AI-Systems

Efficient ML

[294-162]

Amir Gholami & Joseph E. Gonzalez

Acknowledgments

Many slides from Jonathan Frankle, Prof. Kurt Keutzer, Pallas Group

Agenda for Today

- 1:10-2:00: Efficient ML and Ideas in ML Frameworks
- 2:00-2:45: PC Meeting Discussions
- 2:45-3:00: Break
- 3:00-4:00: Guest Lecture by Tianqi Chen

Objectives For Today

- Get a good understanding of Pruning:
  - Methods for pruning applied to Inference
  - Pruning for efficient Training (Lottery Hypothesis Ticket)
- Learn About Quantization
  - Integer-only Quantization
  - Mixed Precision Quantization
- Important Ideas in ML Frameworks

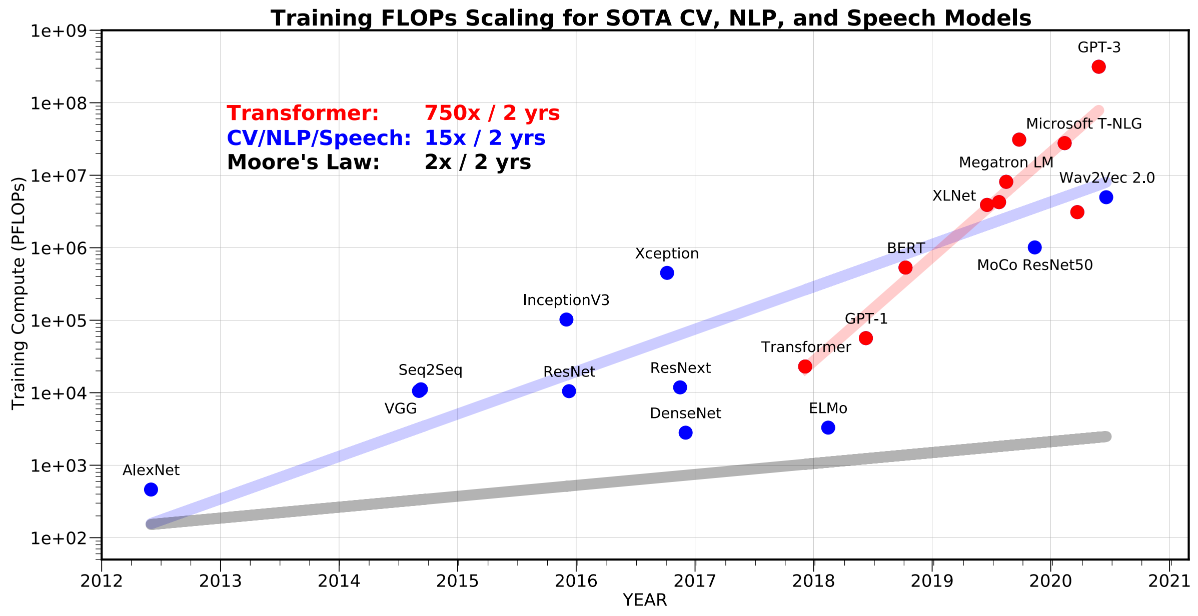
Efficient ML with Pruning

Most of the slides in this section are courtesy of Jonathan Frankle [Harvard/MosaicML]

Neural Network Training is Costly

Research Questions:

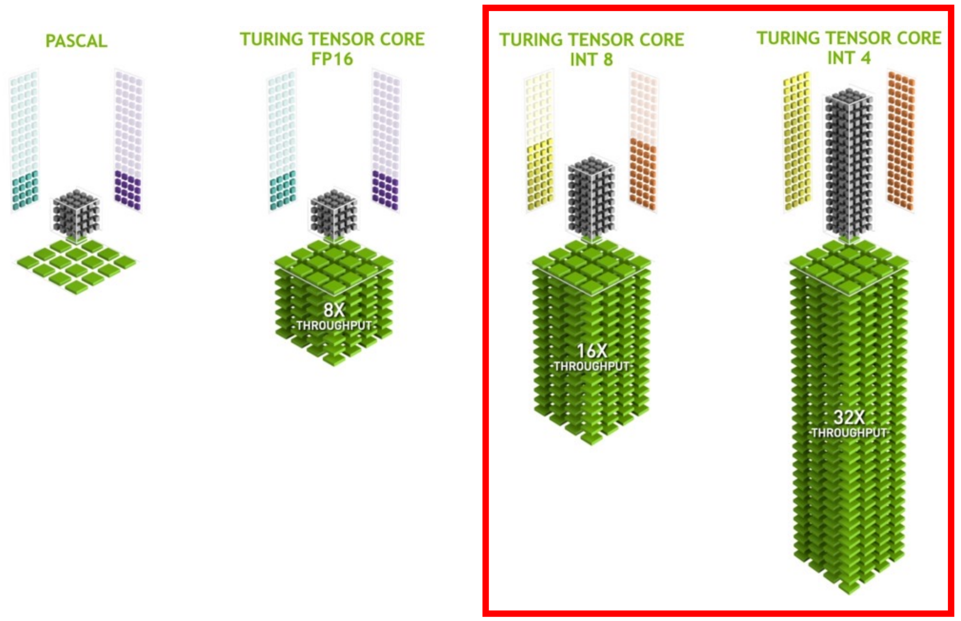
- Fundamentally do we need such large models to learn the representations that we are looking for?
- Can we reduce this cost without sacrificing accuracy?



Amir Gholami, Zhewei Yao, Sehoon Kim, Michael W. Mahoney, Kurt Keutzer, AI and Memory Wall, Riselab Medium Blogpost, 2021.

Reduced Precision Training

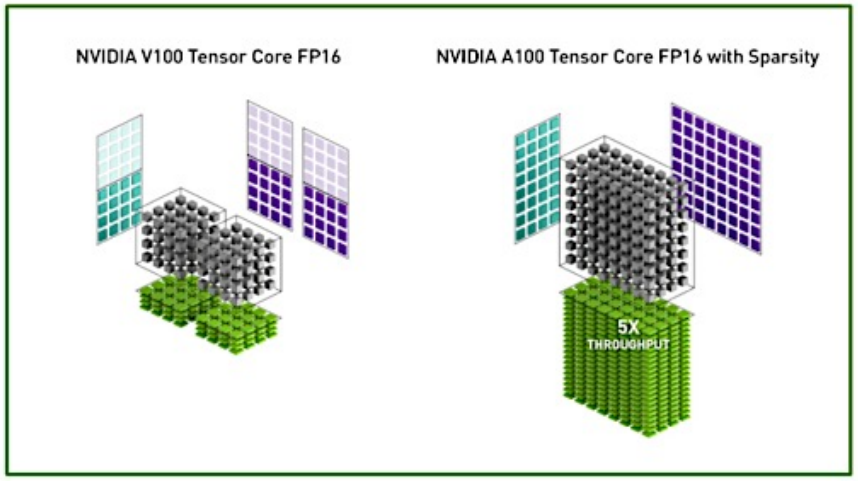
- ➢ As we saw in lec 2, we have been very successful in FP16 training which enabled 10x improvements in throughput
  - ➢ But it has been very difficult to go to reduce the precision further

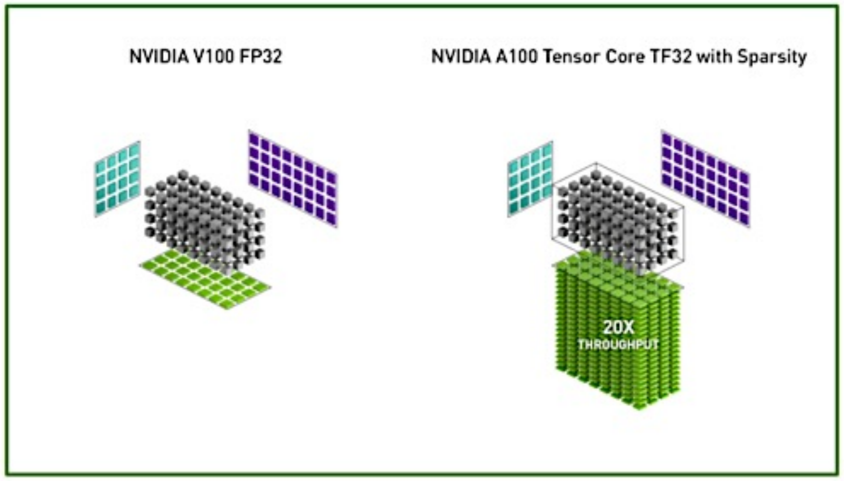


INT8/4 training often leads to significant accuracy degradation and is still an open research problem

How about Utilizing Sparsity?

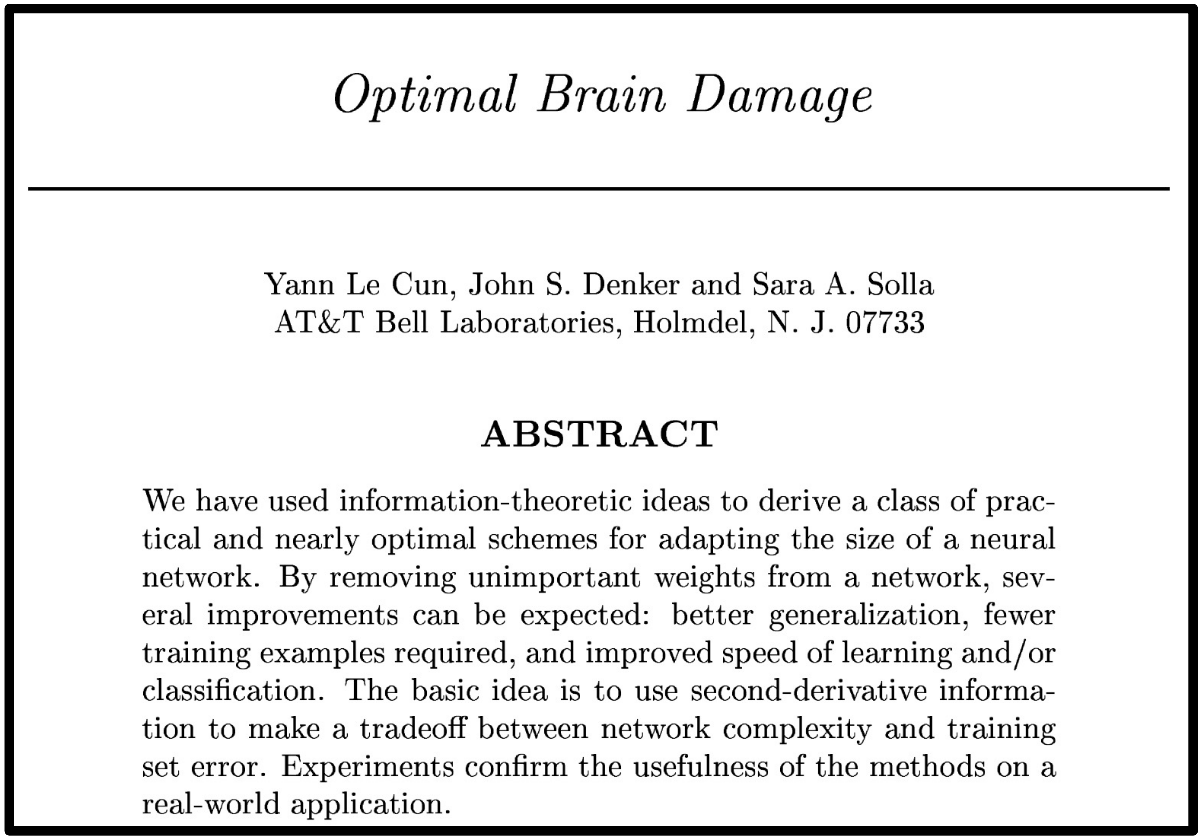
- New HW generations such as A100 have built-in support for accelerating sparse workloads
  - HW vendors can achieve significantly higher gains if sparse training becomes feasible for ML like FP16


NVIDIA V100 Tensor Core FP16
NVIDIA A100 Tensor Core FP16 with Sparsity
5X
THROUGHPUT



Sparsity: One of the Challenging Problems in ML

1990



Optimal Brain Damage

Yann Le Cun, John S. Denker and Sara A. Solla
AT&T Bell Laboratories, Holmdel, N. J. 07733

ABSTRACT

We have used information-theoretic ideas to derive a class of practical and nearly optimal schemes for adapting the size of a neural network. By removing unimportant weights from a network, several improvements can be expected: better generalization, fewer training examples required, and improved speed of learning and/or classification. The basic idea is to use second-derivative information to make a tradeoff between network complexity and training set error. Experiments confirm the usefulness of the methods on a real-world application.

Background: Neural Network Pruning

prune /pru:n/ verb

To reduce the extent of [a neural network] by removing **superfluous** or unwanted parts.

