The University of Texas at Austin
Electrical and Computer Engineering
Cockrell School of Engineering

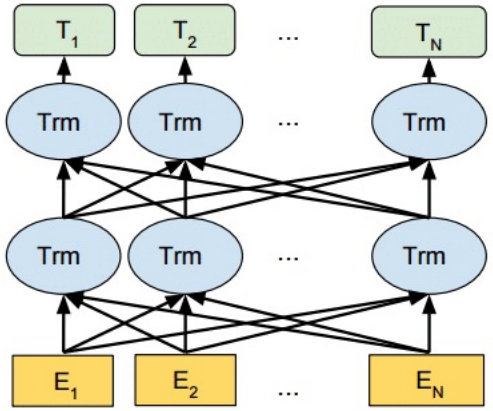
Fall 2023

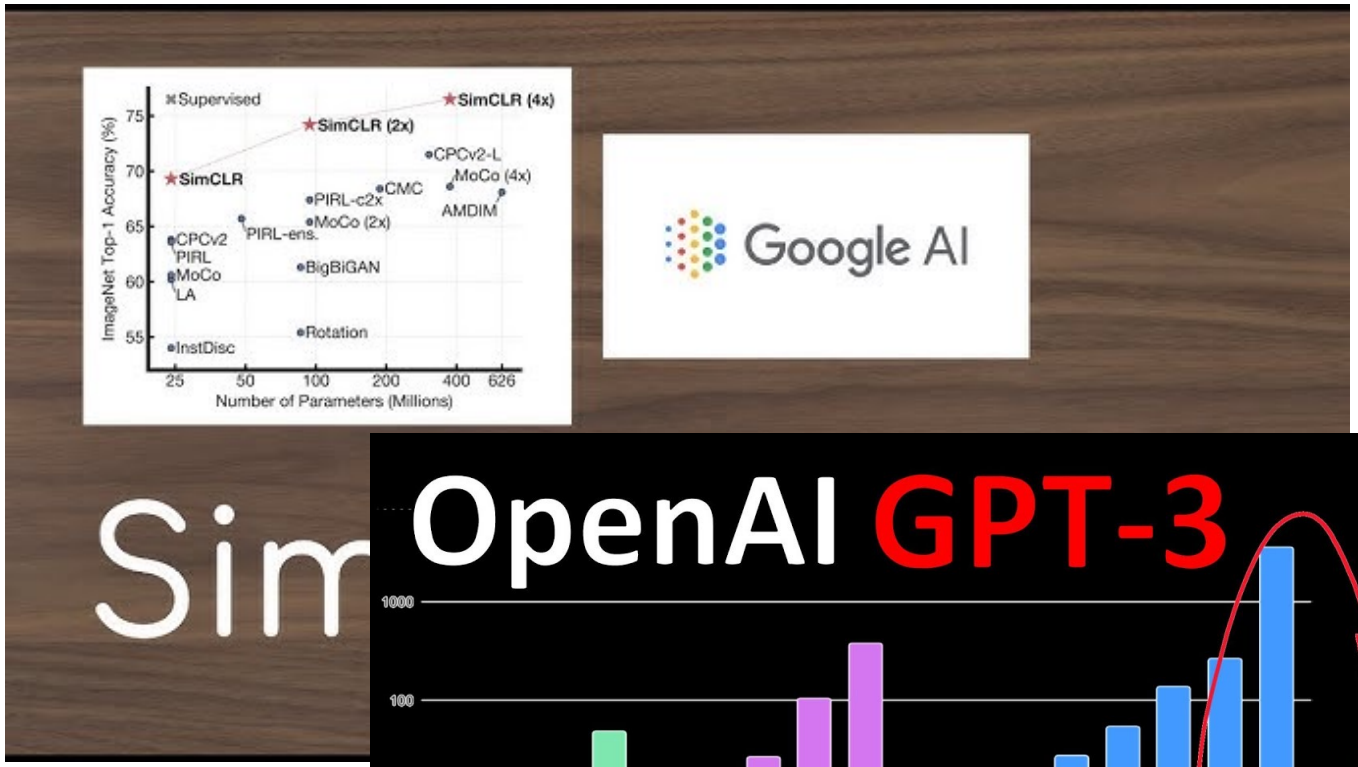
ADVANCED TOPICS IN COMPUTER VISION

Atlas Wang
Associate Professor, The University of Texas at Austin

Visual Informatics Group@UT Austin
https://vita-group.github.io/

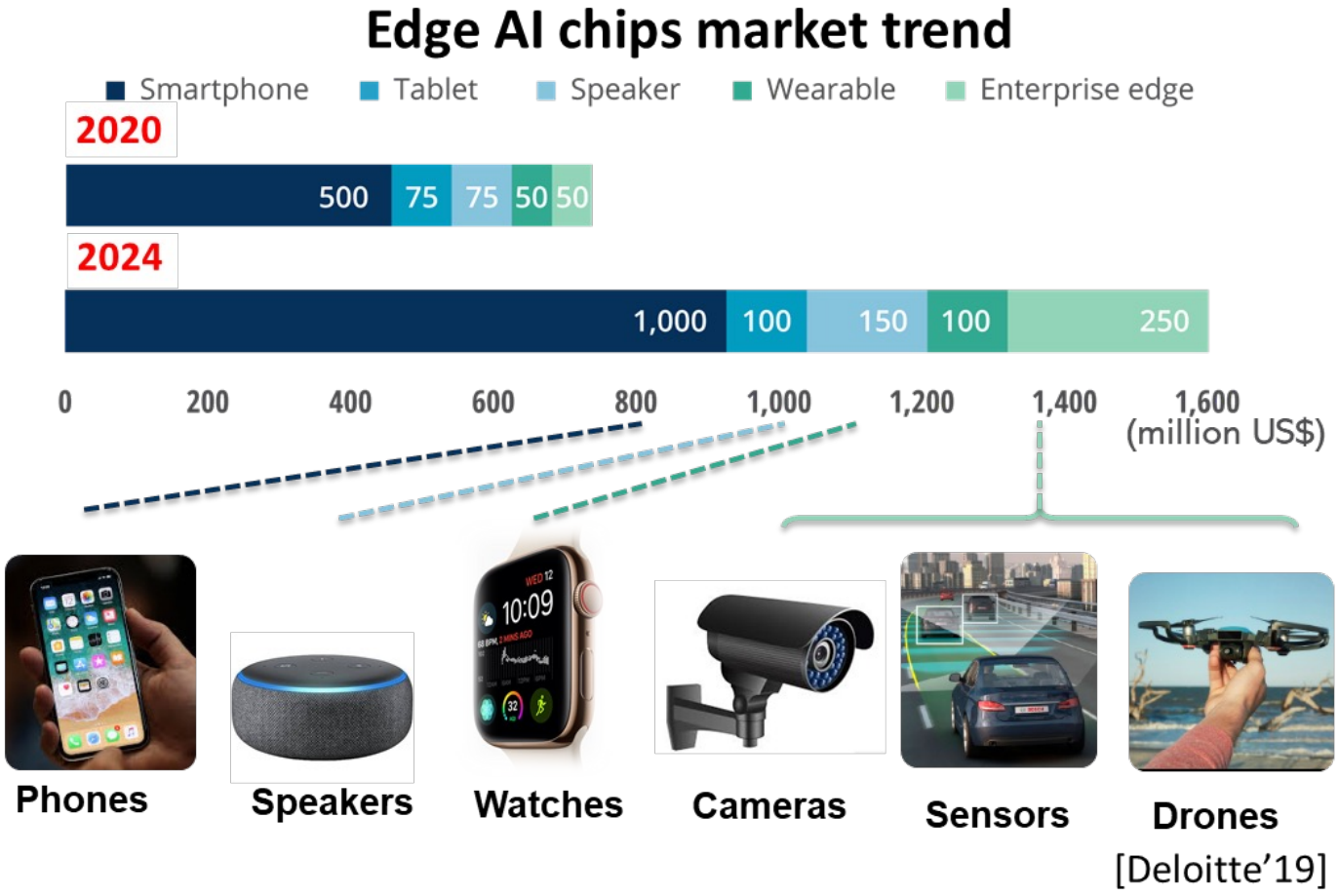
ML researchers like to go BIG



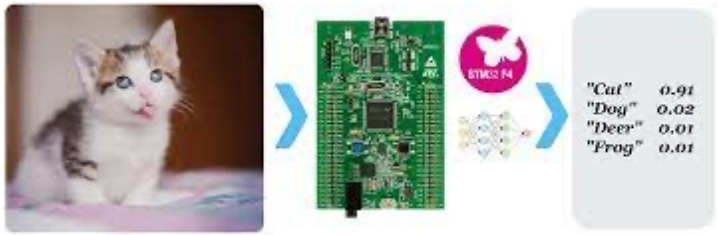


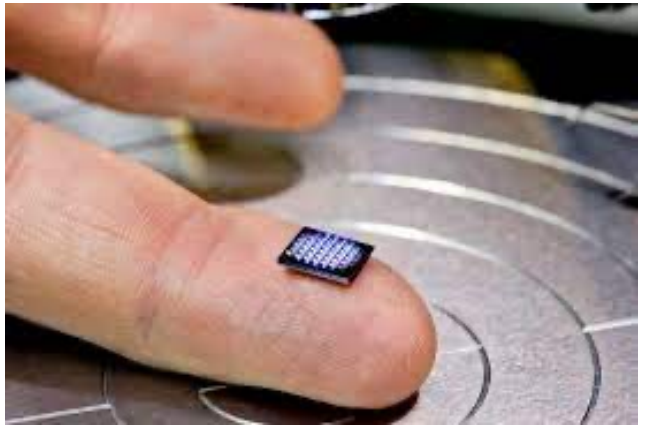
Big NNs seem to be more capable at everything…

….While the world prefers going TINY

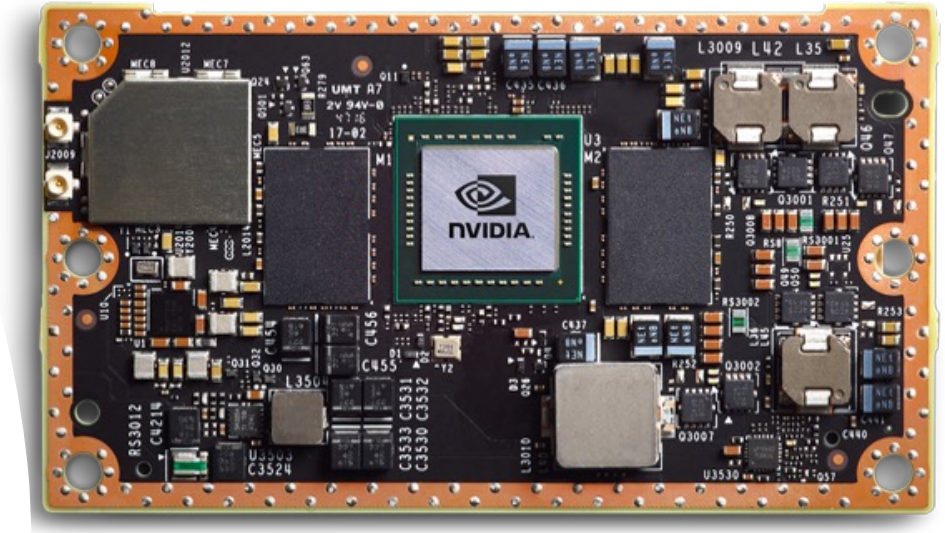


[Deloitte'19]





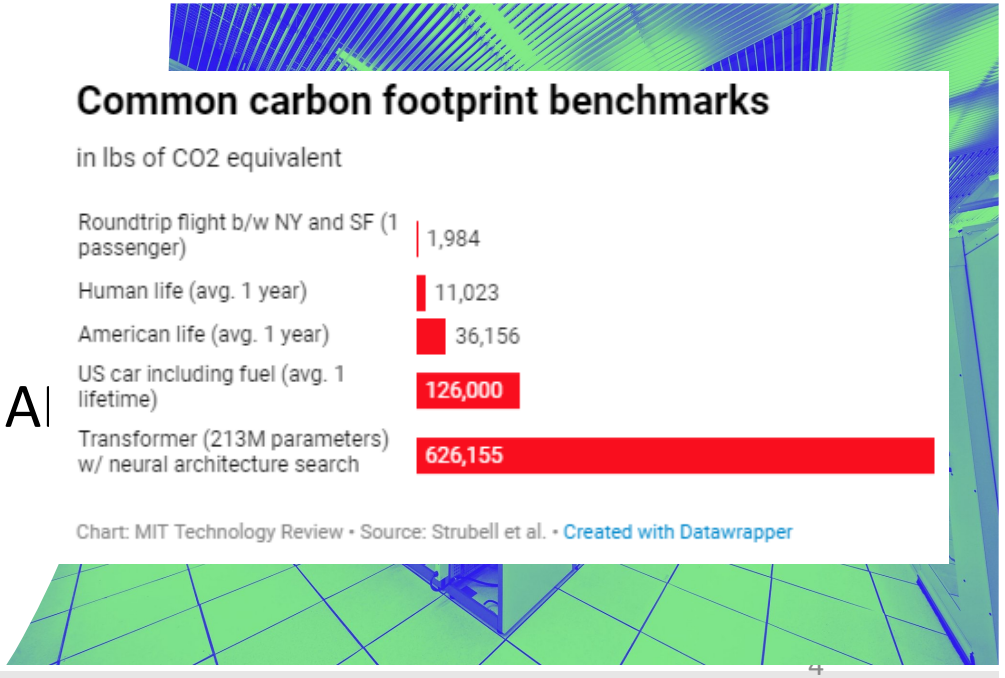
Deep Learning on a Budget



- Three Top Concerns**:**
  - **Storage and Memory**
  - **Speed or Latency**
  - **Energy Efficiency**

- The three goals all pursue “light weight”
- … but they are often **not aligned***
- … so need to **consider all** in implementation
- … and for both **Inference** and **Training**

- Broad economic viability requires energy efficient AI
- Energy efficiency of a brain is **100x better** than current SOTA hardware!



* Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks, IEEE ISSCC 2016

Model Compression

- Training Phase:
  - The easiest way to extract a lot of knowledge from the training data is to learn many different models in parallel.
  - 3B: Big Data, Big Model, Big Ensemble
  - Imagenet: 1.2 million pictures in 1,000 categories.
  - AlexNet: ~ 240Mb, VGG16: ~550Mb
- Testing Phase:
  - Want small and specialist models.
  - Minimize the amount of computation and the memory footprint.
  - Real time prediction
  - Even able to run on mobile devices.

