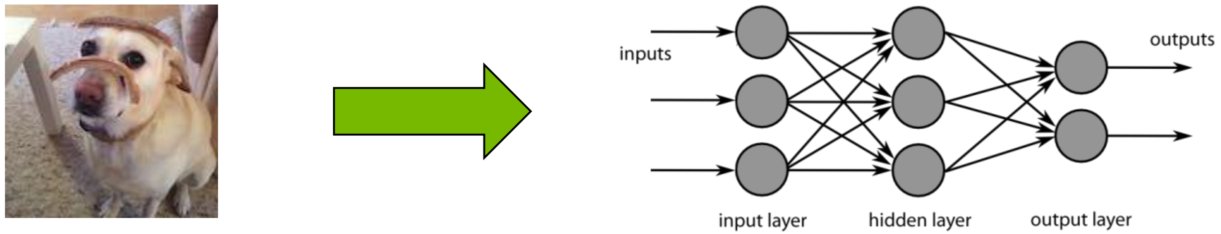


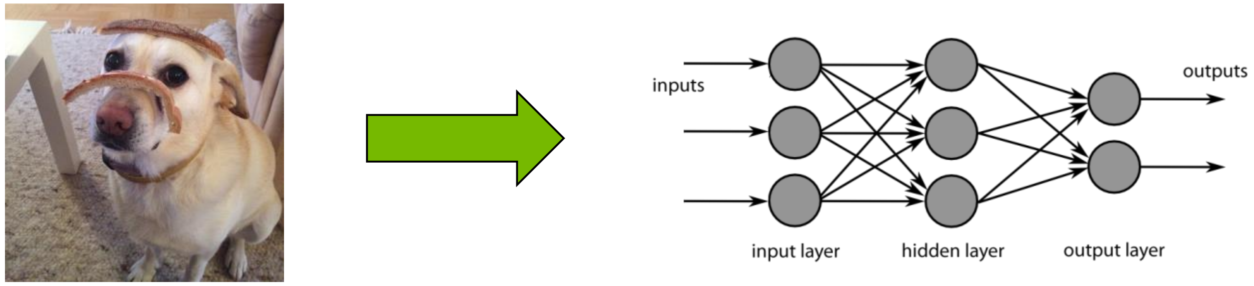
FastAI entry

< 3h to 93% on ImageNET

- 45 epochs
- 1024 BS
- Progressive resizing

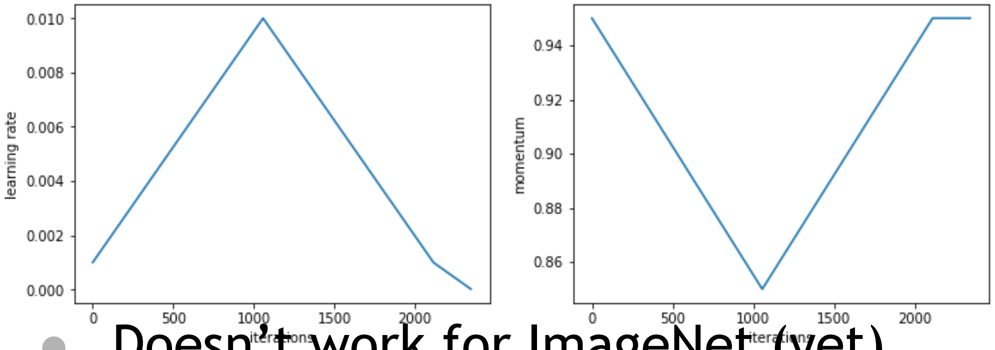






Super convergence

- Very large learning rate (up to 10x higher than usual)
- Cyclical Learning Rate
- Cyclical Momentum



- Doesn't work for ImageNet (yet)

What’s next?

NVIDIA Software Engineer

Intern / Junior / Regular / Senior

warsawcareers@nvidia.com

Deep Learning Algos	Data Processing	Deep Learning Libs	Math Libs
Will do: > R&D on DL algorithms > Deliver fast training recipes > Optimize training for GPU	**Will do:** > Contribute to Open Source > Develop data processing pipeline > Work with data augmentation and compression	**Will do:** > Develop C/C++ low level code > Analyze low level architecture of CPU & GPU > Optimize performance and memory usage	**Will do:** > Develop & optimize GPU math libraries
Skills needed: + C / C++ / Python + DL know-how + DL Frameworks (Tensorflow, PyTorch, MxNet)	**Skills needed:** + Modern C++ + System Design + Parallel Computing	**Skills needed:** + CUDA and/or parallel processing + Understanding building blocks of Deep Learning	**Skills needed:** + C / C++ + Math background (esp. Numerical Methods)
Nice to have skills: * CUDA / GPU Computing * Statistics	**Nice to have skills:** * CUDA or OpenCL * Image & Signal Processing * DL Frameworks	**Nice to have:** + Performance analysis of GPU code	**Nice to have skills:** * CUDA * Floating point arithmetic * Low level optimization

Location: Q22, Warsaw, Poland warsawcareers@nvidia.com

NVIDIA