(Oxford English Dictionary)

Goal: reduce the cost of inference and/or Training

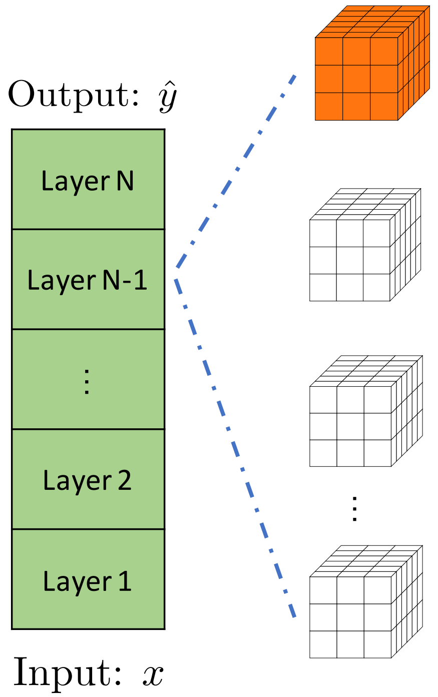
Background: Neural Network Pruning

Pruning similar to the **Jenga** game. We only remove blocks that are not sensitive.

Popular metrics for measuring sensitivity:

- Magnitude
- Gradients
- Hessian

Image from UniversityCoop



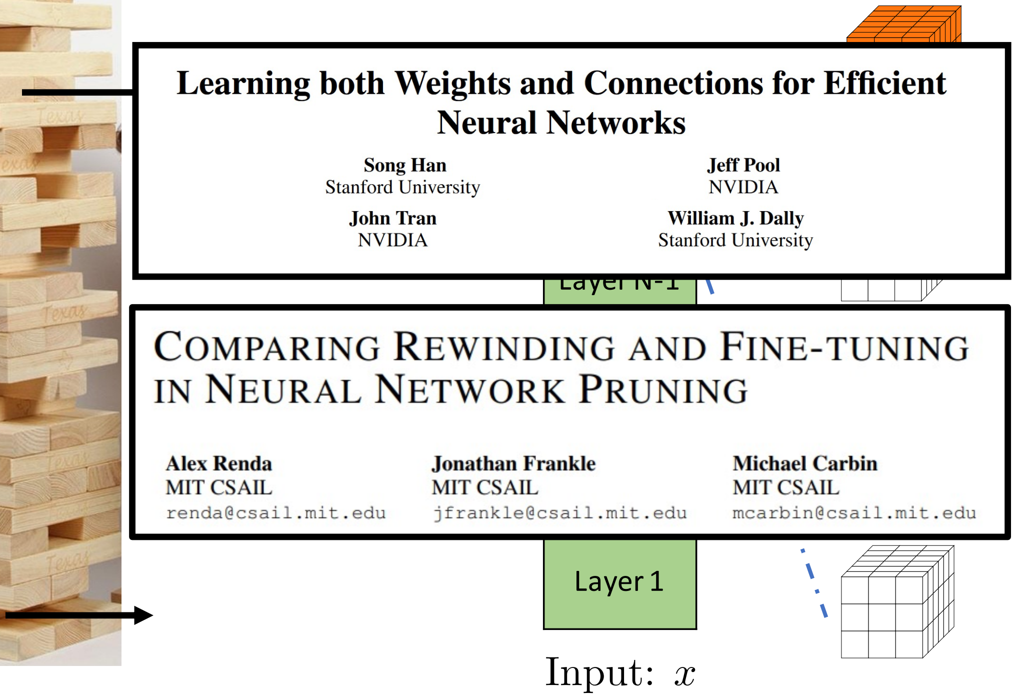
Yu S*, Yao Z*, Gholami* A, Dong Z*, Mahoney MW, Keutzer K. Hessian-Aware Pruning and Optimal Neural Implant. WACV, 2022.
LeCun Y, Denker J, Solla S. Optimal brain damage. Advances in neural information processing systems. 1989.

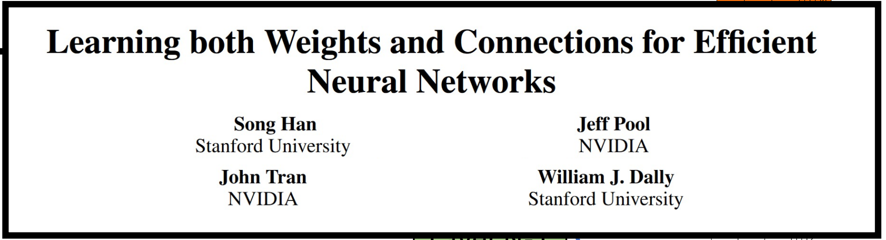
Background: Neural Network Pruning

Pruning similar to the **Jenga** game. We only remove blocks that are not sensitive.

There are many metrics for measuring sensitivity:

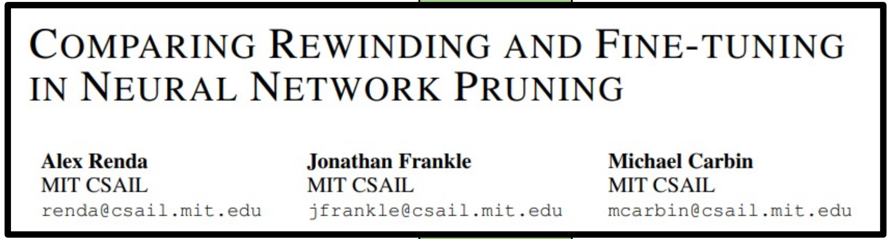
- Magnitude
- Gradients
- Hessian





Learning both Weights and Connections for Efficient Neural Networks

Song Han Stanford University
Jeff Pool NVIDIA
John Tran NVIDIA
William J. Dally Stanford University



COMPARING REWINDING AND FINE-TUNING IN NEURAL NETWORK PRUNING

Alex Renda MIT CSAIL renda@csail.mit.edu
Jonathan Frankle MIT CSAIL jfrankle@csail.mit.edu
Michael Carbin MIT CSAIL mcarbin@csail.mit.edu

Image from UniversityCoop

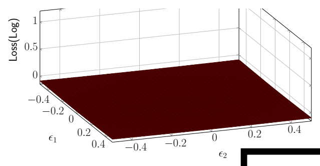
Yu S*, Yao Z*, Gholami* A, Dong Z*, Mahoney MW, Keutzer K. Hessian-Aware Pruning and Optimal Neural Implant. WACV, 2022.
LeCun Y, Denker J, Solla S. Optimal brain damage. Advances in neural information processing systems. 1989.

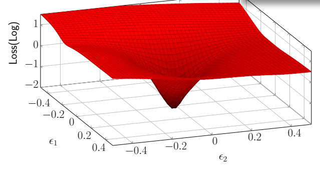
Background: Neural Network Pruning

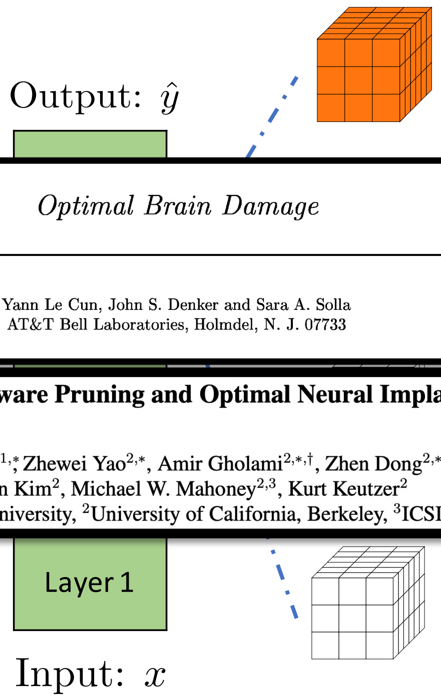
Pruning similar to the **Jenga** game. We only remove blocks that are not sensitive.

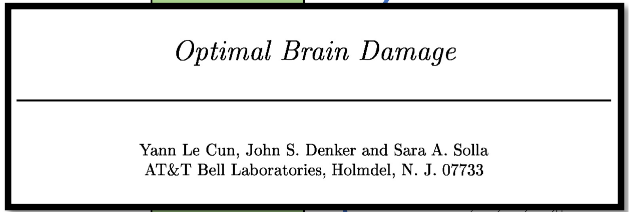
There are many metrics for measuring sensitivity:

- Magnitude
- Gradients
- Hessian



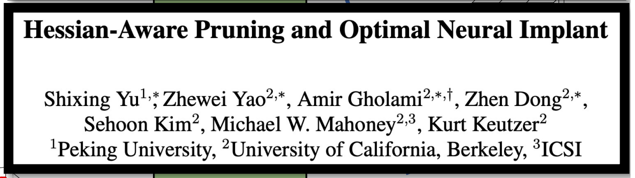






Optimal Brain Damage

Yann Le Cun, John S. Denker and Sara A. Solla
AT&T Bell Laboratories, Holmdel, N. J. 07733



Hessian-Aware Pruning and Optimal Neural Implant

Shixing Yu[1,*], Zhewei Yao[2,*], Amir Gholami[2,*,†], Zhen Dong[2,*], Sehoon Kim[2], Michael W. Mahoney[2,3], Kurt Keutzer[2]
[1]Peking University, [2]University of California, Berkeley, [3]ICSI

Image from UniversityCoop

LeCun Y, Denker J, Solla S. Optimal brain damage. Advances in neural information processing systems. 1989.
Yu S*, Yao Z*, Gholami* A, Dong Z*, Mahoney MW, Keutzer K. Hessian-Aware Pruning and Optimal Neural Implant. WACV, 2022.

Hessian Aware Pruning

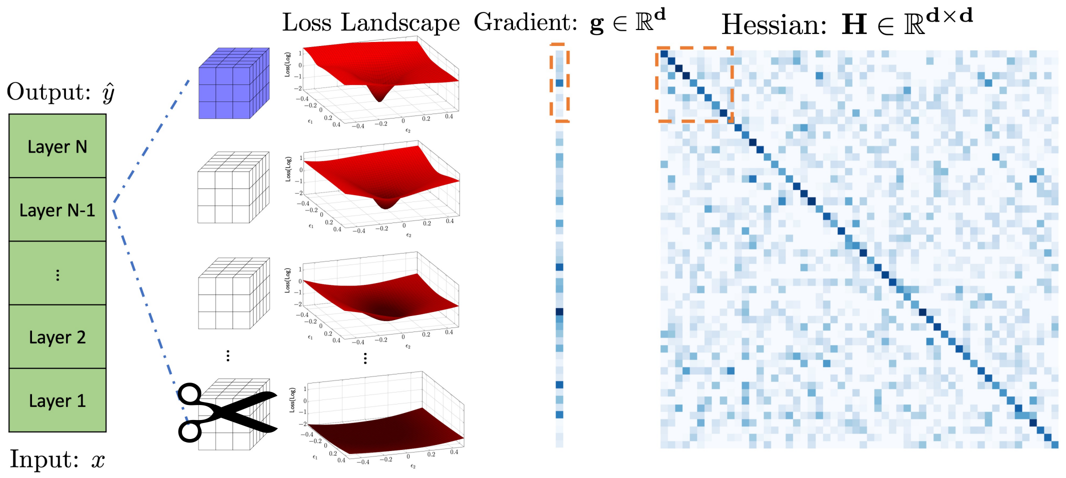
➢ Why Second-order instead of magnitude pruning?

$$L(w) = \frac{1}{N}\sum_{i=1}^{N} l(x_i, y_i, w)$$

$$\Delta L = L(w + \Delta w) - L(w)$$

$$\Delta L \approx g^T \Delta w + \frac{1}{2}\Delta w^T H \Delta w$$

$$\Delta L = \frac{Trace(H_{p,p})}{2p} \|w_p\|_2^2$$

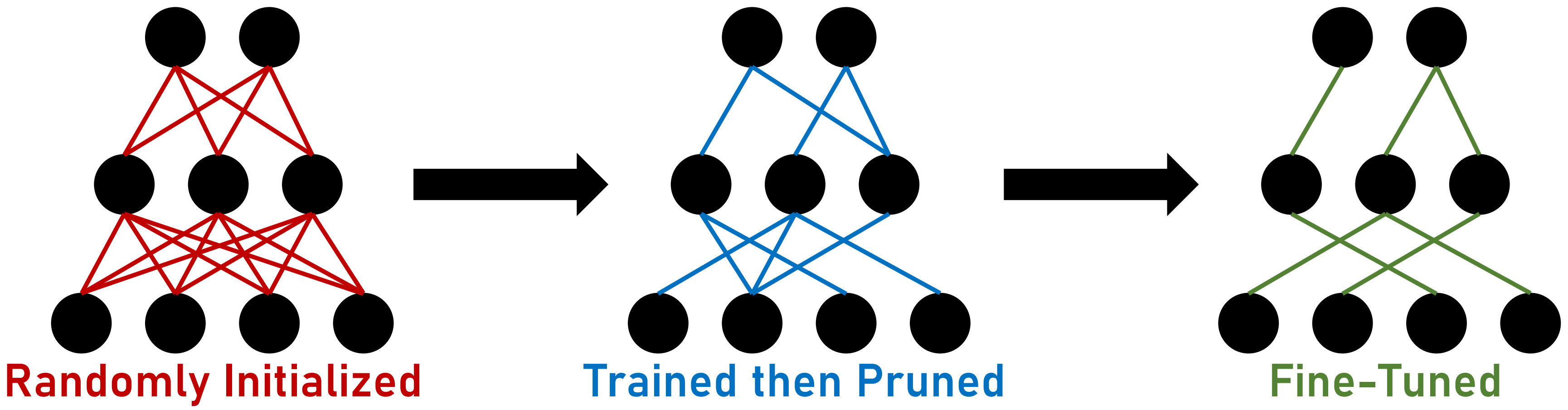


Loss Landscape
Gradient: $\mathbf{g} \in \mathbb{R}^{\mathbf{d}}$
Hessian: $\mathbf{H} \in \mathbb{R}^{\mathbf{d}\times\mathbf{d}}$
Output: $\hat{y}$
Layer N
Layer N-1
⋮
Layer 2
Layer 1
Input: x

LeCun Y, Denker J, Solla S. Optimal brain damage. Advances in neural information processing systems. 1989.
Yu S*, Yao Z*, Gholami* A, Dong Z*, Mahoney MW, Keutzer K. Hessian-Aware Pruning and Optimal Neural Implant. WACV, 2022.
Yao Z*, Gholami A*, Keutzer K, Mahoney M. PyHessian: Neural networks through the lens of the Hessian. AAAI, 2020.