Two Main Streams

- **“Transfer”:** How to transfer knowledge from big general model (teacher) to small specialist models (student)?
  - Example: “Distilling the Knowledge in a Neural Network”, G. Hinton et. al., 2015

- **“Compress”:** How to reduce the size of the same model, during or after training, without losing much accuracy.
  - Example: “Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding”, S. Han et. al., 2016

- **Comparison:** Knowledge Transfer provides a way to train a new small model inheriting from big general models, while Deep Compression Directly does the surgery on big models, using a pipeline: pruning, quantization & Huffman coding.

Knowledge Transfer/“Distillation”: Main Idea

- Introduce “Soft targets” as one way to transfer the knowledge from big models.
  - Classifiers built from a softmax function have a great deal more information contained in them than just a classifier;
  - The correlations in the softmax outputs are very informative.

- Hard Target: the ground truth label (one-hot vector)
- Soft Target: $q_i = \frac{exp(z_i/T)}{\sum_j exp(z_j/T)}$ T is “temperature”, z is logit

- More information in soft targets

cow	dog	cat	car	
0	1	0	0	original hard targets

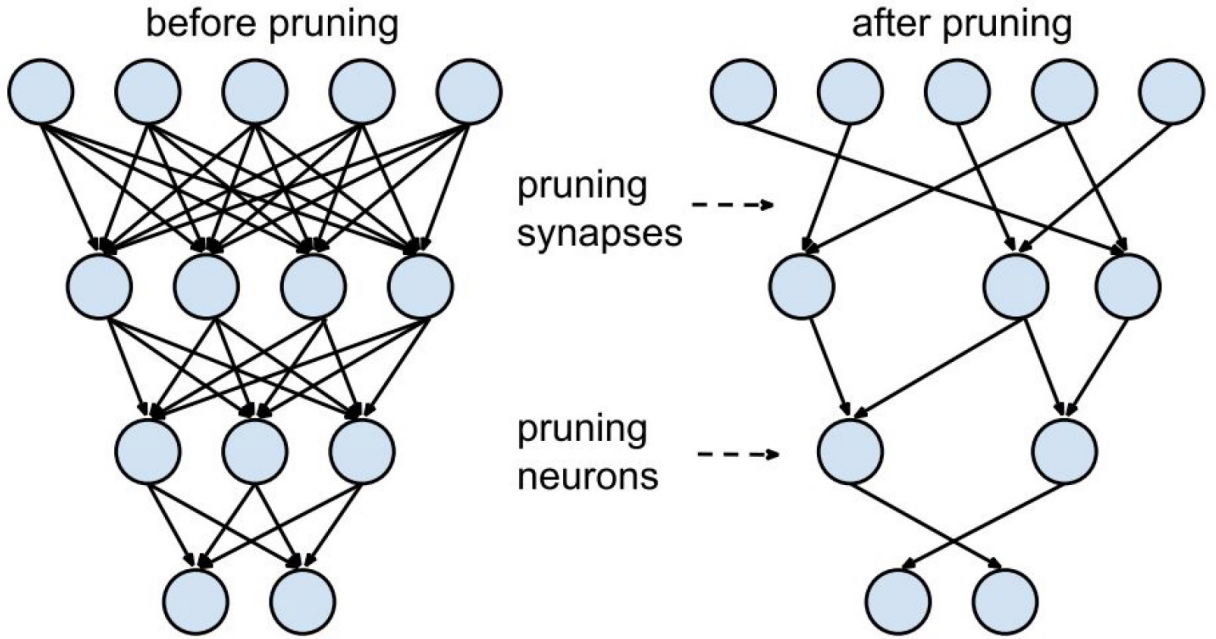
cow	dog	cat	car	
.05	.3	.2	.005	softened output of ensemble

Hinton’s Observation: If we can extract the knowledge from the data using very big models or ensembles of models, it is quite easy to distill most of it into a much smaller model for deployment.

More follow-up observations: teachers can be weak, or even the same as student …

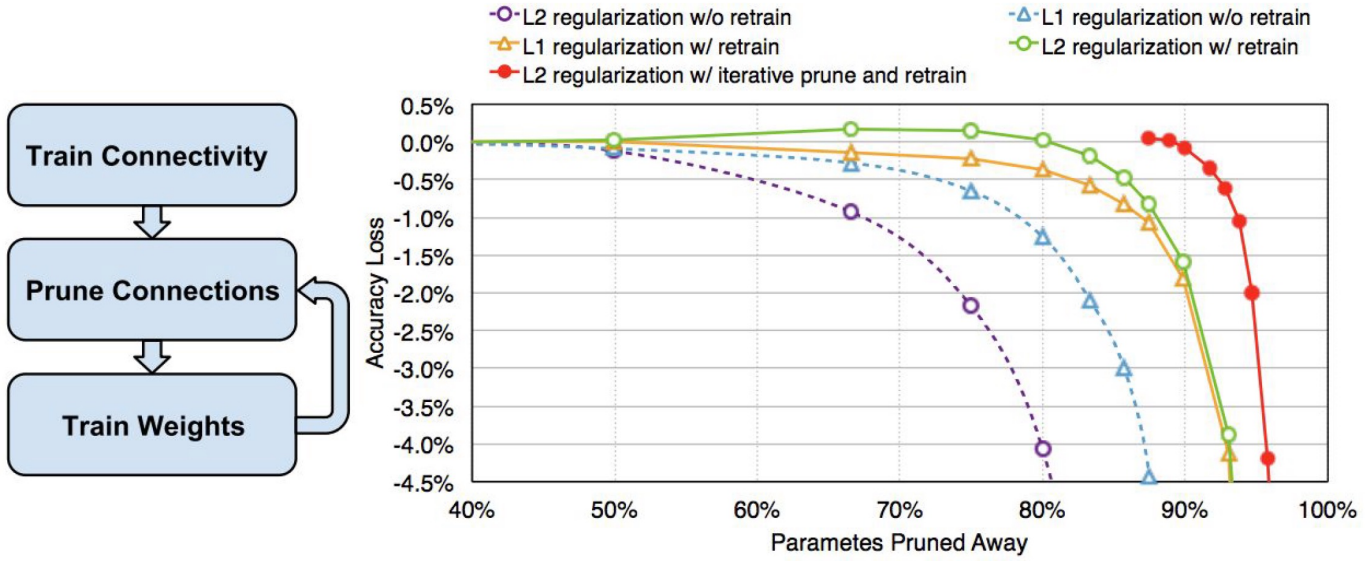
Deep Compression: Main Idea (i)

Pruning



Deep Compression: Main Idea (ii)

Retrain to Recover Accuracy



Network pruning can save 9x to 13x parameters without drop in accuracy

Deep Compression: Main Idea (iii)

Weight Sharing (Trained Quantization)

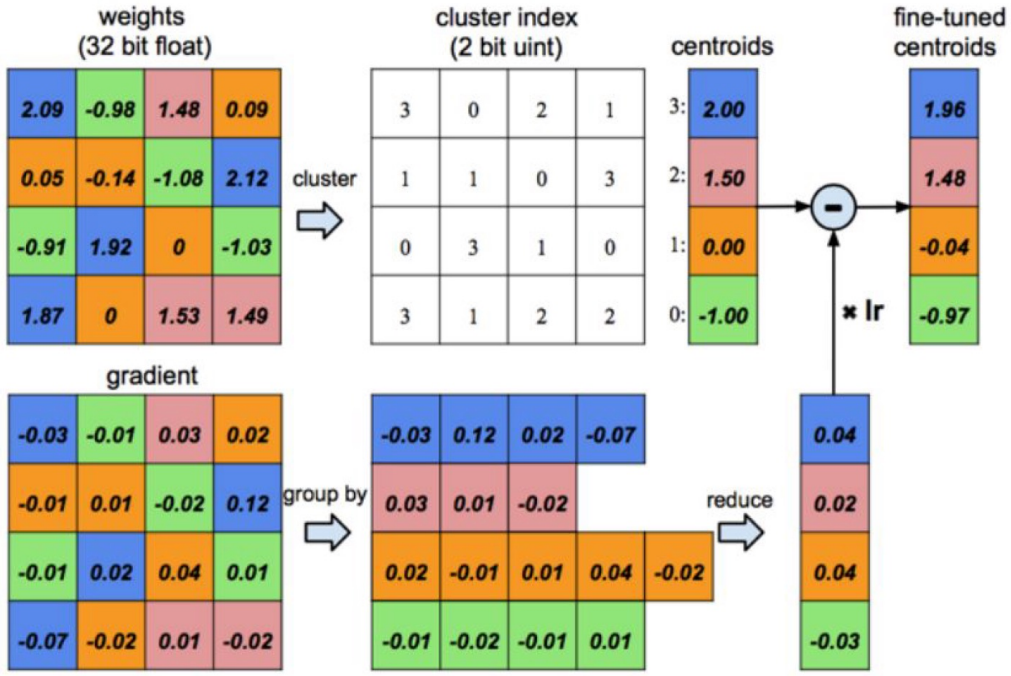
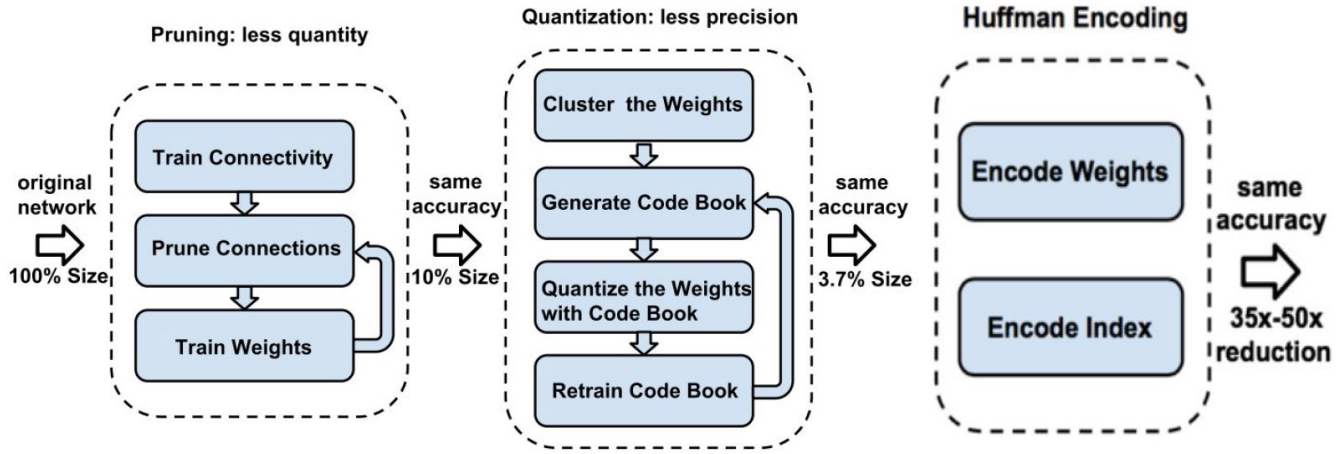


Figure 3: Weight sharing by scalar quantization (top) and centroids fine-tuning (bottom)

Deep Compression: Main Idea (iv)

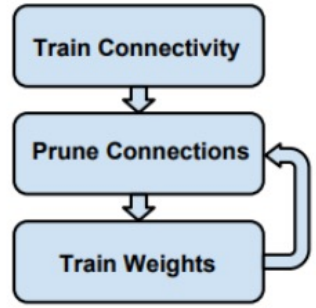
Huffman Coding



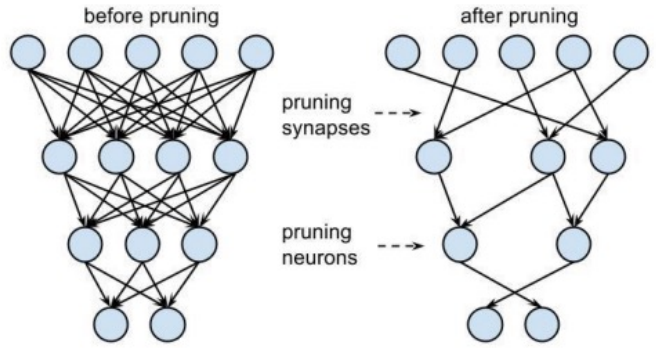
More About Pruning

- Determining **low-saliency parameters**, given a pre-trained network
- Follows the framework proposed by LeCun et al. (1990):

1. **Train** a deep model until convergence
2. **Delete** "unimportant" connections w.r.t. a certain criteria
3. **Re-train** the network
4. **Iterate** to step 2, or stop



- Defining which connection is unimportant can vary
  - Weight magnitudes (L^2, L^1, …)
  - Mean activation [Molchanov et al., 2016]
  - Avg. % of Zeros (APoZ) [Hu et al., 2016]
  - Low entropy activation [Luo et al., 2017]
  - …



Human Brain Prunes too!

50 Trillion Synapses

This image is in the public domain

Newborn

1000 Trillion Synapses

This image is in the public domain

1 year old

500 Trillion Synapses

This image is in the public domain

Adolescent

Optimal Brain Damage (OBD)

- Network pruning **perturbs weights $\mathbf{W}$** by **zeroing** some of them
- How the loss L would be changed when $\mathbf{W}$ is perturbed?

- **OBD** approximates L by the 2nd order Taylor series:

$$\delta L \simeq \underbrace{\sum_i \frac{\partial L}{\partial w_i}\delta w_i}_{\text{1st order}} + \underbrace{\frac{1}{2}\sum_i \frac{\partial^2 L}{\partial {w_i}^2}\delta w_i^2 + \frac{1}{2}\sum_{i,j} \frac{\partial^2 L}{\partial w_i \partial w_j}\delta w_i \delta w_j}_{\text{2nd order}} + O(||\delta \mathbf{W}||^3)$$

- **Problem:** Computing $H = \left(\frac{\partial L}{\partial w_i \partial w_j}\right)_{i,j}$ is usually intractable
  - Requires $O(n^2)$ on # weights
  - Neural networks usually have enormous number of weights
    - e.g. AlexNet: **60M** parameters ⇒ H consists ≈ $\mathbf{3.6 \times 10^{15}}$ elements

Optimal Brain Damage (OBD)

- **Problem:** Computing $H = \left(\frac{\partial L}{\partial w_i \partial w_j}\right)_{i,j}$ is usually intractable
- Two additional assumptions for tractability

  1. **Diagonal** approximation: $H = \frac{\partial^2 L}{\partial w_i \partial w_j} = 0 \quad \text{if} \quad i \neq j$
  2. **Extremal** assumption: $\frac{\partial L}{\partial w_i} = 0 \quad \forall i$
     - **W** would be in a local minima if it's pre-trained

- Now we get: $\delta L \simeq \frac{1}{2} \sum_i \frac{\partial^2 L}{\partial {w_i}^2} \delta w_i^2 + O(||\delta \mathbf{W}||^3)$
  - It only needs $\text{diag}(H) := \left(\frac{\partial^2 L}{\partial w_i^2}\right)_i$
- $\mathbf{diag}(\boldsymbol{H})$ can be computed in $O(n)$, allowing a backprop-like algorithm
  - For details, see [LeCun et al., 1987]

Optimal Brain Damage (OBD)

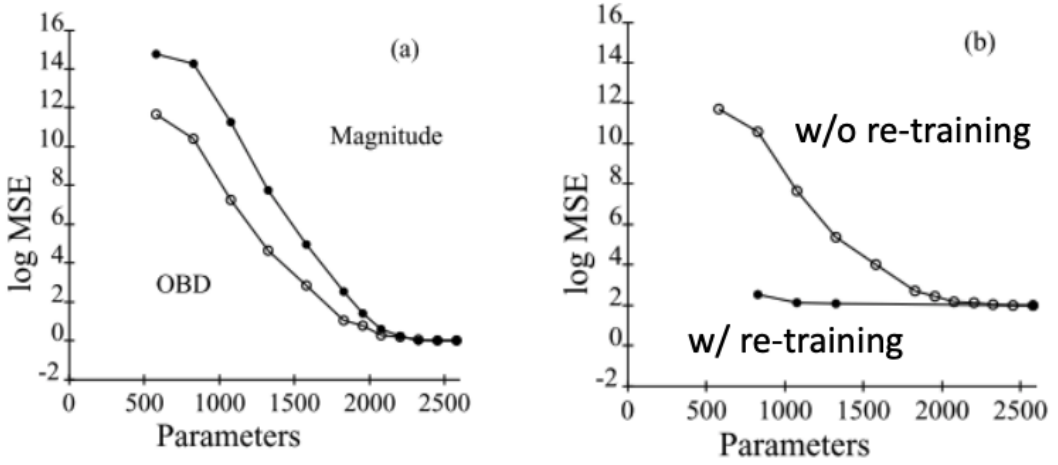
- How the loss L would be changed when $\mathbf{W}$ is perturbed?

$$L(\delta \mathbf{W}) \simeq \frac{1}{2} \sum_i \frac{\partial^2 L}{\partial {w_i}^2} \delta w_i^2 =: \sum_i \frac{1}{2} h_{ii} \delta w_i^2$$

- The **saliency** for each weight $\Rightarrow$ $s_i := \frac{1}{2} h_{ii} |w_i|^2$

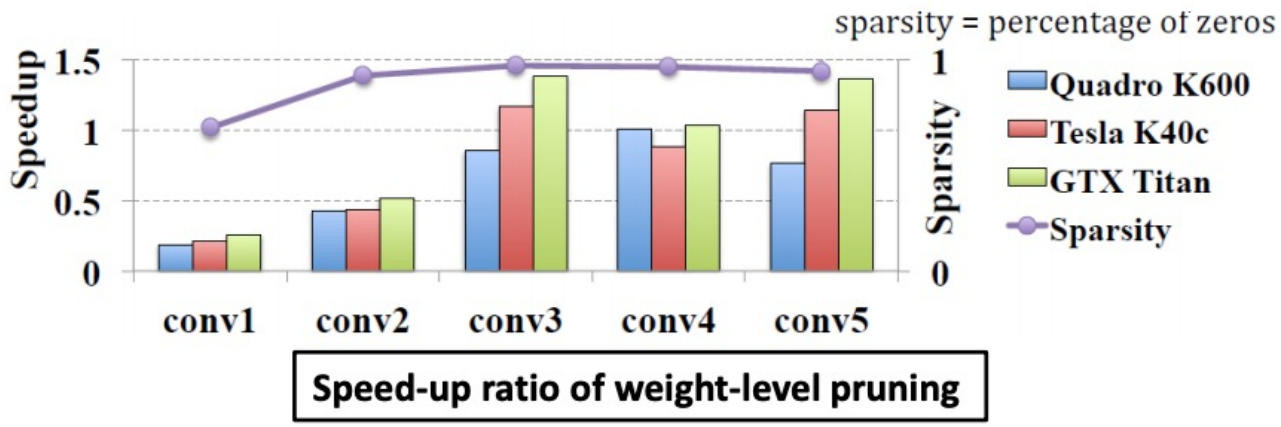
$$s_i := |w_i|$$

- OBD shows robustness on pruning compared to magnitude-based deletion
- After re-training, the original test accuracy is recovered



Structured Sparsity

- "Un-structured" weight-level pruning may not engage a **practical speed-up**
  - Despite of extremely high sparsity, actual speed-ups in GPU is limited



Speed-up ratio of weight-level pruning

Non-structured sparsity (poor data pattern)

Structured sparsity (regular data pattern)

5× speedup after concatenation of nonzero rows and columns

Structured sparsity

- **Structured sparsity** can be induced by adding group-lasso regularization

$$\min_{\mathbf{W}} \mathcal{L}(\mathbf{W}) + \lambda \sum_{l=1}^{L} R_g(\mathbf{W}^{(l)}), \;\; R_g(\mathbf{w}) = \sum_{g=1}^{G} \|\mathbf{w}^{(g)}\|_2$$

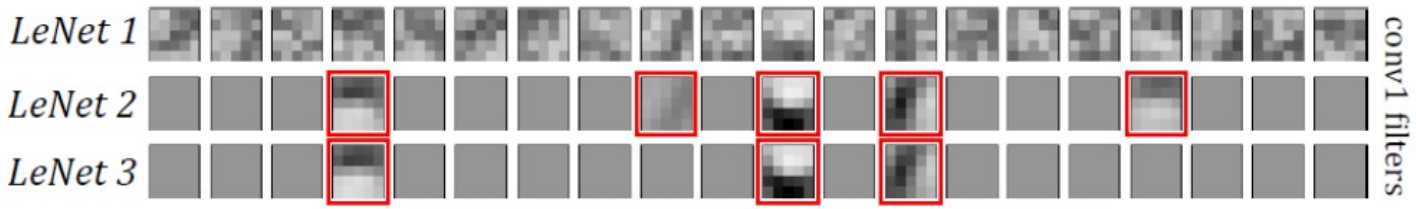
- **Filter-wise** and **channel-wise**: (# filters, # channels)

$$R_g(\mathbf{W}^{(l)}) = \sum_{n_l=1}^{N_l} \|\mathbf{W}^{(l)}_{n_l,:,:,:}\|_2 + \sum_{c_l=1}^{C_l} \|\mathbf{W}^{(l)}_{:,c_l,:,:}\|_2$$

Table 1: Results after penalizing unimportant filters and channels in *LeNet*

LeNet #	Error	Filter # [§]	Channel # [§]	FLOP [§]	Speedup [§]
1 (*baseline*)	0.9%	20—50	1—20	100%—100%	1.00×—1.00×
2	0.8%	5—19	1—4	25%—7.6%	1.64×—5.23×
3	1.0%	3—12	1—3	15%—3.6%	1.99×—7.44×

[§] In the order of *conv1*—*conv2*



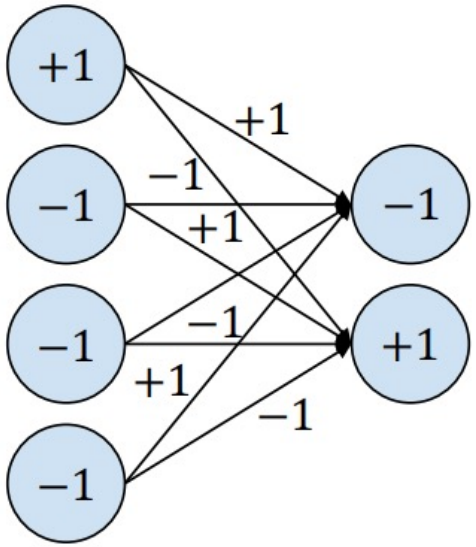
Fewer but smoother feature extractors

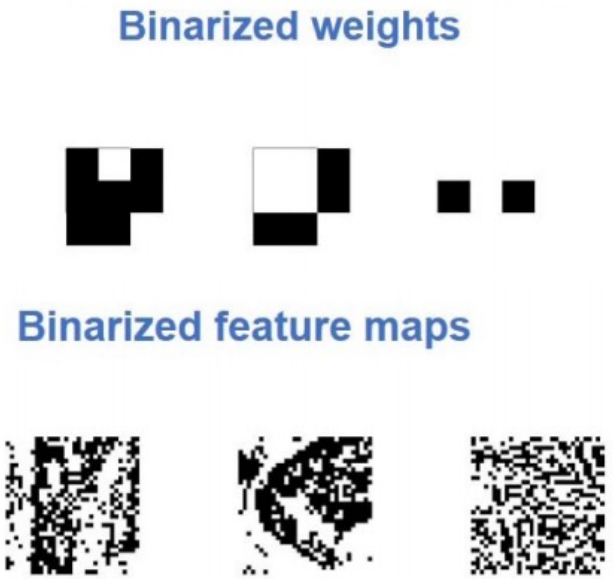
More About Quantization

- Neural networks can be even **binarized** (**+1** or **-1**)
  - DNNs trained to use binary weights and binary activations

- Expensive **32-bit MAC** (**M**ultiply-**AC**cumulate) ⇒ Cheap **1-bit XNOR-Count**
  - "MAC == XNOR-Count": when the weights and activations are ± 1

1s in bits





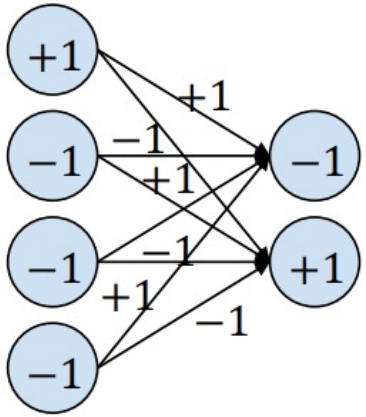
Binary Neural Networks

- **Idea**: Training real-valued nets (W_r) treating binarization (W_b) as noise
  - Training W_r is done by **stochastic gradient descent**
- **Binarization** ($W_r \rightarrow W_b$) occurs for each forward propagation
  - On each of **weights**: $W_b = \text{sign}(W_r)$
  - … also on each **activation**: $a_b = \text{sign}(a_r)$
- Gradients for W_r is estimated from $\frac{\partial L}{\partial W_b}$ [Bengio et al., 2013]
  - "Straight-through estimator": Ignore the binarization during backward!

$$\frac{\partial L}{\partial W_r} = \frac{\partial L}{\partial W_b} \mathbf{1}_{|W_r| \leq 1}$$

$$\frac{\partial L}{\partial a_r} = \frac{\partial L}{\partial a_b} \mathbf{1}_{|a_r| \leq 1}$$

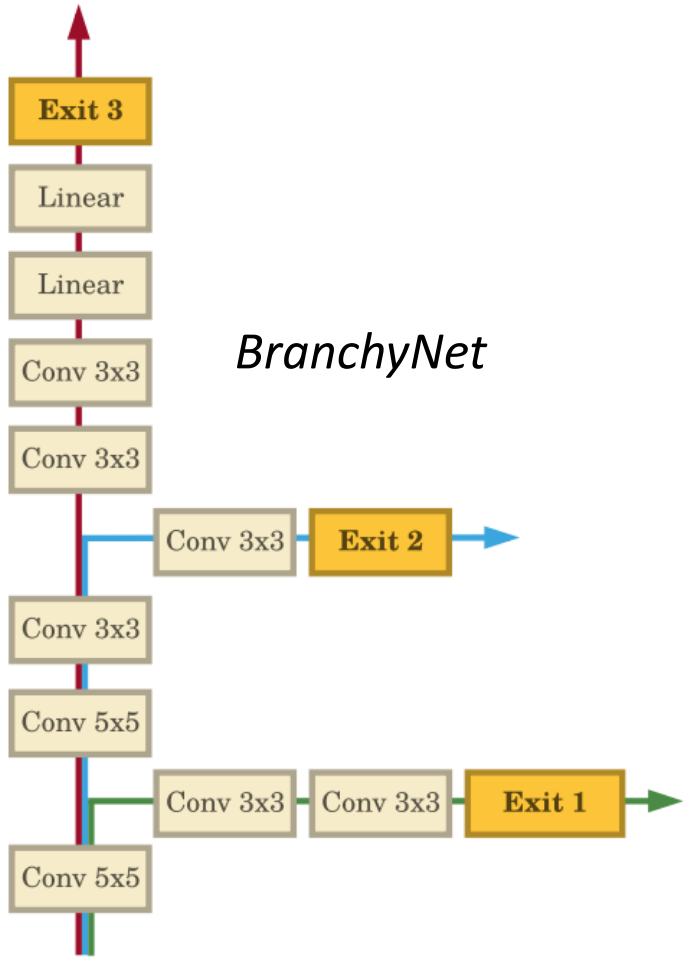
  - Cancelling gradients for better performance
    - When the value is too large



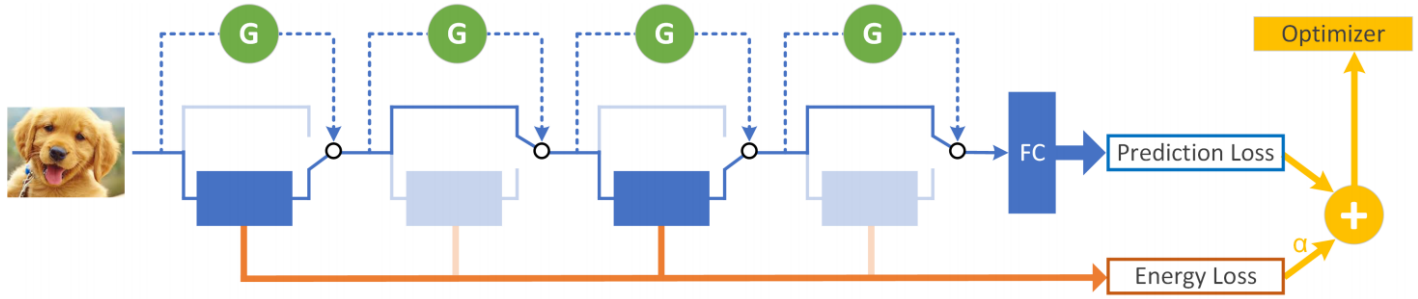
Dynamic Inference

- Only execute a fraction of the network per needed
- Can enable both “input-dependent” and “resource-dependent” forms