Background: Neural Network Pruning

1) Train the network
2) Remove superfluous structure
3) Fine-tune the network
4) Optionally: prune and fine-tune iteratively



Pruning Can be Effective

- If we can indeed find pruned models that achieve high accuracy that means that there exists sub-networks that have enough representational power.
  - **Then why not sparse models to begin with?**

Different structured pruning methods

Method	Acc.(%)	Param.(%)	FLOPs(%)
VGG16	93.96	100.0	100.0
L1[36]	93.40	36.0	65.7
SSS[25]	93.02	26.2	58.4
VarP[73]	93.18	26.7	60.9
HRank[37]	93.43	17.1	46.5
GAL-0.05[39]	92.03	22.4	60.4
HRank[37]	92.34	17.9	34.7
GAL-0.1[39]	90.73	17.8	54.8
HAP	**93.66**	**10.1**	**29.7**

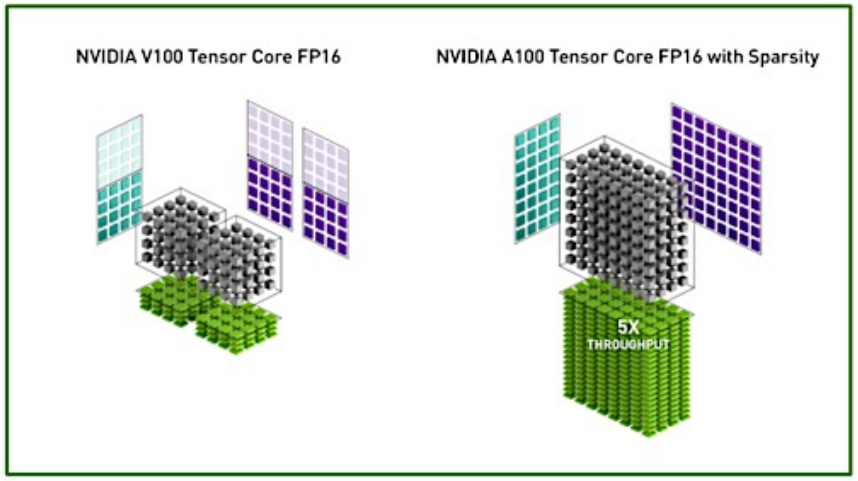
Yu S*, Yao Z*, Gholami A*, Dong Z*, Kim S, Mahoney MW, Keutzer K. Hessian-Aware Pruning and Optimal Neural Implant. WACV, 2022.

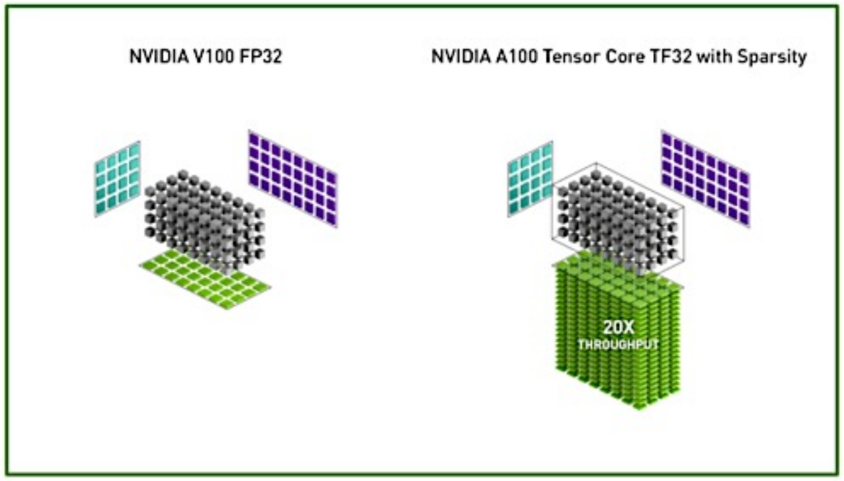
Research Question

Research Question: If we can prune models after training, can we train smaller models?

Research Question: Training Sparse Models?

If we could train small models that are sparse from the beginning, then we can significantly reduce the cost of training


NVIDIA V100 Tensor Core FP16
NVIDIA A100 Tensor Core FP16 with Sparsity
5X
THROUGHPUT



Research Question

If we can prune models after training, can we train smaller models?

Capacity for representation vs. optimization

Research Question

If we can prune models after training, can we train smaller models?

Capacity for representation vs. optimization

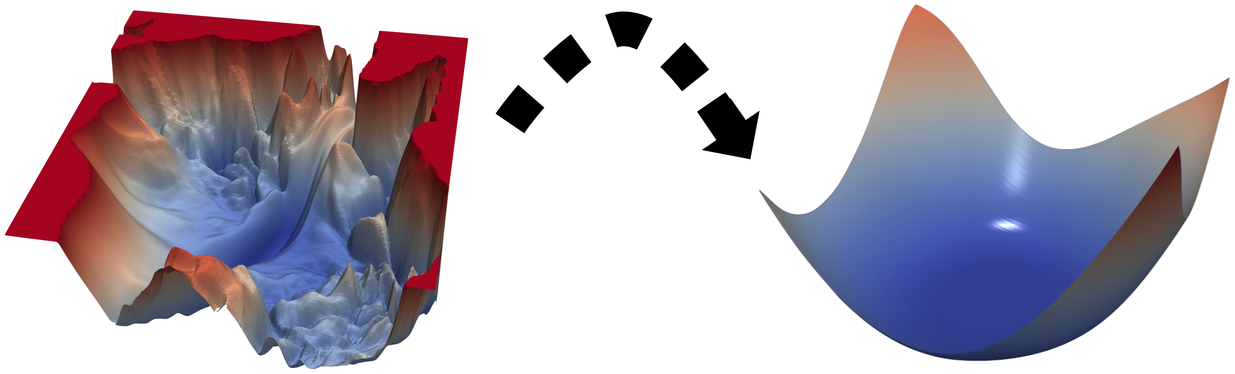
Research Question

If we can prune models after training, can we train smaller models?

Capacity for representation vs. optimization ??

Overparameterized Models

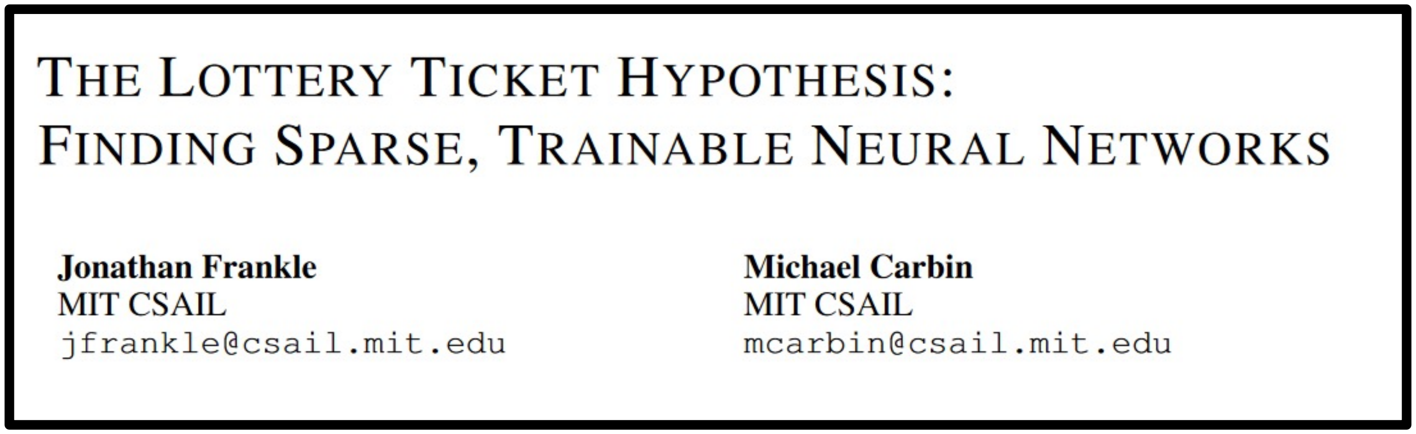
- Current Hypothesis: Empirical results have consistently been showing that large models are easier to train and generalize better
  - It appears as if adding more parameters makes the optimization problem easier
  - If this is true, then is training small models infeasible?



Adding more parameters
(Disclaimer: This hypothesis is highly debated and is not always true!)

Is Training Small-Scale NNs Possible?

ICLR 2019

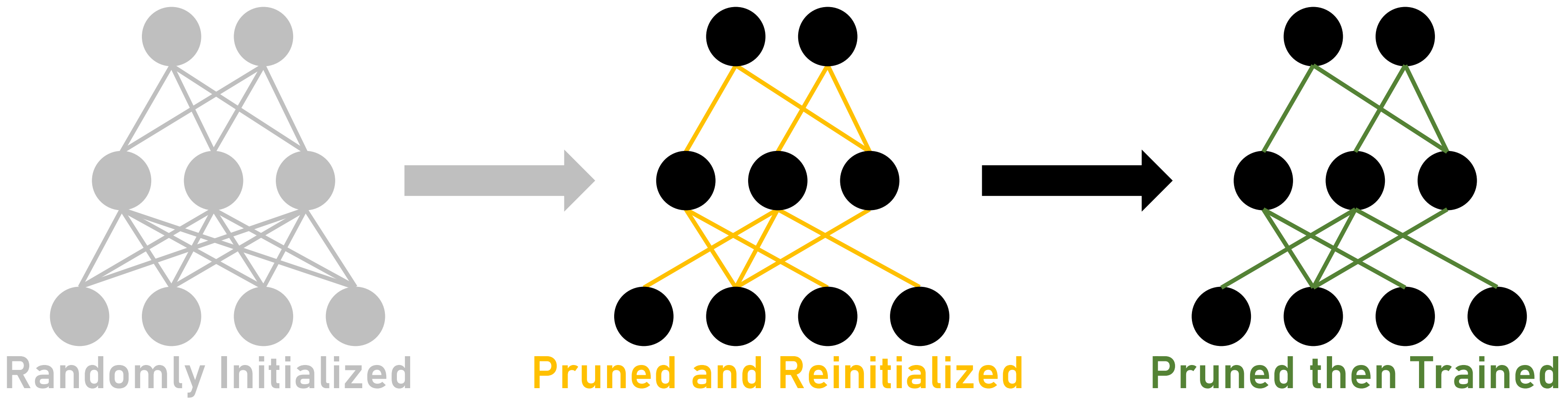

THE LOTTERY TICKET HYPOTHESIS:
FINDING SPARSE, TRAINABLE NEURAL NETWORKS

Jonathan Frankle
MIT CSAIL
jfrankle@csail.mit.edu

Michael Carbin
MIT CSAIL
mcarbin@csail.mit.edu

Attempting to Train a Pruned Network

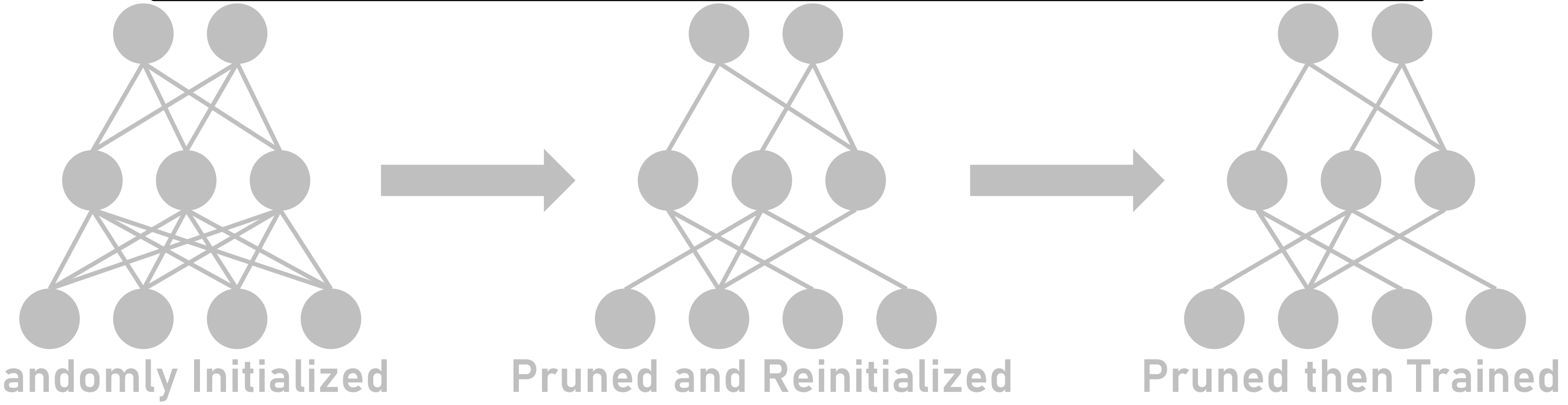
1) Train the network
2) Remove superfluous structure
3) Randomly initialize the pruned network
4) Train it to convergence