BranchyNet



SkipNet

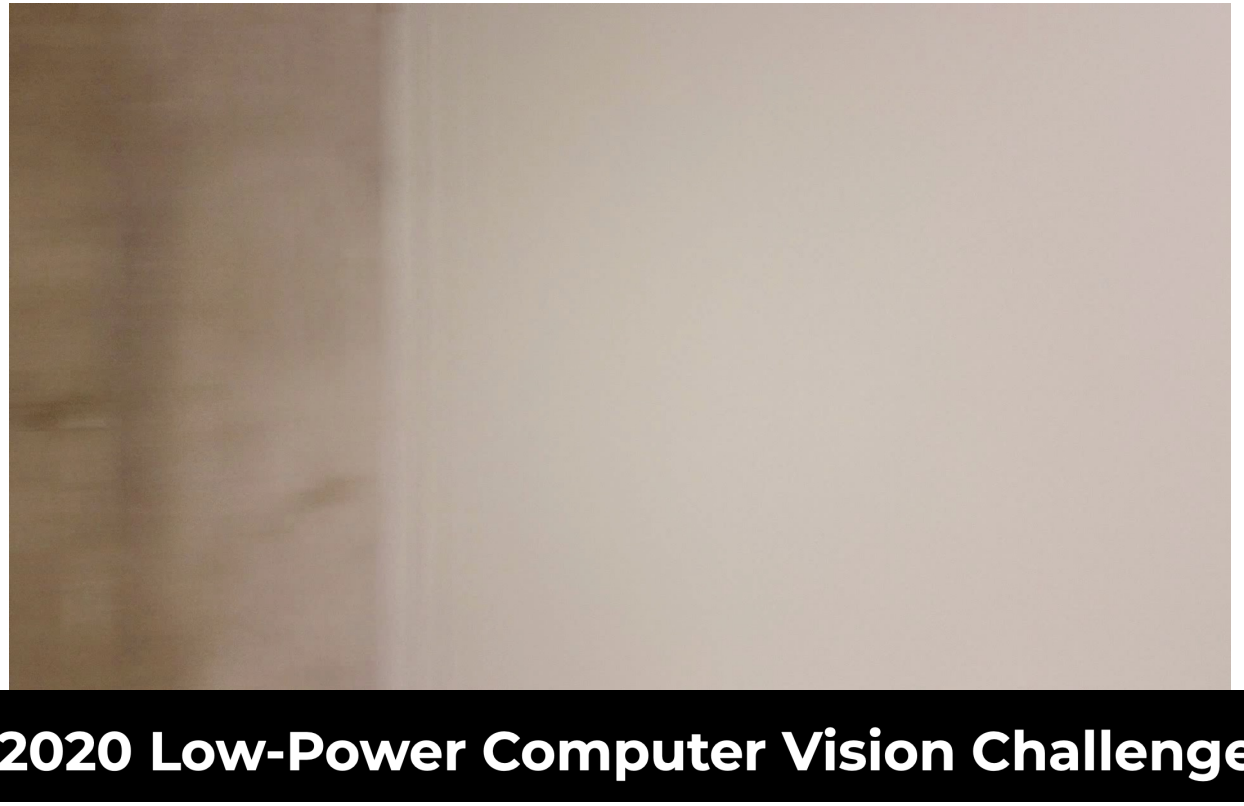


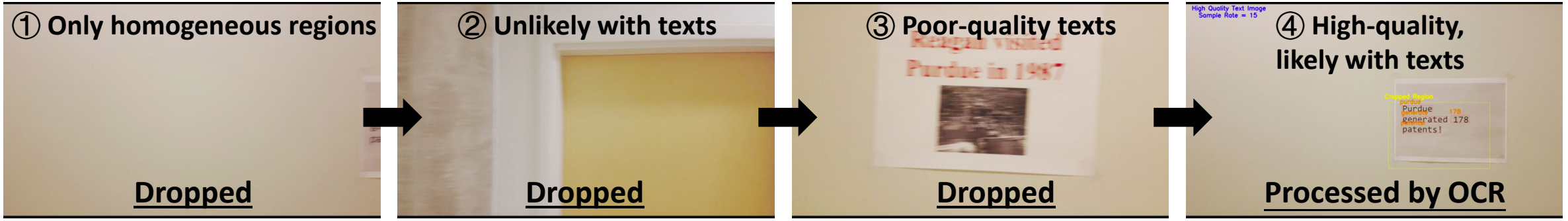
Real-World Efficient ML: Way to Go

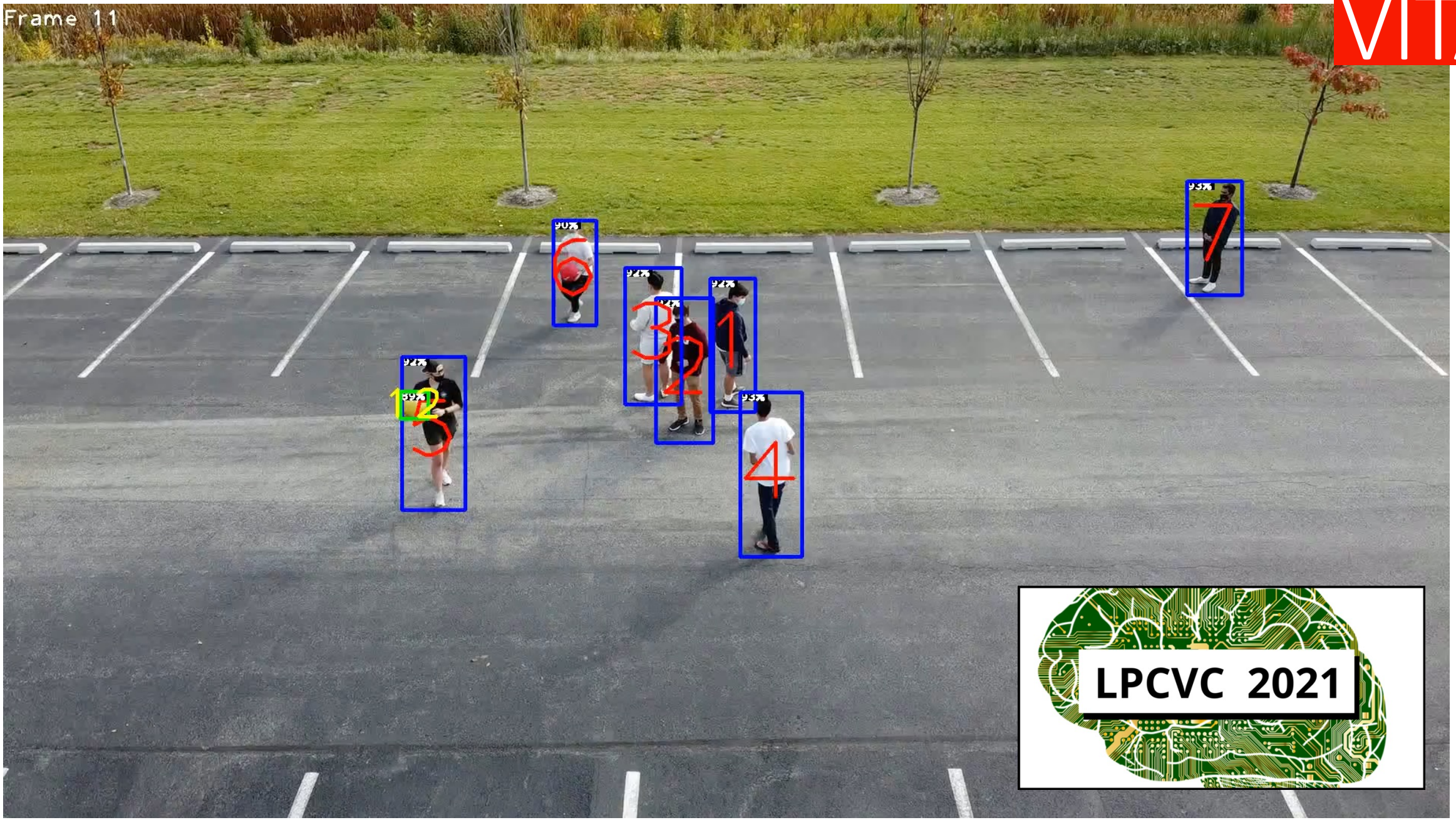
- Jointly utilizing several compression means
  - Also, can choose efficient “by-design” models (MobileNets, or even non-deep models, etc.)
  - Channel pruning is in fact very similar to NAS
- **Data processing** is often a key concern, maybe more important
- **Hardware co-design** is another key concern
- Resource constraints & user demands often **change over time**
- From single task to multi-task and lifelong learning …

Demo: Energy-Efficient UAV-Based Text Spotting System

- **Task:** UAV-based low-energy video understanding (Raspberry Pi 3B+)
- Our group has been leading the show!
  - **2021 IEEE Low-Power Computer Vision (LPCV) Challenge, 1st prize (video track)** among 31 university & company teams that submitted 249 independent solutions
  - **2020 IEEE Low-Power Computer Vision (LPCV) Challenge, 2nd prize (video track**), among ~ 90 solutions






Frame 11
VITA
LPCVC 2021

What is Sparse (in Neural Networks)?

Sparsity in Deep Neural Networks:

Pruning, Lottery Tickets, Sparse Training, Low-Rank, Mixture of Experts…
(Han et al. 16, Frankle et al. 18, Evci et al. 20, Hu et al. 21)

What does Sparsity offer us in neural networks?

Efficiency

- ❑ FLOPs
- ❑ Wall-clock Time
- ❑ Memory
- ❑ Energy, etc.

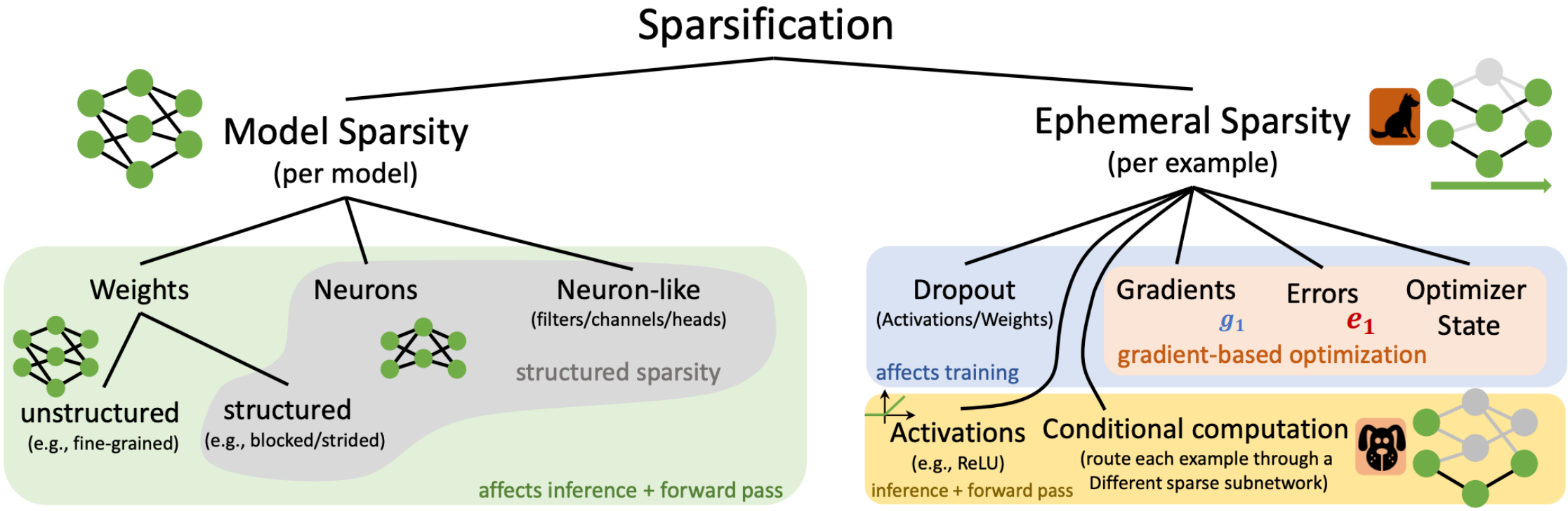
Reliability

- ❑ Cross-domain Generalization
- ❑ Few-shot Learning
- ❑ Robustness, etc.

Recent reference for "overview":

- S. Liu and Z. Wang, "Ten Lessons We Have Learned in the New" Sparseland": A Short Handbook for Sparse Neural Network Researchers", ArXiv 2023

Sparsity in Neural Networks is Versatile!



Source: Hoefler, T., Alistarh, D., Ben-Nun, T., Dryden, N., & Peste, A., "Sparsity in Deep Learning: Pruning and growth for efficient inference and training in neural networks", ***JMLR 2021****.*

Sparsity in Neural Networks is Versatile!

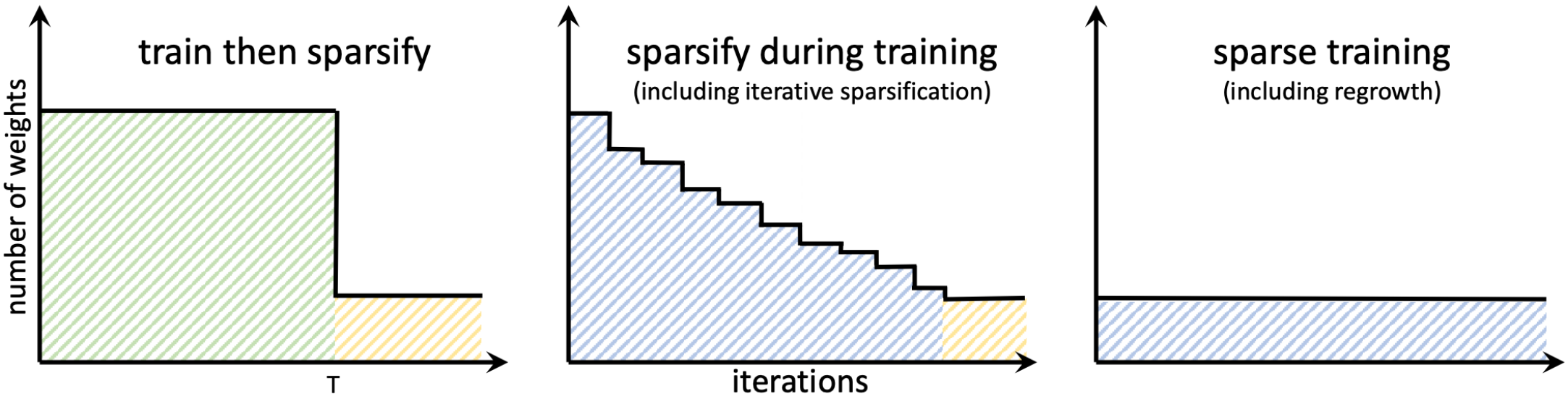
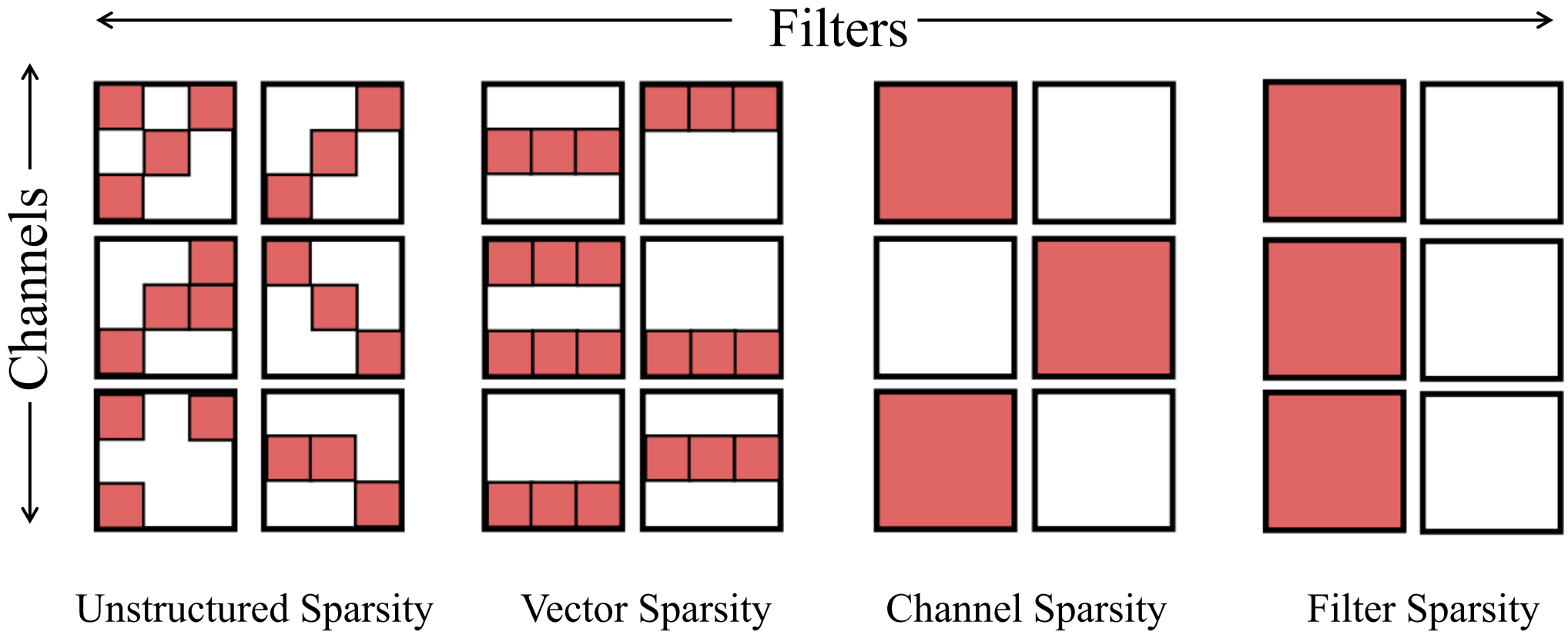


Figure 7: Overview of structural sparsification schedules.

Source: Hoefler, T., Alistarh, D., Ben-Nun, T., Dryden, N., & Peste, A., "Sparsity in Deep Learning: Pruning and growth for efficient inference and training in neural networks", ***JMLR 2021****.*

Sparsity in Neural Networks is Versatile!

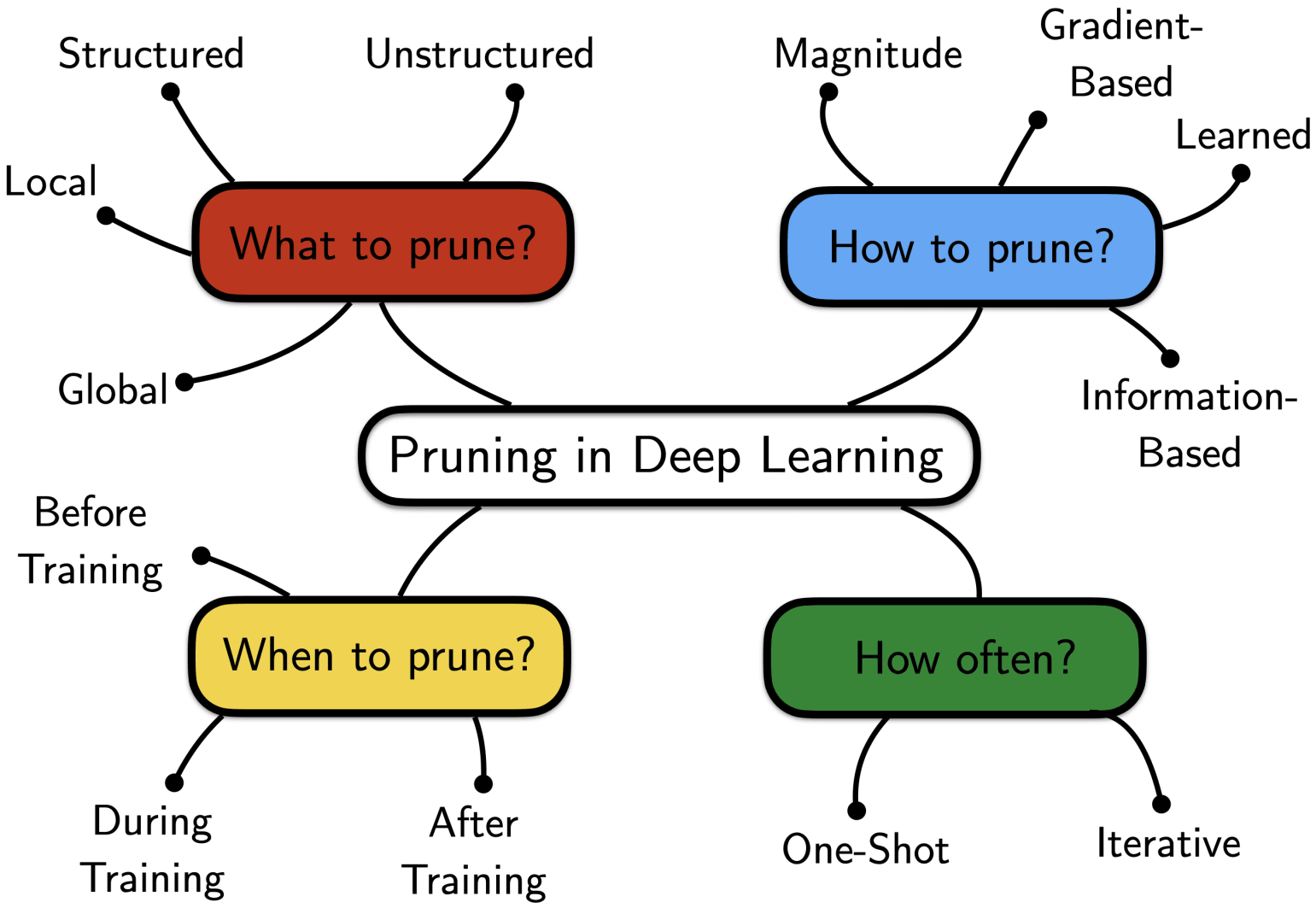


Sparsity in Neural Networks is Versatile!

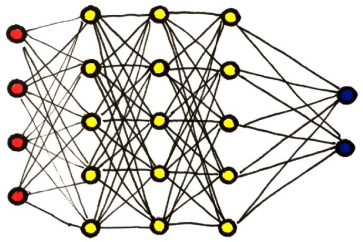
*A finer-grained **overview** plot by **Robert Lange**, TU Berlin*

(only illustrating sparsity in weights, e.g., "pruning")

Source: https://roberttlange.github.io/posts/2020/06/lottery-ticket-hypothesis/

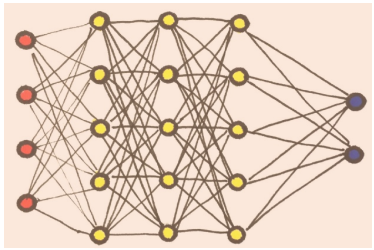


"Old-Fashioned" Sparsity in Deep Neural Networks



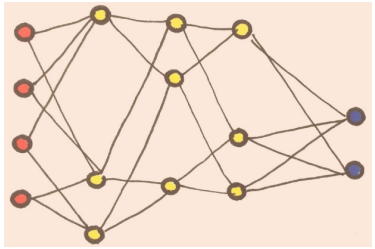
Random NNs

Training



Trained NNs

Pruning

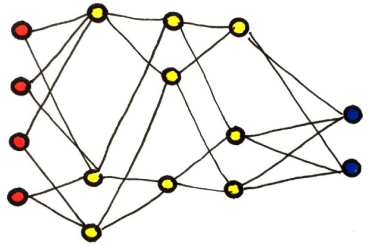


Trained Sparse NNs

Training

Trainable &
Well-performed

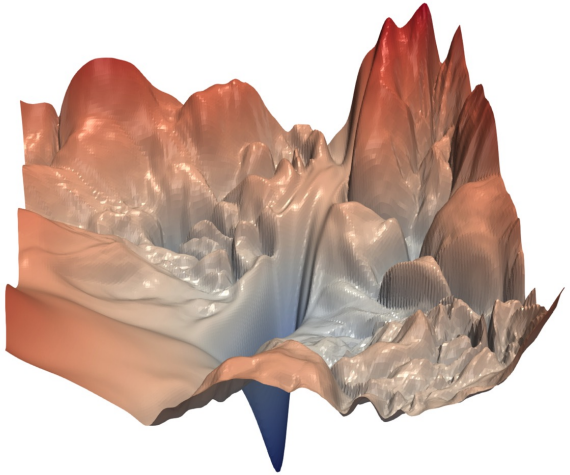
Pruning



Sparse NNs

Training

NOT
Trainable!



01 Han et al., ICLR'15

02 Li et al., ICLR'16

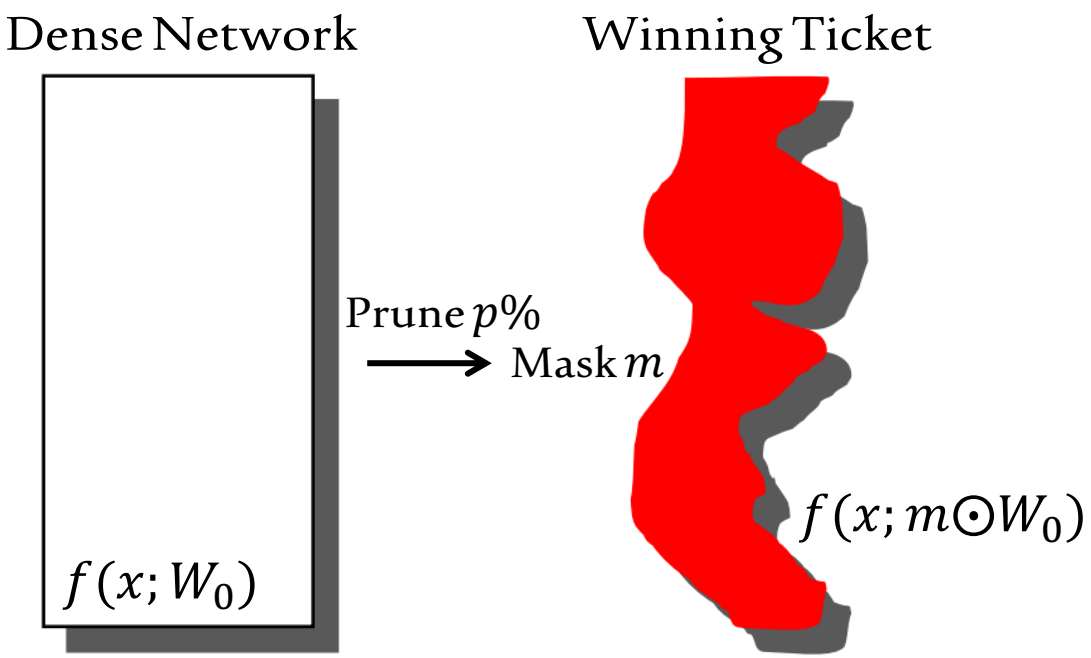
03 Frankle et al., ICLR'19

04 Evci et al., ICML'19

⋮

Newer Solution: Lottery Ticket Hypothesis

A randomly-initialized, dense NN contains a subnetwork that can be separately trained from initialization and match the test accuracy of the original NN after training for at most the same number of iterations. (Frankle et al. ICLR'19)



Win What?

Newer Solution: Lottery Ticket Hypothesis

A randomly-initialized, dense NN contains a subnetwork that can be separately trained from initialization and match the test accuracy of the original NN after training for at most the same number of iterations. (Frankle et al. ICLR'19)

Win What?

03 Trainable from the beginning

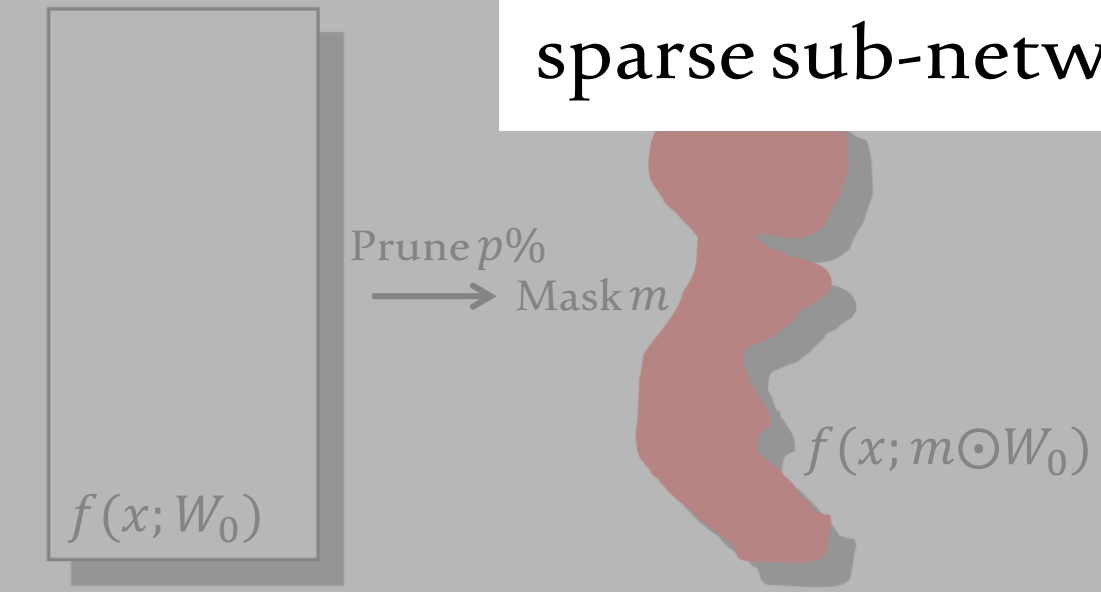
02 Shorter training time & extremely sparse

01 Matched or even better performance

Newer Solution: Lottery Ticket Hypothesis

A randomly-initialized, dense NN contains a subnetwork that can be separately trained from initialization and match the test accuracy of the original NN after training for at most the same number of iterations. (Frankle et al. ICLR'19)

As long as we know which sparse sub-network is winning!



in What?