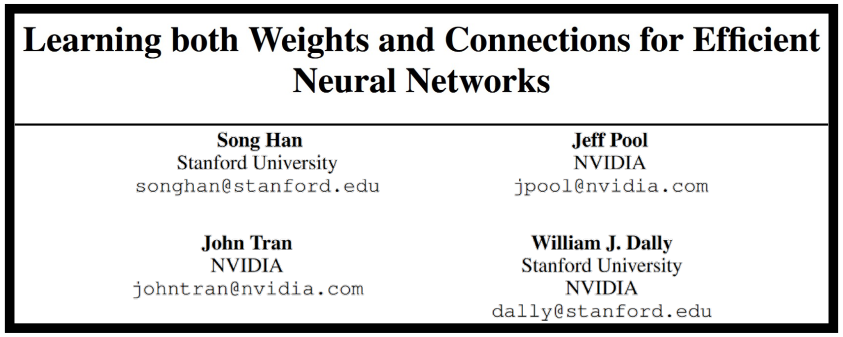
Attempting to Train a Pruned Network

1) Train the network
2) Remove superfluous structure
3) Randomly initialize the pruned network
4) T

Does not reach full accuracy



Attempting to Train a Pruned Network


Learning both Weights and Connections for Efficient Neural Networks

Song Han
Stanford University
songhan@stanford.edu

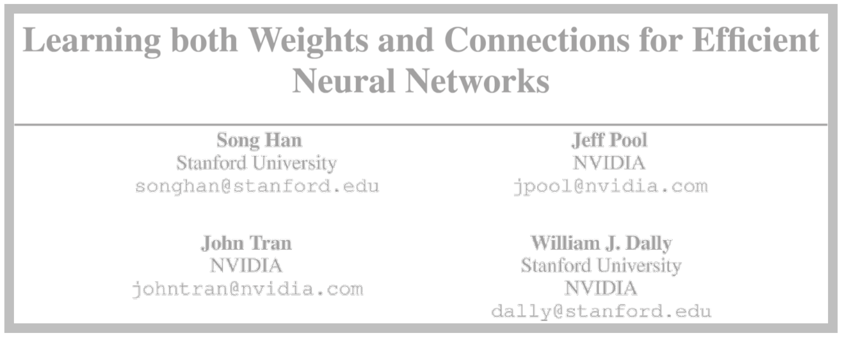
Jeff Pool
NVIDIA
jpool@nvidia.com

John Tran
NVIDIA
johntran@nvidia.com

William J. Dally
Stanford University
NVIDIA
dally@stanford.edu

It is better to retain the weights from the initial training phase for the connections that survived pruning than it is to reinitialize.

Attempting to Train a Pruned Network

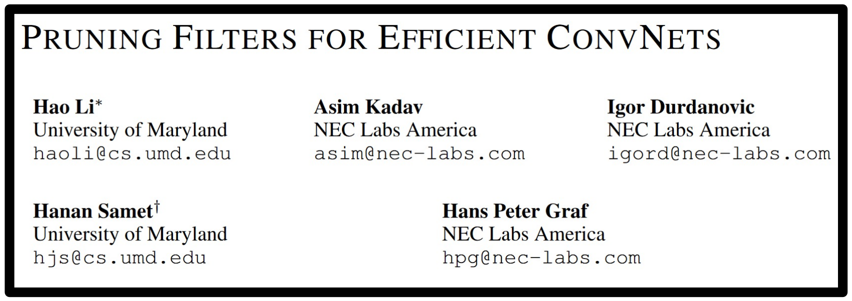

Learning both Weights and Connections for Efficient Neural Networks

Song Han
Stanford University
songhan@stanford.edu

Jeff Pool
NVIDIA
jpool@nvidia.com

John Tran
NVIDIA
johntran@nvidia.com

William J. Dally
Stanford University
NVIDIA
dally@stanford.edu


PRUNING FILTERS FOR EFFICIENT CONVNETS

Hao Li*
University of Maryland
haoli@cs.umd.edu

Asim Kadav
NEC Labs America
asim@nec-labs.com

Igor Durdanovic
NEC Labs America
igord@nec-labs.com

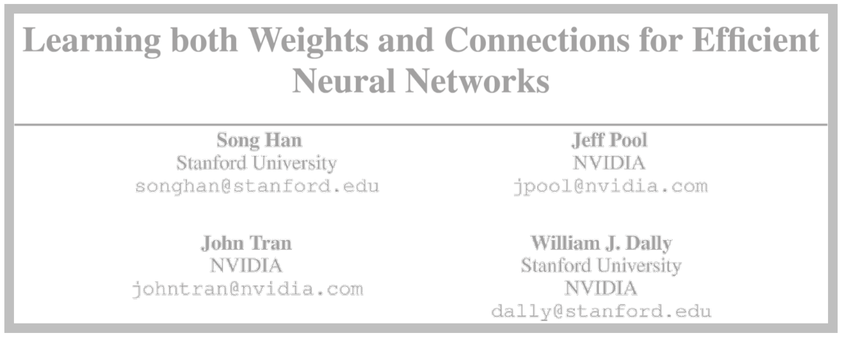
Hanan Samet†
University of Maryland
hjs@cs.umd.edu

Hans Peter Graf
NEC Labs America
hpg@nec-labs.com

It is better to retain the weights from the initial training phase for the connections that survived pruning than it is to reinitialize.

Training a pruned model from scratch performs worse than retraining a pruned model, which may indicate the difficulty of training a network with small capacity.

Attempting to Train a Pruned Network

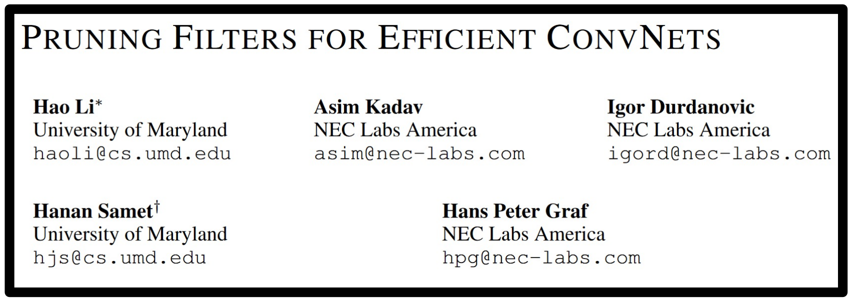

Learning both Weights and Connections for Efficient Neural Networks

Song Han
Stanford University
songhan@stanford.edu

Jeff Pool
NVIDIA
jpool@nvidia.com

John Tran
NVIDIA
johntran@nvidia.com

William J. Dally
Stanford University
NVIDIA
dally@stanford.edu


PRUNING FILTERS FOR EFFICIENT CONVNETS

Hao Li[*]
University of Maryland
haoli@cs.umd.edu

Asim Kadav
NEC Labs America
asim@nec-labs.com

Igor Durdanovic
NEC Labs America
igord@nec-labs.com

Hanan Samet[†]
University of Maryland
hjs@cs.umd.edu

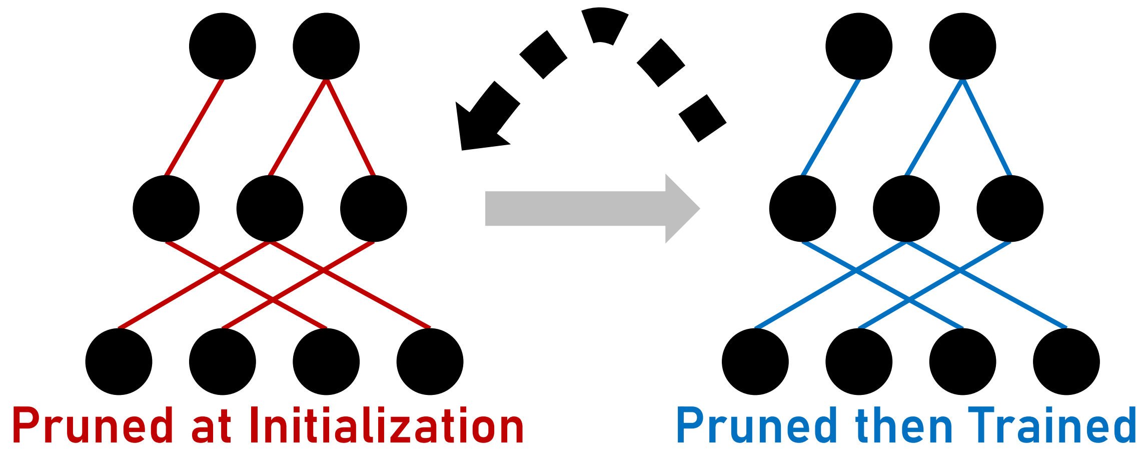
Hans Peter Graf
NEC Labs America
hpg@nec-labs.com

It is better to retain the weights from the initial training phase for the connections that survived pruning than it is to reinitialize.

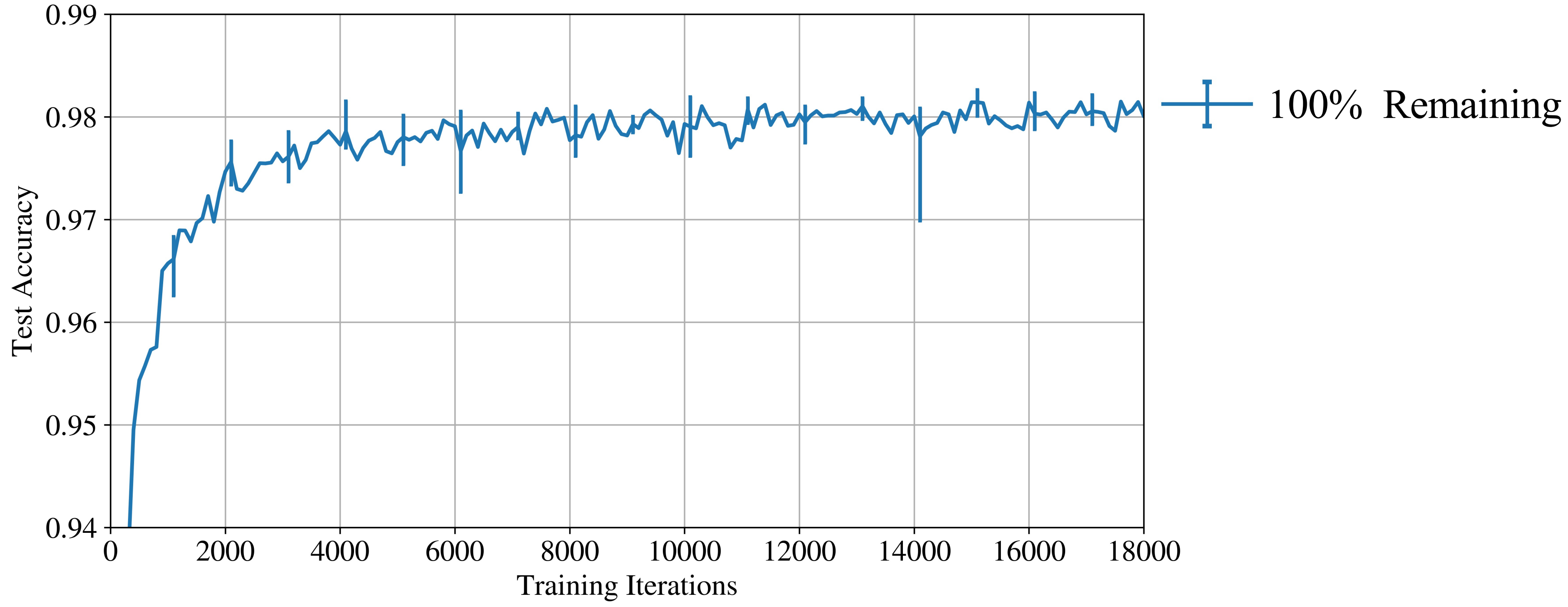
Training a pruned model from scratch performs worse than retraining a pruned model, which may indicate the difficulty of training a network with small capacity.