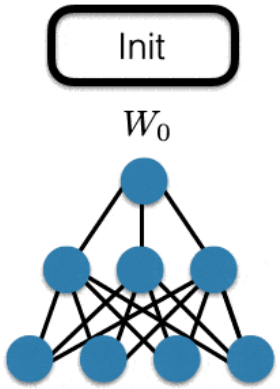
03 Trainable from the beginning

02 Shorter training time & extremely sparse

01 Matched or even better performance

How to Find the Desired Sparse Model?

Iterative Magnitude Pruning



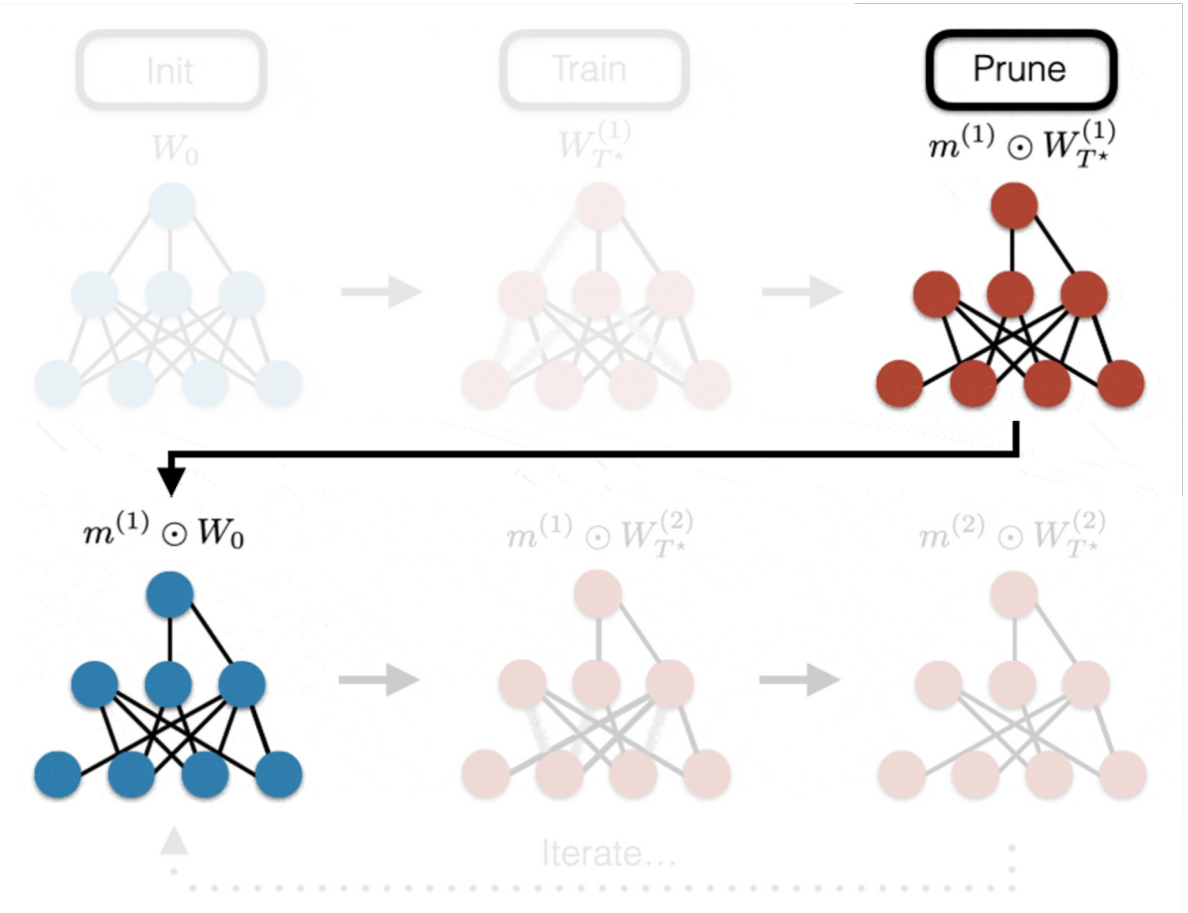
How?

a) Randomly initialize a dense network
b) Train it like normal
c) Prune unimportant weights
d) Reset remaining weight to their values from a) exactly
e) Repeat b-d) iteratively

[Frankle et al. 2019] @RobertTLange

How to Find the Desired Sparse Model?

Iterative Magnitude Pruning



[Frankle et al. 2019] @RobertTLange

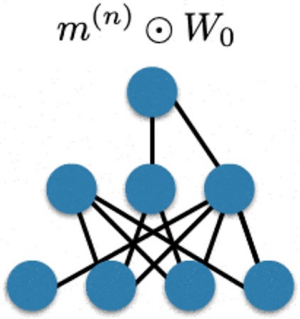
How?

a) Randomly initialize a dense network
b) Train it like normal
c) Prune unimportant weights
d) Reset remaining weight to their values from a) exactly
e) Repeat b-d) iteratively

How to Find the Desired Sparse Model?

The Problem?

Obtaining this good sparse mask is expensive...



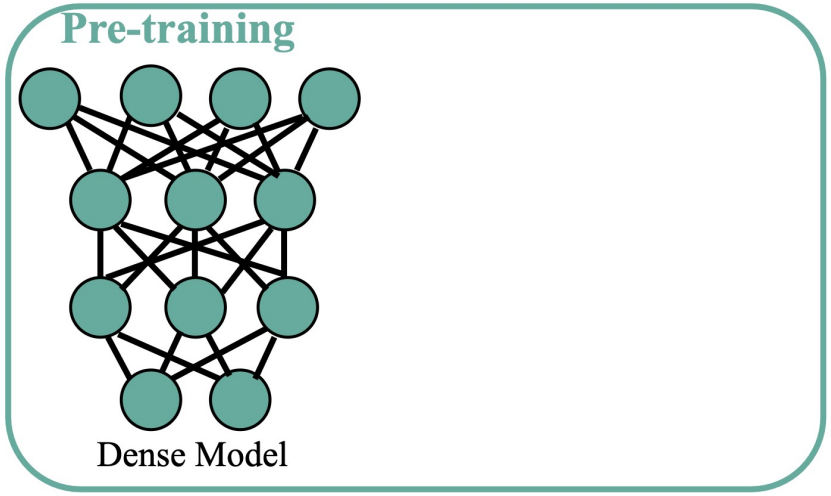
Winning Tickets!

- **Sparse Mask**
- Original Initialization

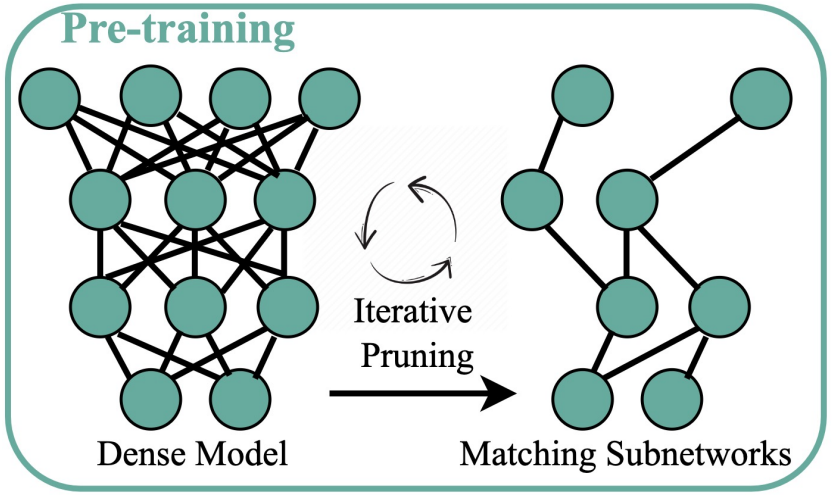
How?

a) Randomly initialize a dense network
b) Train it like normal
c) Prune unimportant weights
d) Reset remaining weight to their values from a) exactly
e) Repeat b-d) iteratively

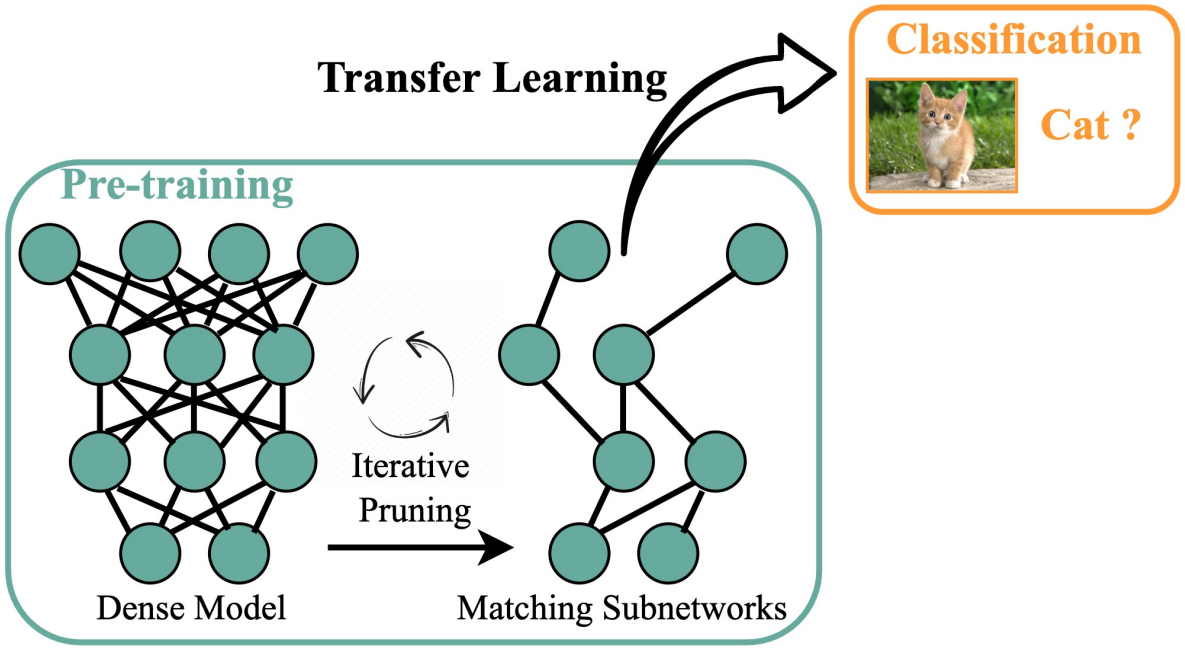
Finding Mask is Expensive? **Re-using a Pre-made Mask!**



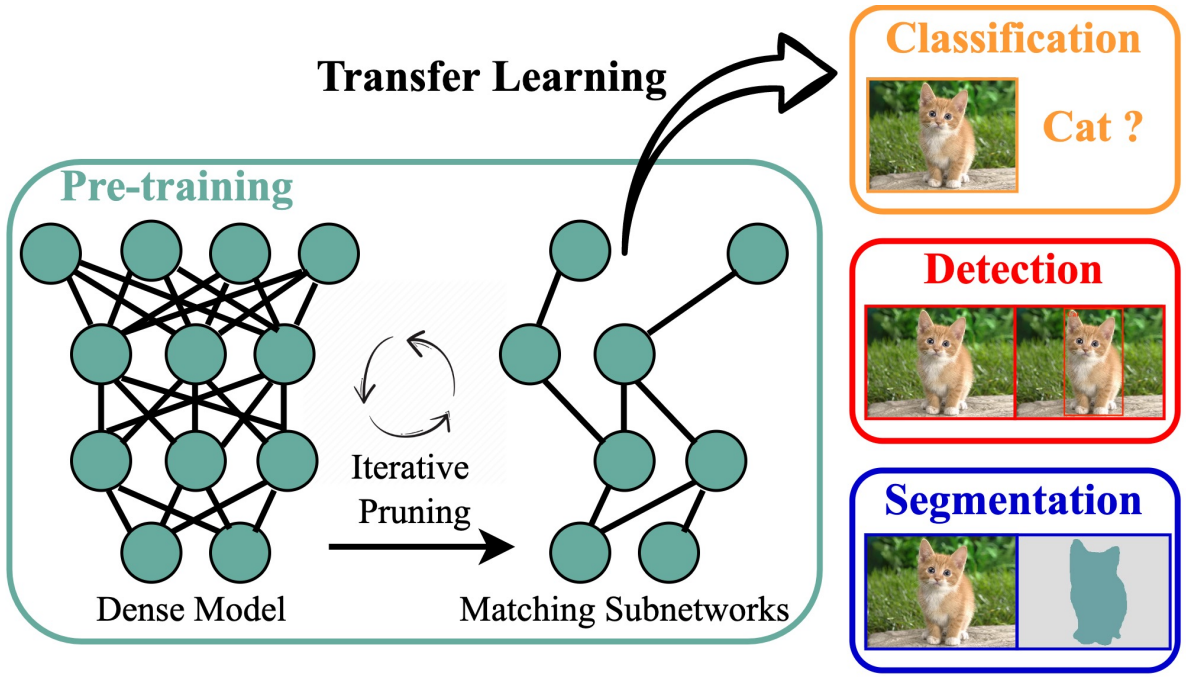
Finding Mask is Expensive? **Re-using a Pre-made Mask!**