Iterative Magnitude Pruning (IMP)

1) Train the network
2) Remove superfluous structure
3) Reset each weight to its original initialization
4) Train it to convergence and repeat iteratively

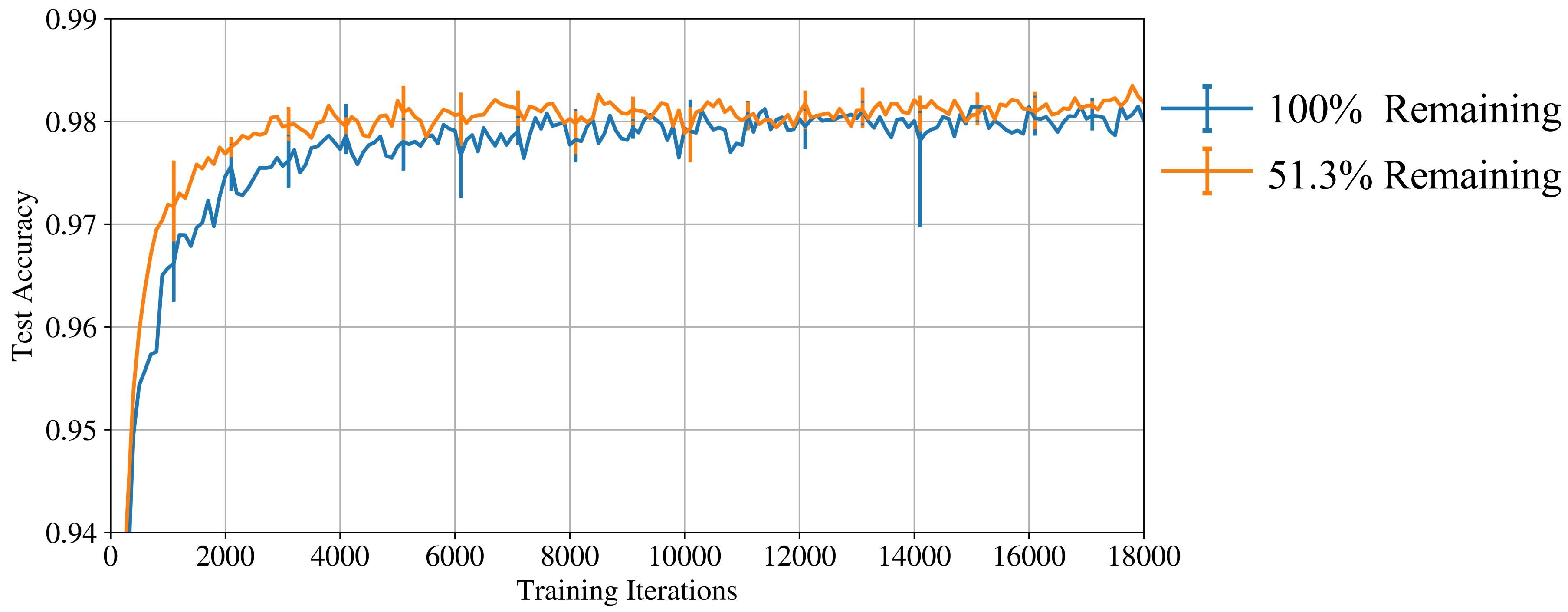


Results



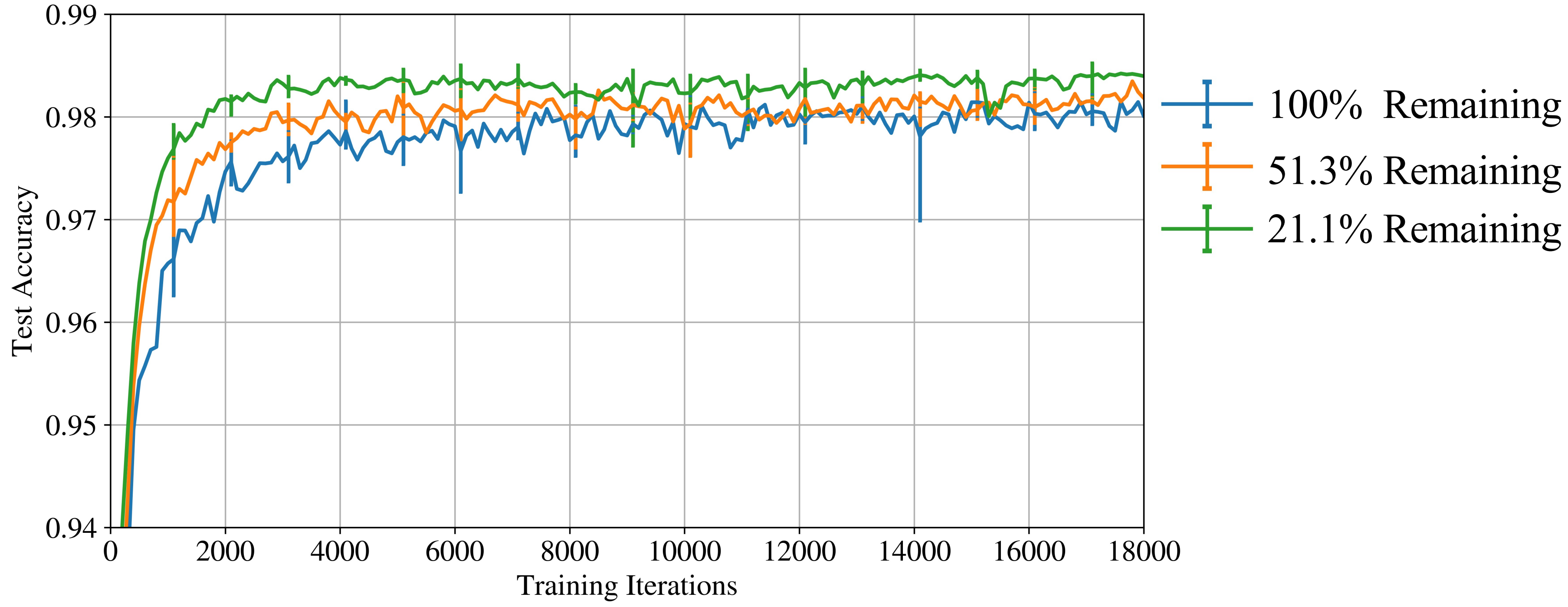
LeNet 300-100-10 for MNIST fully-connected 300K parameters

Results



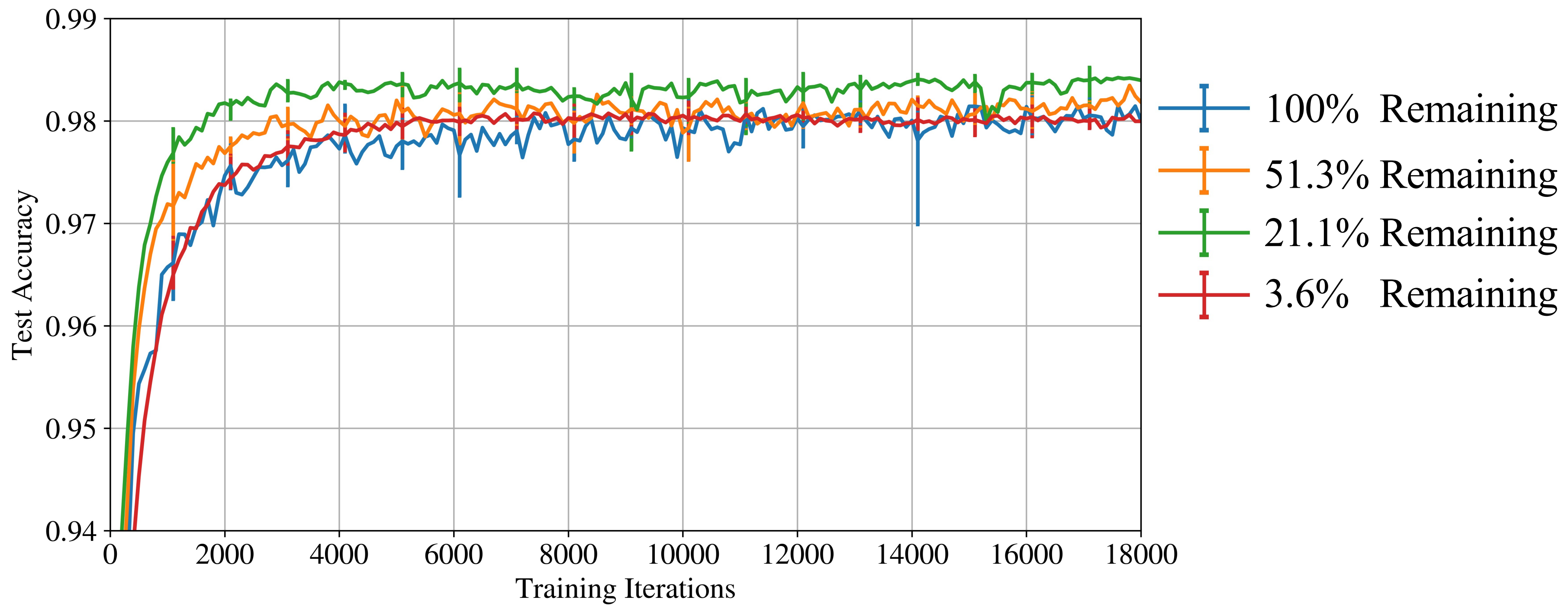
LeNet 300-100-10 for MNIST fully-connected 300K parameters

Results



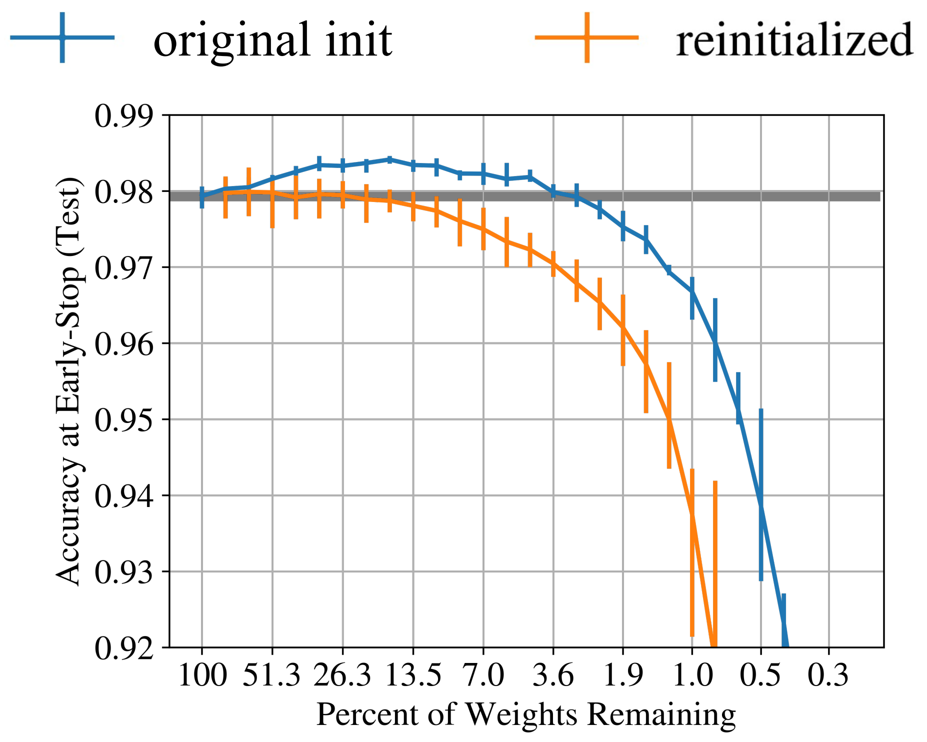
LeNet 300-100-10 for MNIST fully-connected 300K parameters

Results



LeNet 300-100-10 for MNIST fully-connected 300K parameters

Results

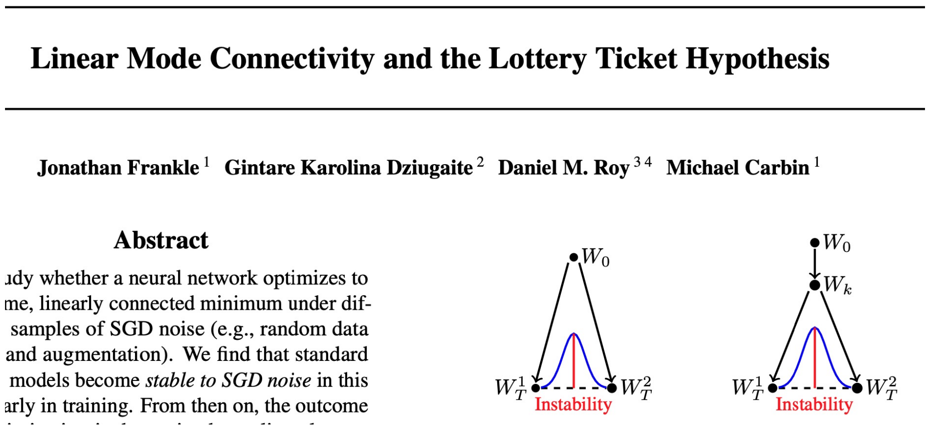


LeNet 300-100-10 for MNIST fully-connected 300K parameters

Limitations

- Subnetworks are found retroactively
- Exploiting pruning for better performance requires software and hardware support
- Small, vision networks and tasks
  - Does not work for larger models and more complicated datasets