Finding Mask is Expensive? **Re-using a Pre-made Mask!**

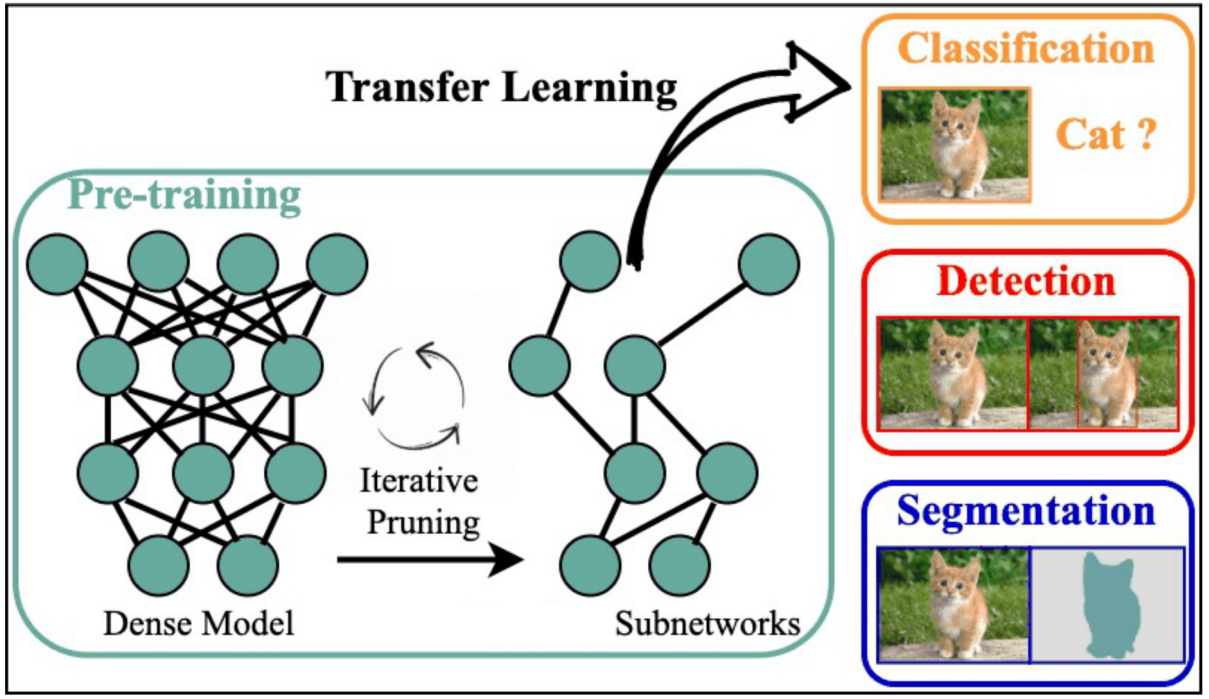


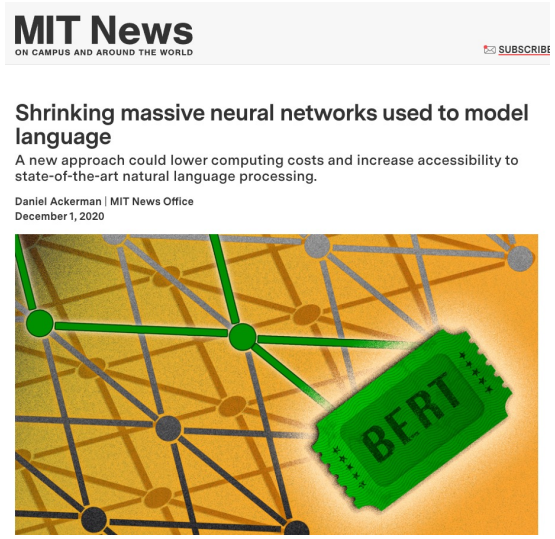
Finding Mask is Expensive? **Re-using** a Pre-made Mask!



Finding Mask is Expensive? Re-using a Pre-made Mask!

Q: Can a "lottery" **universally** transfer to all downstream tasks the same well?
If yes, the extraordinary cost of finding sparse masks can be amortized by **re-using**



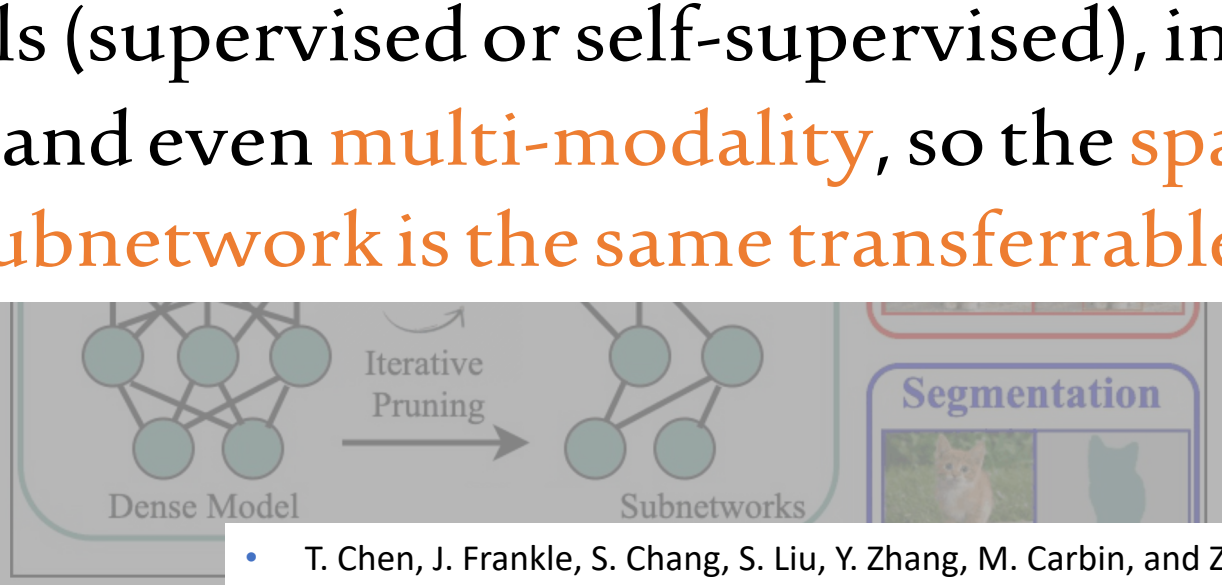


[**Chen** et al. NeurIPS'20, CVPR'21, ICML'22]

Finding Mask is Expensive? Re-using a Pre-made Mask!

Q: Can a "lottery" **universally** transfer to all downstream tasks the same well?
If yes, the extraordinary cost of finding sparse masks can be amortized by **re-using**

IMP can find you a good mask on pre-trained models (supervised or self-supervised), in NLP, CV and even multi-modality, so the sparse subnetwork is the same transferrable!

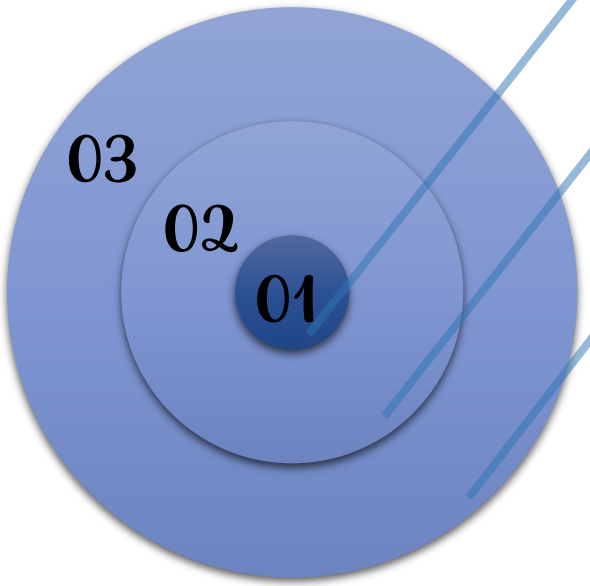


- T. Chen, J. Frankle, S. Chang, S. Liu, Y. Zhang, M. Carbin, and Z. Wang, *"The Lottery Tickets Hypothesis for Supervised and Self-supervised Pre-training in Computer Vision Models"*, **CVPR 2021**
- T. Chen, J. Frankle, S. Chang, S. Liu, Y. Zhang, Z. Wang, and M. Carbin, "*The Lottery Ticket Hypothesis for Pre-trained BERT Networks"*, **NeurIPS 2020.**

A New Promising Regime in the Era of Foundation Models

Instead of **Dense Gigantic** Pre-training

New Regime:



Pre-training a Large Model

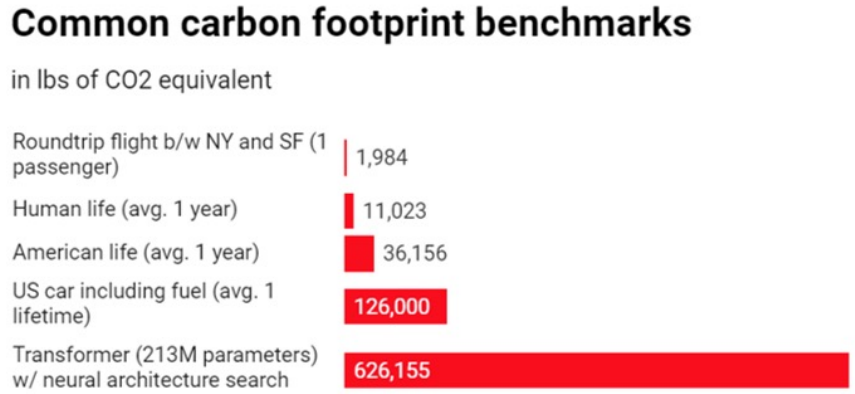
Finding its **Sparse** Lottery

Deploying and **Transferring**

Training Big NNs from scratch is Expensive. Can Sparsity Help?

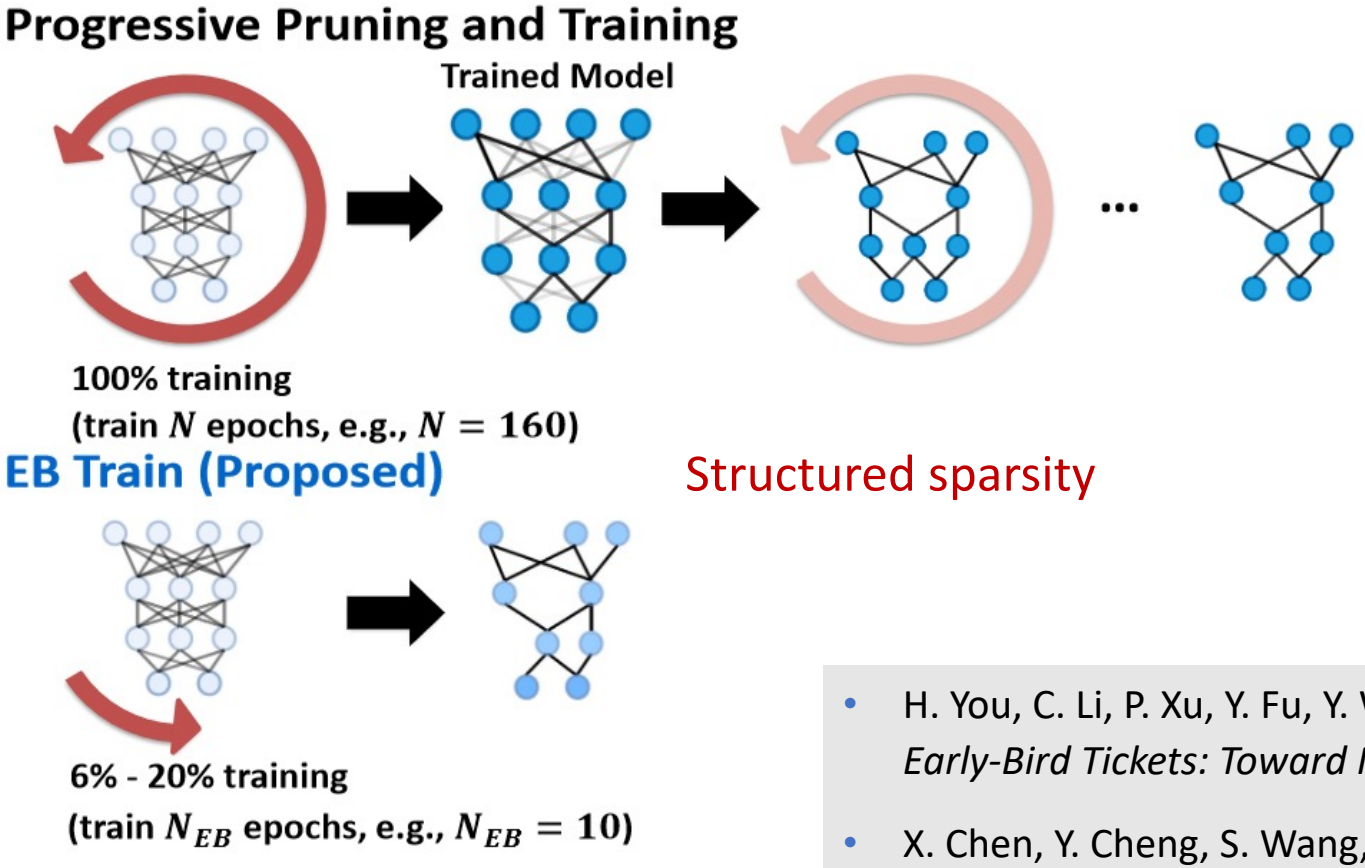
Energy and Policy Considerations for Deep Learning in NLP

Emma Strubell **Ananya Ganesh** **Andrew McCallum**
College of Information and Computer Sciences
University of Massachusetts Amherst
{strubell, aganesh, mccallum}@cs.umass.edu



Model	Hardware	Power (W)	Hours	kWh·PUE	CO_2e	Cloud compute cost
$\text{Transformer}_{base}$	P100x8	1415.78	12	27	26	\$41–\$140
Transformer_{big}	P100x8	1515.43	84	201	192	\$289–\$981
ELMo	P100x3	517.66	336	275	262	\$433–\$1472
BERT_{base}	V100x64	12,041.51	79	1507	1438	\$3751–\$12,571
BERT_{base}	TPUv2x16	—	96	—	—	\$2074–\$6912
NAS	P100x8	1515.43	274,120	656,347	626,155	\$942,973–\$3,201,722
NAS	TPUv2x1	—	32,623	—	—	\$44,055–\$146,848
GPT-2	TPUv3x32	—	168	—	—	\$12,902–\$43,008

Surprise - Good Sparsity Can be Found Early!