Stay Tuned for the PC Meeting



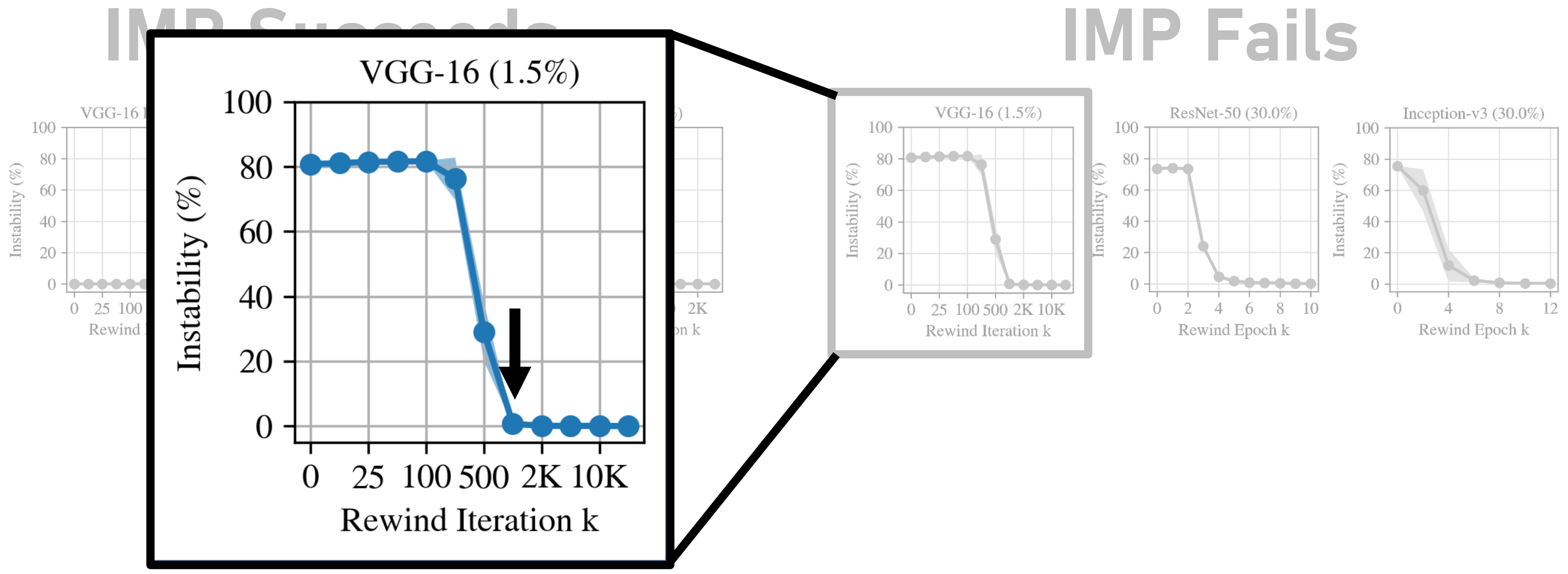
Linear Mode Connectivity and the Lottery Ticket Hypothesis

Jonathan Frankle [1] **Gintare Karolina Dziugaite** [2] **Daniel M. Roy** [3 4] **Michael Carbin** [1]

Abstract

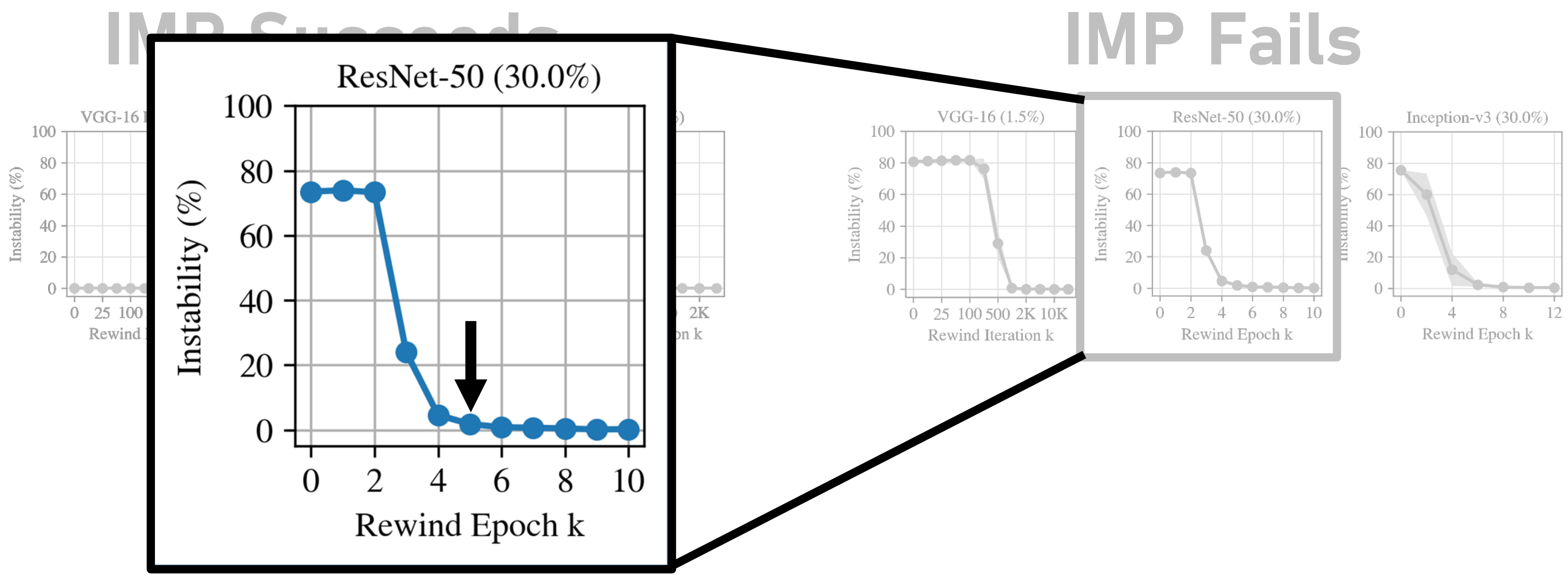
ıdy whether a neural network optimizes to
me, linearly connected minimum under dif-
samples of SGD noise (e.g., random data
and augmentation). We find that standard
models become *stable to SGD noise* in this
arly in training. From then on, the outcome

Sneak Peak of Results For More Complex Models/Datasets



Iteration 1K out of 62K = 1.6% into training

Sneak Peak of Results For More Complex Models/Datasets



Epoch 5 out of 90 = 5.5% into training

High Level Findings in the literature on this topic

Sparse subnetworks reach full accuracy:

In small networks, this occurs at initialization.

For large networks, this occurs early in training.

(we will see the details during the PC meeting).

But this alone does not help us improve the speed up training. However, this is an active area of research, and if we could find the lottery hypothesis ticket in a more efficient manner, then this could significantly improve training cost.

Another Limitation: Structured vs Unstructured Sparsity

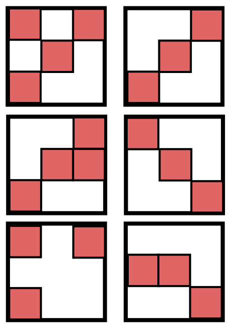
Another Limitation

Unstructured Sparsity:

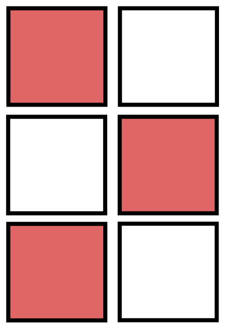
- ➢ Enables significantly higher levels of sparsity (~5%)
- ➢ Harder to accelerate in HW

Structured Sparsity:

- ➢ Easier to accelerate in HW
- ➢ Does not allow as much sparsity (~60%)



Unstructured Sparsity



Structured Sparsity

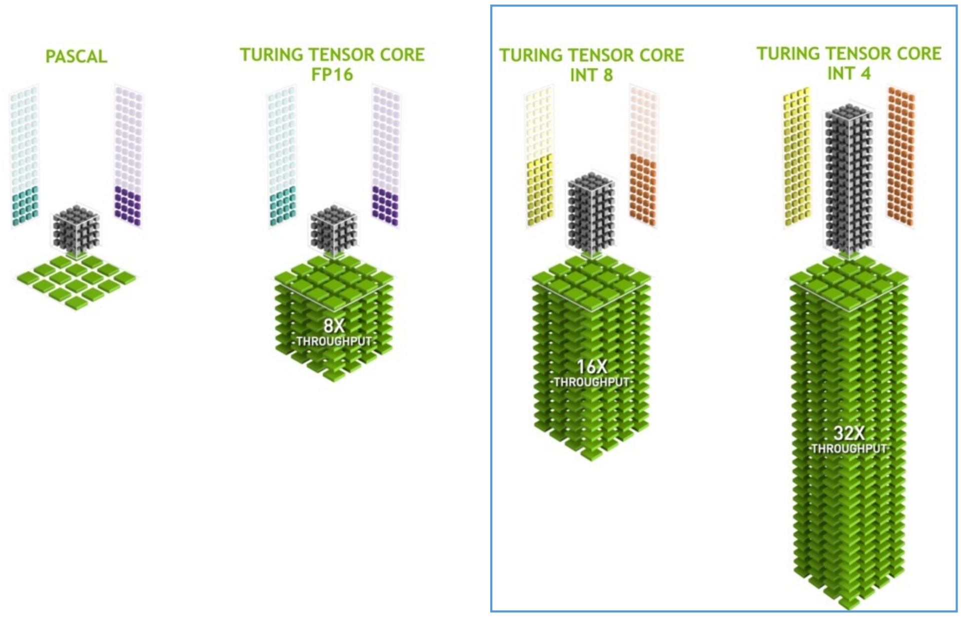
Further Reading

- We only talked about weight sparsity, but there is a large body of work on activation sparsity, token pruning, training free pruning, …
- If you are interested to learn more about pruning:
- Sparsity in Deep Learning: Pruning and growth for efficient inference and training in neural networks
- The state of sparsity in deep neural networks
- What is the State of Neural Network Pruning?

Efficient ML with Quantization

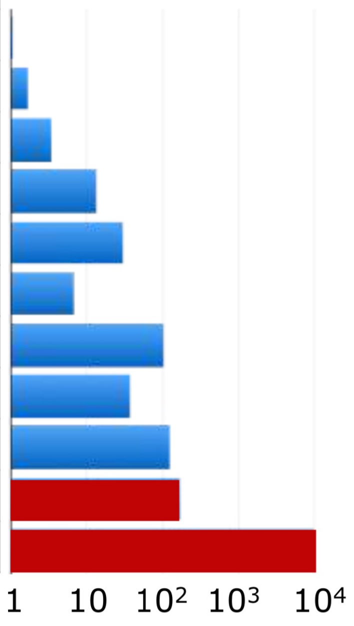
Quantization Enables Fewer Memory Accesses

- Memory accesses are the principal cost in both latency and energy
- Lower precision weights in DNN mean each memory access brings more data values
- More data values few accesses overall



Operation:	Energy (pJ)
8b Add	0.03
16b Add	0.05
32b Add	0.1
16b FP Add	0.4
32b FP Add	0.9
8b Multiply	0.2
32b Multiply	3.1
16b FP Multiply	1.1
32b FP Multiply	3.7
32b SRAM Read (8KB)	5
32b DRAM Read	640

Relative Energy Cost

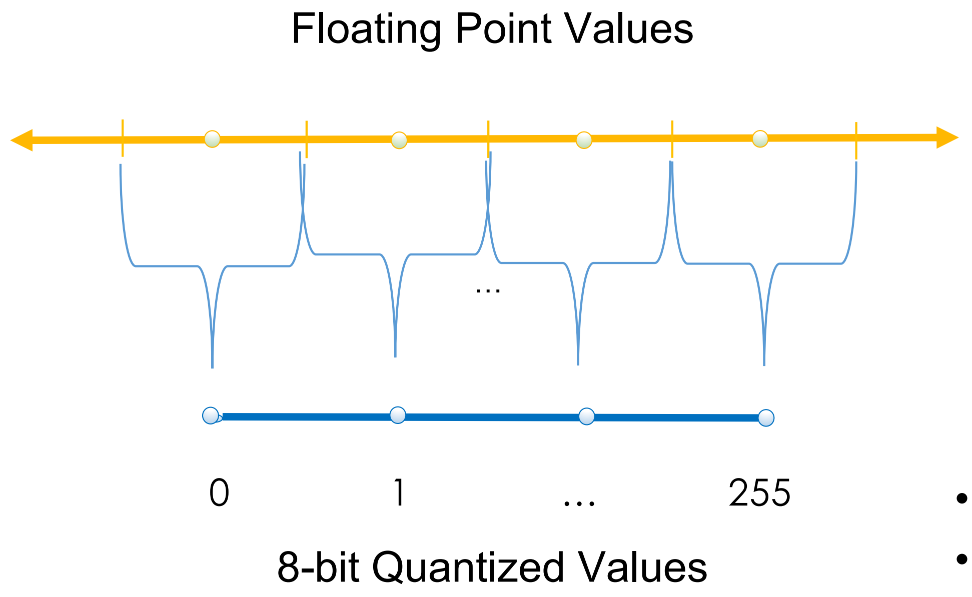


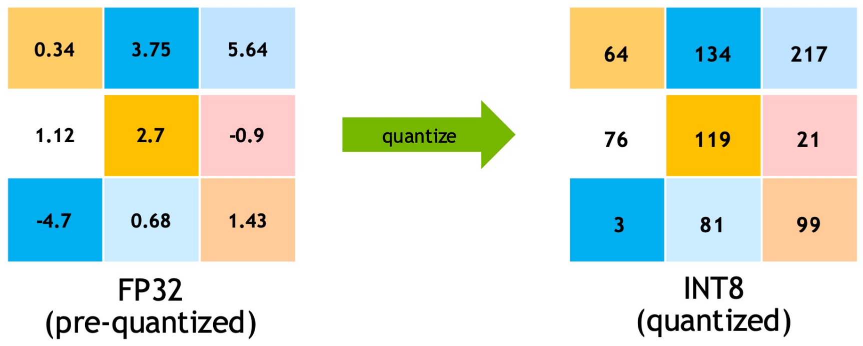
[**Horowitz**, *ISSCC* 2014]

Image Credit: Sdxcentral, nvidia
Table credit: Mark Horowitz

Quantization

➢ **Uniform quantization** is a linear mapping from floating point values to quantized integer values





0.34	3.75	5.64
1.12	2.7	-0.9
-4.7	0.68	1.43

64	134	217
76	119	21
3	81	99

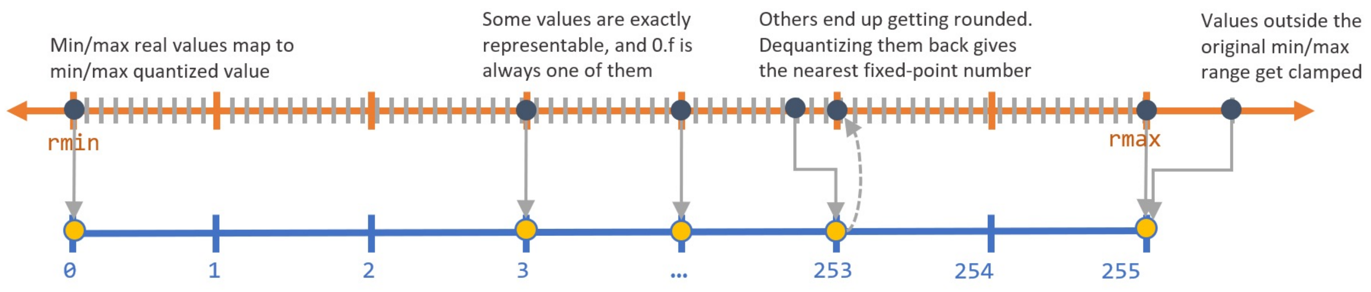
- **Less memory footprint**
- **Faster inference** on integer arithmetic units

Uniform Quantization

- r: Real value
- $r_{max},\ r_{min}$: max/min range of values
- B: Quantization bits
- S (FP32), z (int): Scale and bias
- q: Fixed point quantized values

$$r = \frac{r_{max} - r_{min}}{2\hat{}B - 1}(q - z)$$

$$r = S(q - z)$$

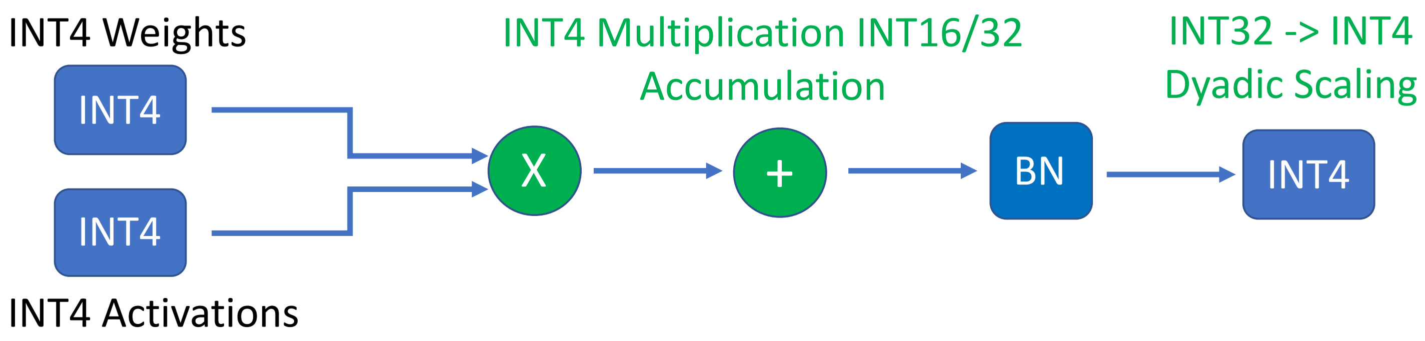


Min/max real values map to min/max quantized value
Some values are exactly representable, and 0.f is always one of them
Others end up getting rounded. Dequantizing them back gives the nearest fixed-point number
Values outside the original min/max range get clamped
rmin
rmax
0
1
2
3
...
253
254
255

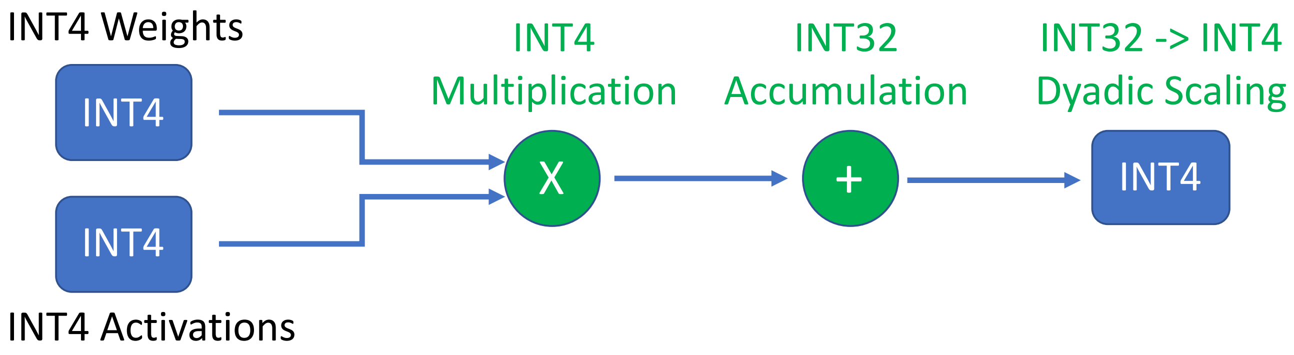
Uniform 8-bit Quantization

Illustration from Sahni Manas

Integer-only Quantization!


INT4 Weights
INT4
INT4
INT4 Activations
INT4 Multiplication INT16/32 Accumulation
X
+
BN
INT32 -> INT4 Dyadic Scaling
INT4

- Key idea is to not fold BN during training and only do BN folding at the end, otherwise you will effectively be training a model without BN which is known to be hard to optimize



Integer-only Quantization Works

(a) *MobileNetV2*

Method	Int	Uni	BL	Precision	Size	BOPS	Top-1
Baseline	✗	✓	73.03	W32A32	13.2	322	73.03
RVQuant (Park et al., 2018)	✗	✗	70.10	W8A8	3.3	20	70.29
CalibTIB(Hubara et al., 2020)	✗	✓	71.90	W8A8	3.3	20	71.60
HAWQV3	✓	✓	73.03	W8A8	3.3	20	**73.01**

Almost not accuracy degradation

(c) *InceptionV3*

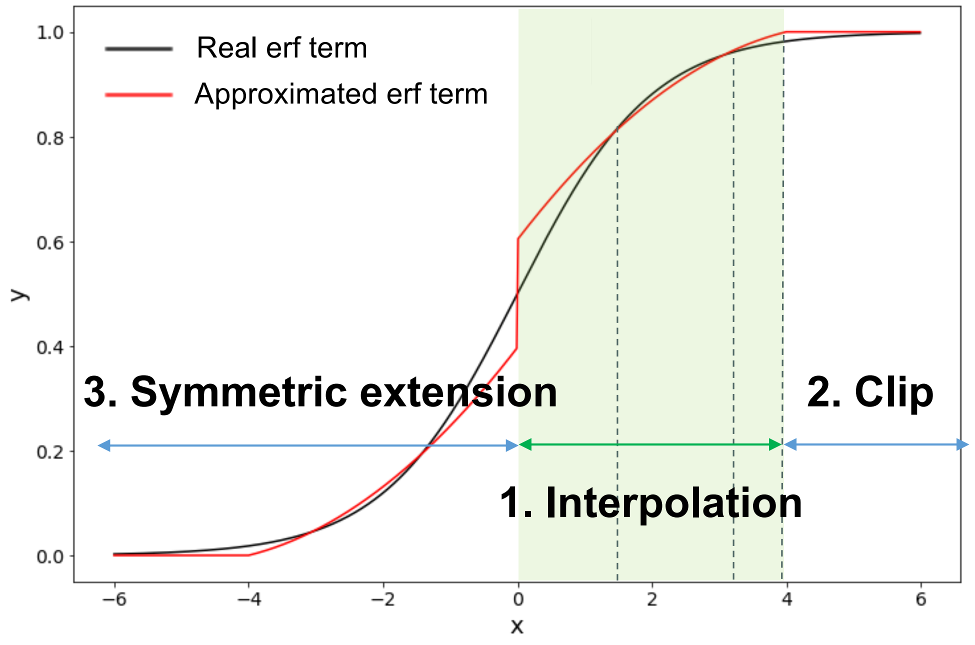
Method	Int	Uni	BL	Precision	Size	BOPS	Top-1
Baseline	✗	✓	78.88	W32A32	90.9	5850	78.88
Integer Only (Jacob et al., 2018)	✓	✓	78.30	W8A8	22.7	366	74.20
RVQuant (Park et al., 2018)	✗	✗	74.19	W8A8	22.7	366	74.22
HAWQV3	✓	✓	78.88	W8A8	22.7	366	**78.76**

Almost not accuracy degradation

Z. Yao*, Z. Dong*, Z. Zheng*, A. Gholami*, E. Tan, J. Li, L. Yuan, Q. Huang, Y. Wang, M. W. Mahoney, K. Keutzer, HAWQ-V3: Dyadic Neural Network Quantization in Mixed Precision, ICML'21.
Online Code: https://github.com/Zhen-Dong/HAWQ

Integer-only Can be applied to general Non-Linearities

- **Proposed Solution:** Approximate erf with 2nd-order polynomial



$$\mathrm{GELU}(x) := x\Phi(x) = x \cdot \overbrace{\frac{1}{2}\left[1 + \mathrm{erf}(x/\sqrt{2})\right]}^{\text{Real erf term}}$$

$$\text{i-GELU}(x) := x \cdot \frac{1}{2}\left[\underbrace{1 + \mathrm{L}\left(\frac{x}{\sqrt{2}}\right)}_{\text{Approximated erf term}}\right]$$

Can be computed with integer arithmetic

Integer-only Softmax

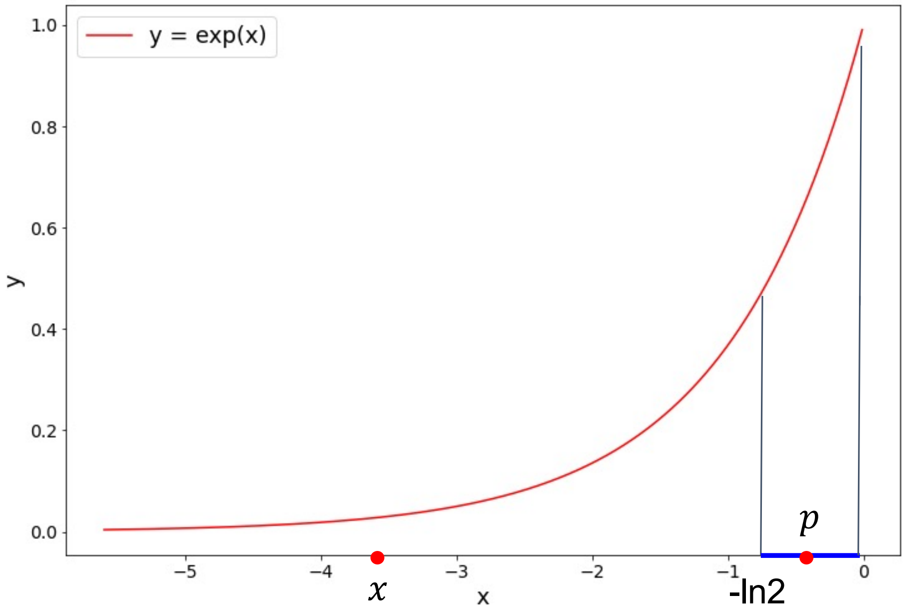
- We can even approximate exp(x) with 2nd-order polynomials with very low error

- Approximate exponential in the interval [-ln2, 0]

$$\tilde{x} = (-\ln 2)z + p$$

Quotient z: integer
Remainder p: lies in [-ln2, 0]

$$\exp(\tilde{x}) = 2^{-z} \underbrace{\exp(p)}_{\text{Compute with approximation}} = \exp(p) >> z$$



Integer-only Quantization Works: Transformers

(a) RoBERTa-Base

	Int-only	MNLI-m	MNLI-mm	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	Avg.
Baseline	✗	**87.8**	**87.4**	**90.4**	**92.8**	94.6	61.2	**91.1**	90.9	78.0	86.0
I-BERT	✓	87.5	**87.4**	90.2	**92.8**	**95.2**	**62.5**	90.8	**91.1**	**79.4**	**86.3**
Diff		-0.3	0.0	-0.2	0.0	+0.6	+1.3	-0.3	+0.2	+1.4	+0.3

(b) RoBERTa-Large

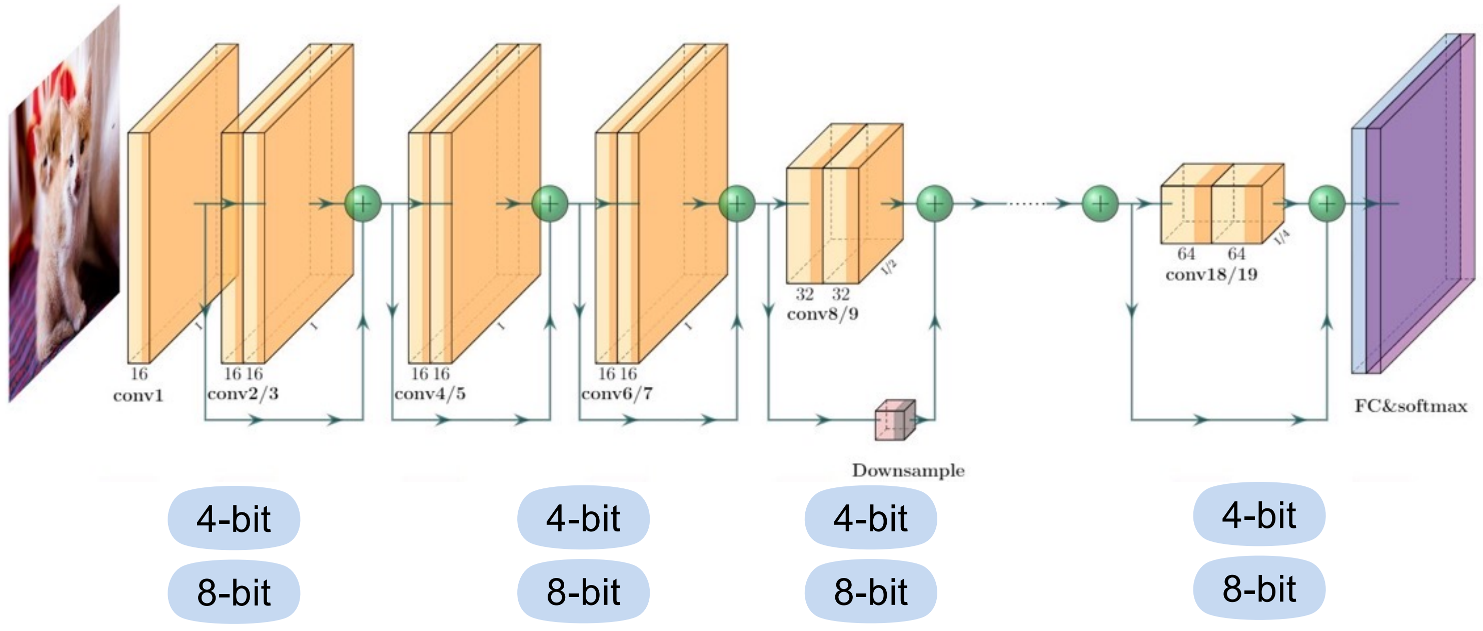
	Int-only	MNLI-m	MNLI-mm	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	Avg.
Baseline	✗	90.0	89.9	92.8	94.1	96.3	68.0	**92.2**	91.8	86.3	89.0
I-BERT	✓	**90.4**	**90.3**	**93.0**	**94.5**	**96.4**	**69.0**	**92.2**	**93.0**	**87.0**	**89.5**
Diff		+0.4	+0.4	+0.2	+0.4	+0.1	+1.0	0.0	+1.2	+0.7	+0.5

- I-BERT is integrated in **HuggingFace** (~8K downloads per month)

S. Kim*, A. Gholami*, Z. Yao*, M. Mahoney, K. Keutzer, **I-BERT: Integer-only BERT Quantization**, ICML, 2021.

What about Sub-INT8 Quantization?

Can we go even further and perform lower precision quantization?



Uniform low precision does not work as it can significantly degrade accuracy => **Use mixed-precision**

- But how should we set the precision for each kernel?

Hessian Aware Quantization

This is somewhat similar to the **Jenga** game. We only remove blocks that are not sensitive.

- ➢ Only use low precision quantization for insensitive parameters (flat loss landscape)
- ➢ Use high precision quantization for sensitive parameters (sharp loss landscape)

This sensitivity can be calculated through Hessian which quantifies the relative sharpness/flatness of the loss landscape.

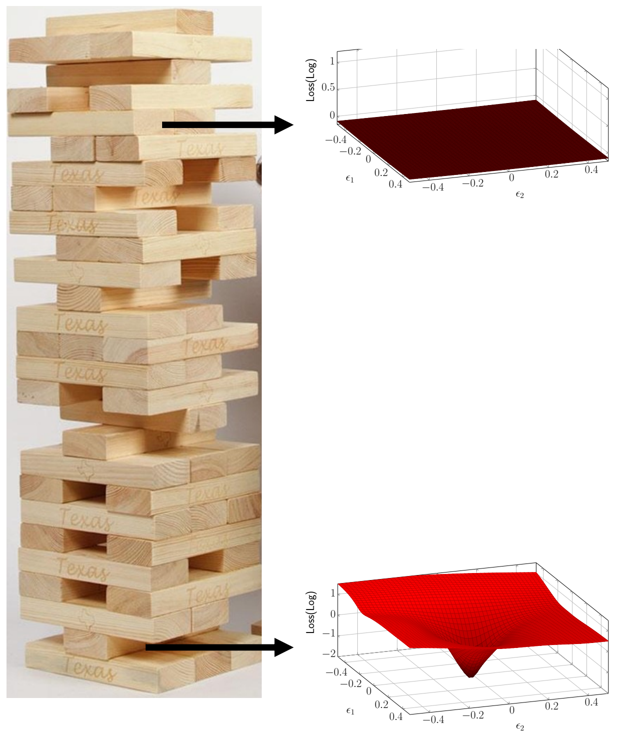
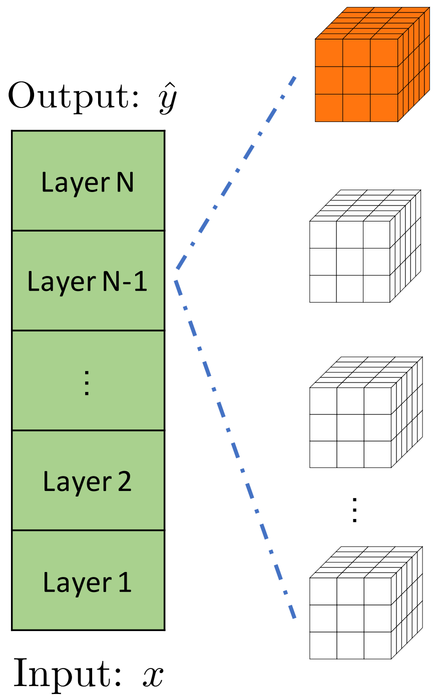


Image from UniversityCoop

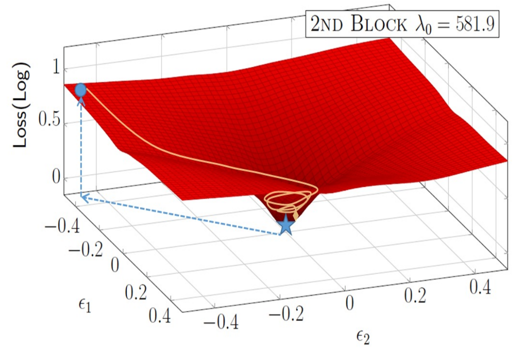


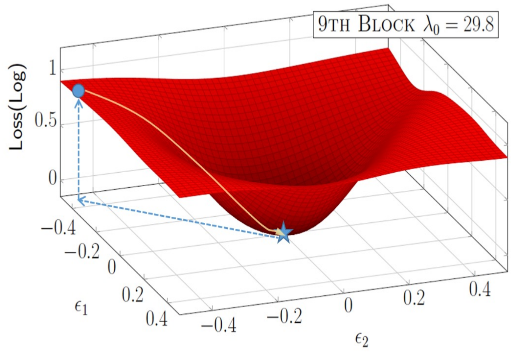
Dong Z, Yao Z, Arfeen D, Gholami A, Mahoney MW, Keutzer K. Hawq-v2: Hessian aware trace-weighted quantization of neural networks. NeurIPS, 2020.
Yu S, Yao Z, Gholami A, Dong Z, Mahoney MW, Keutzer K. Hessian-Aware Pruning and Optimal Neural Implant. WACV, 2022.

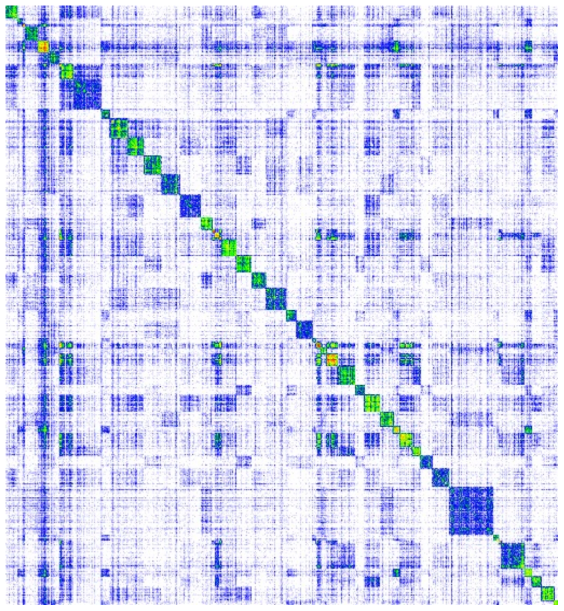
Hessian and Loss Landscape Curvature

Theorem 1 *After quantizing the model to same precision, fine tuning layers that have smaller average trace of Hessian can achieve a smaller loss, compared to layers with larger* ***average trace of Hessian****.*

$$\text{average } tr(H) = \frac{1}{n}\sum_i H_{ii} = \frac{1}{n}\sum_i \lambda_i(H)$$

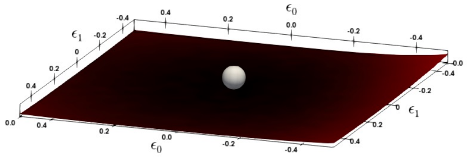


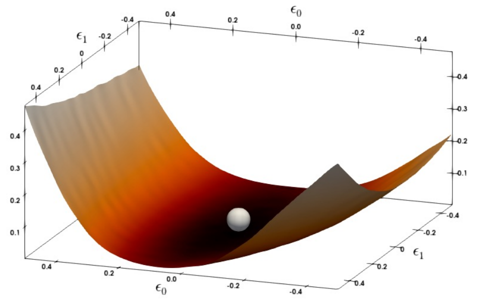


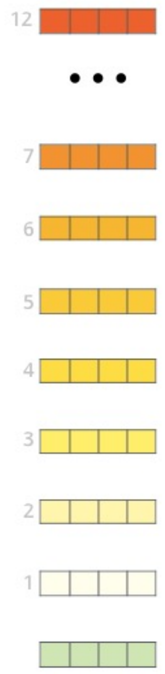


Z. Dong, Z. Yao, D. Arfeen, A. Gholami, M. Mahoney, K. Keutzer, HAWQ-V2: Trace Weighted Hessian Aware Quantization, **NeurIPS 2020.**
Z. Yao*, Z. Dong*, Z. Zheng*, A. Gholami*, E. Tan, J. Li, L. Yuan, Q. Huang, Y. Wang, M. W. Mahoney, K. Keutzer, HAWQ-V3: Dyadic Neural Network Quantization in Mixed Precision, **ICML, 2021**.

Mixed Precision Quantization for NLP


11th Layer


7th Layer



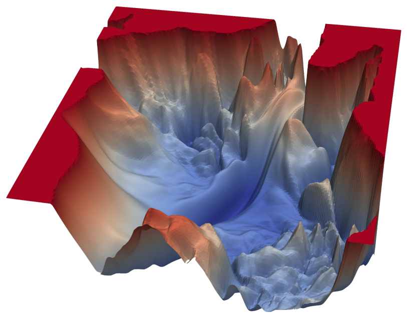
Named Entity Recognition Task

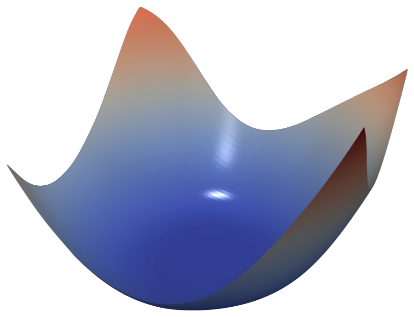
Method	w-bits	e-bits	F_1	Size
Baseline	32	32	95.00	410.9
Q-BERT	8	8	94.79	102.8
DirectQ	4	8	89.86	62.2
Q-BERT	4	8	**94.90**	62.2
DirectQ	3	8	84.92	52.1
Q-BERT	3	8	**94.78**	52.1
Q-BERT$_{\text{MP}}$	2/4 MP	8	**94.55**	52.1
DirectQ	2	8	54.50	42.0
Q-BERT	2	8	**91.06**	42.0
Q-BERT$_{\text{MP}}$	2/3 MP	8	**94.37**	**45.0**

S. Sheng, Z. Dong, J. Ye, L. Ma, Z. Yao, **A. Gholami**, M. Mahoney, K. Keutzer, Q-BERT: Hessian Based Ultra Low Precision Quantization of BERT, **AAAI, 2020.**

Summary:

- Quantization and Pruning of a model depends on the loss landscape
- INT8 Quantization has been proved to be very effective for inference without accuracy degradation
  - Typically yields 2-4x speed up and 4x reduction in memory volume
- Lower Precision Quantization is possible but often there is an accuracy/speed trade-off


Less Robust to Compression


More Robust to Compression

Further Reading

- If you are interested to learn more about quantization see the following papers:
- A survey of quantization methods for efficient neural network inference
- Quantization and training of neural networks for efficient integer-arithmetic-only inference