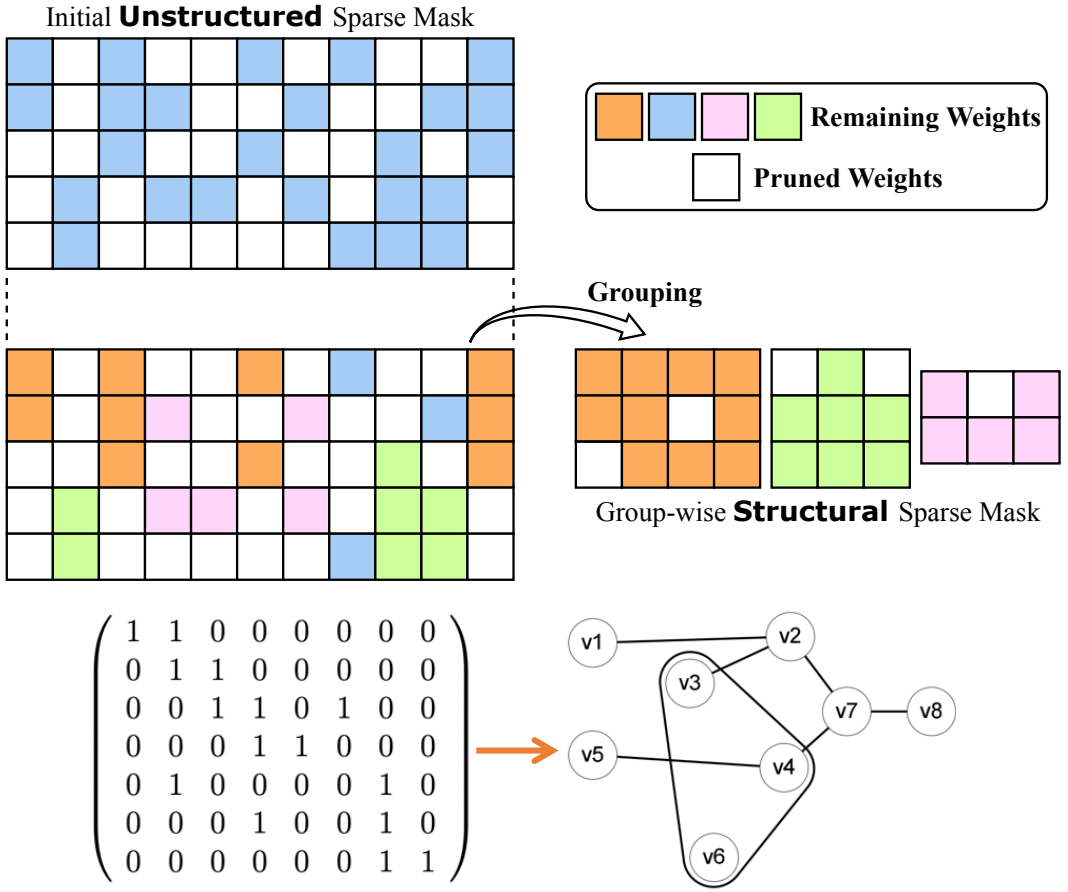


Structured sparsity

- In the first 15-20% of total epochs, the "important" connection subset already emerges and keeps stable!
- **So, pruning can happen early!**
- Training an "early emerging" sparse NN is more efficient not in not only resource, but also data
- Validated on CNNs, BERT & ViTs

- H. You, C. Li, P. Xu, Y. Fu, Y. Wang, X. Chen, R. Baraniuk, Z. Wang, and Y. Lin, "*Drawing Early-Bird Tickets: Toward More Efficient Training of Deep Networks*", **ICLR 2020**
- X. Chen, Y. Cheng, S. Wang, Z. Gan, Z. Wang, and J. Liu, "*EarlyBERT: Efficient BERT Training via Early-Bird Lottery Tickets*", **ACL 2021.**

Lottery Ticket with Hardware Acceleration



60%~65% wall-clock time savings on ImageNet.

Algorithm 3 `IMP-Regroup`

Input: $f(x; m \odot \theta_i)$ with unstructured sparsity s from Algorithm 1, hyperparameters t_1, t_2, b_1, and b_2

Output: $f(x; m \odot \theta_i)$ with group-wise structural mask m at sparsity s^*

while dense block can be found **do**
  Divide the rows of the sparse pruning mask m into t_1 groups using hypergraph partitioning (hMETIS)[a]
  for group $c_i \in \{c_1, c_2, \ldots, c_{t_1}\}$ **do**
    if c_i has $\geq b_1$ rows **then**
      Select columns in c_i that has no less than t_2 non-zero items
      if $\geq b_2$ columns are selected **then**
        Group and Refill the selected columns as well as rows to a dense block, and update m
      end if
    end if
  end for
end while
Set other elements out of dense blocks to 0

T. Chen, X. Chen, X. Ma, Y. Wang, and Z. Wang, "*Coarsening the Granularity: Towards Structurally Sparse Lottery Tickets*", **ICML 2022**

From One Mask to Many: **Dynamic Sparse Training (DST)**

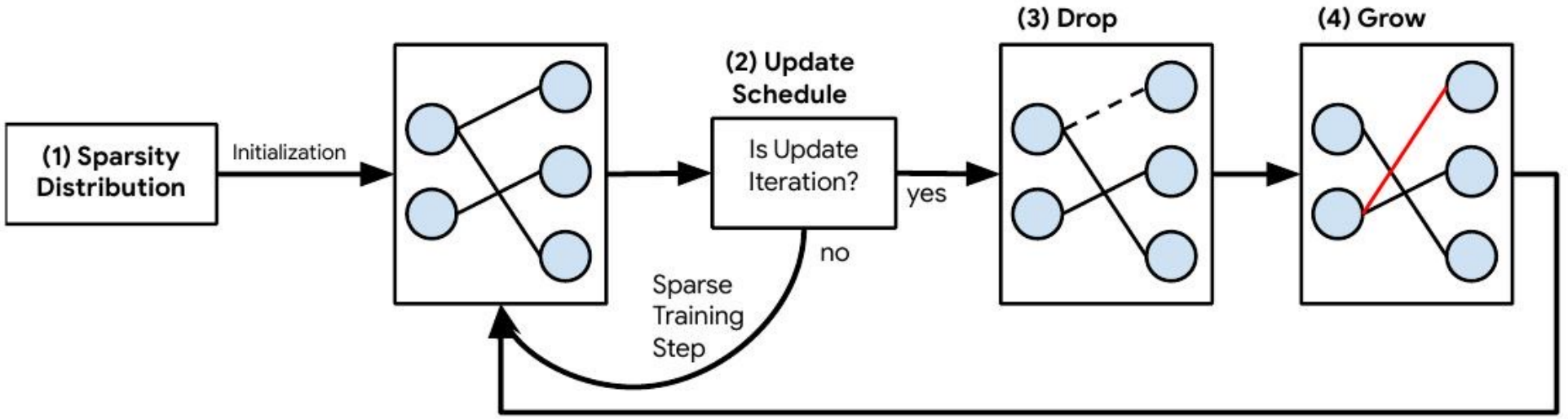
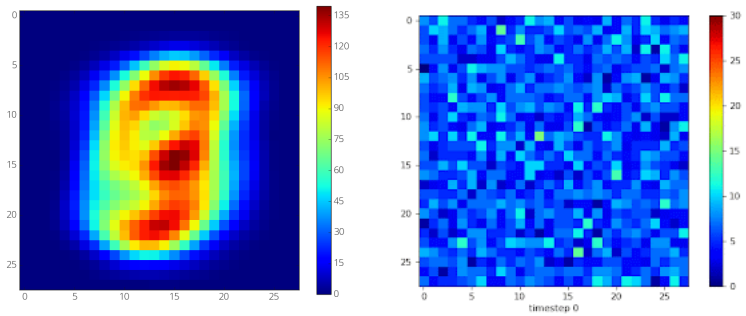


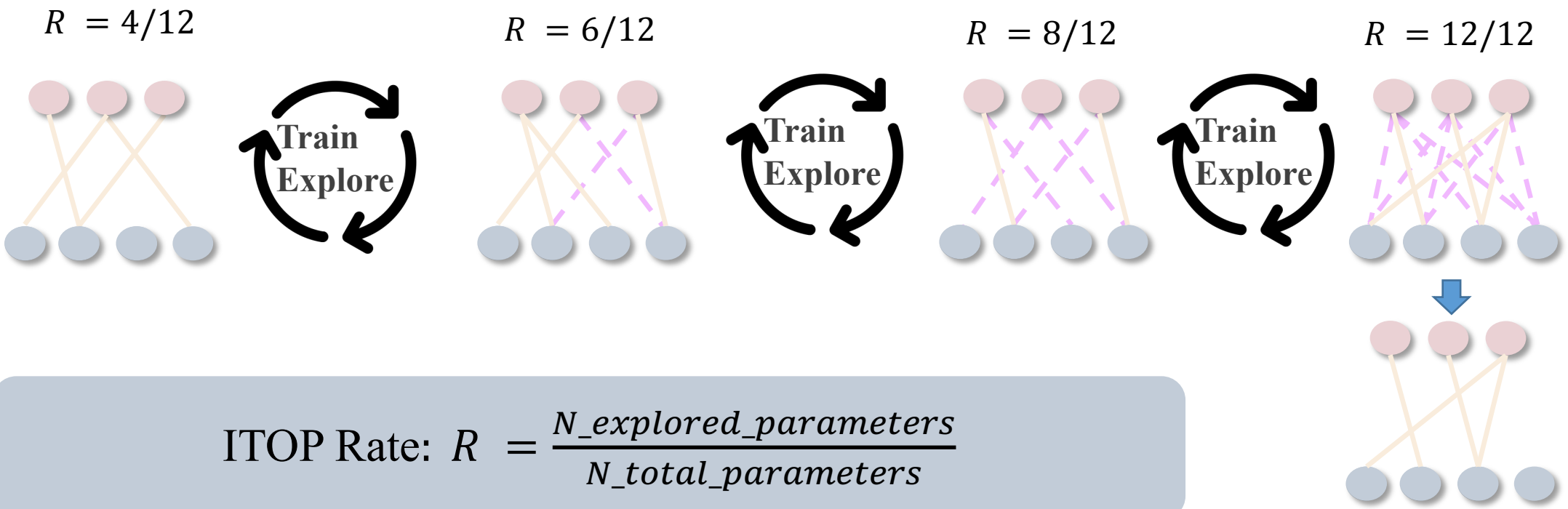
Figure 1: Dynamic sparse training changes connectivity during training to aid optimization.



Evci, Utku, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. "Rigging the lottery: Making all tickets winners." ***ICML 2020***

Source:
https://ai.googleblog.com/2020/09/improving-sparse-training-with-rigl.html

In-Time Over-Parameterization (ITOP): **Exploration** is the Key



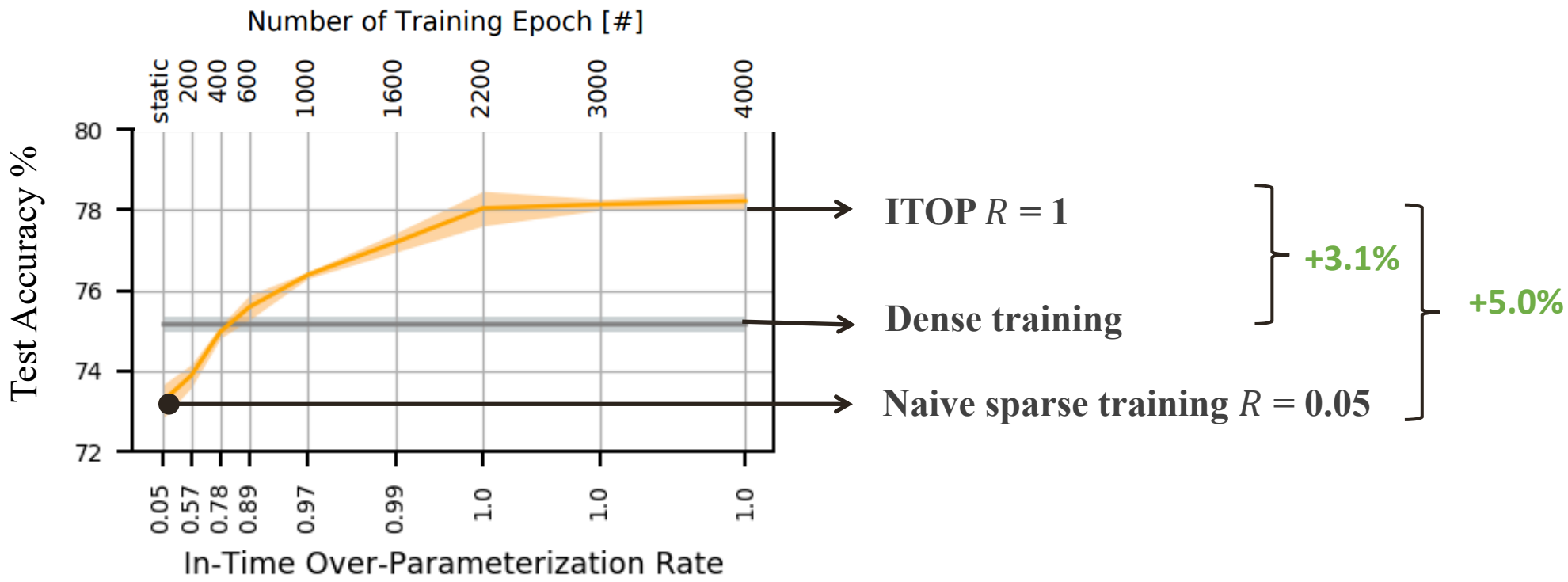
ITOP Rate: $R = \frac{N\_explored\_parameters}{N\_total\_parameters}$

Final Sparse Network

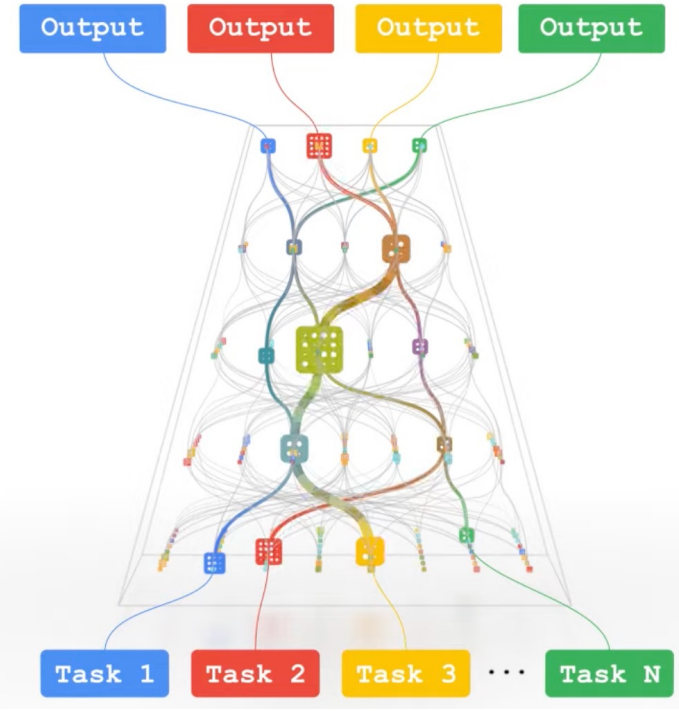
S. Liu, L. Yin, D. Mocanu, and M. Pechenizkiy. "*Do we actually need dense over-parameterization? in-time over-parameterization in sparse training*". **ICML 2021**

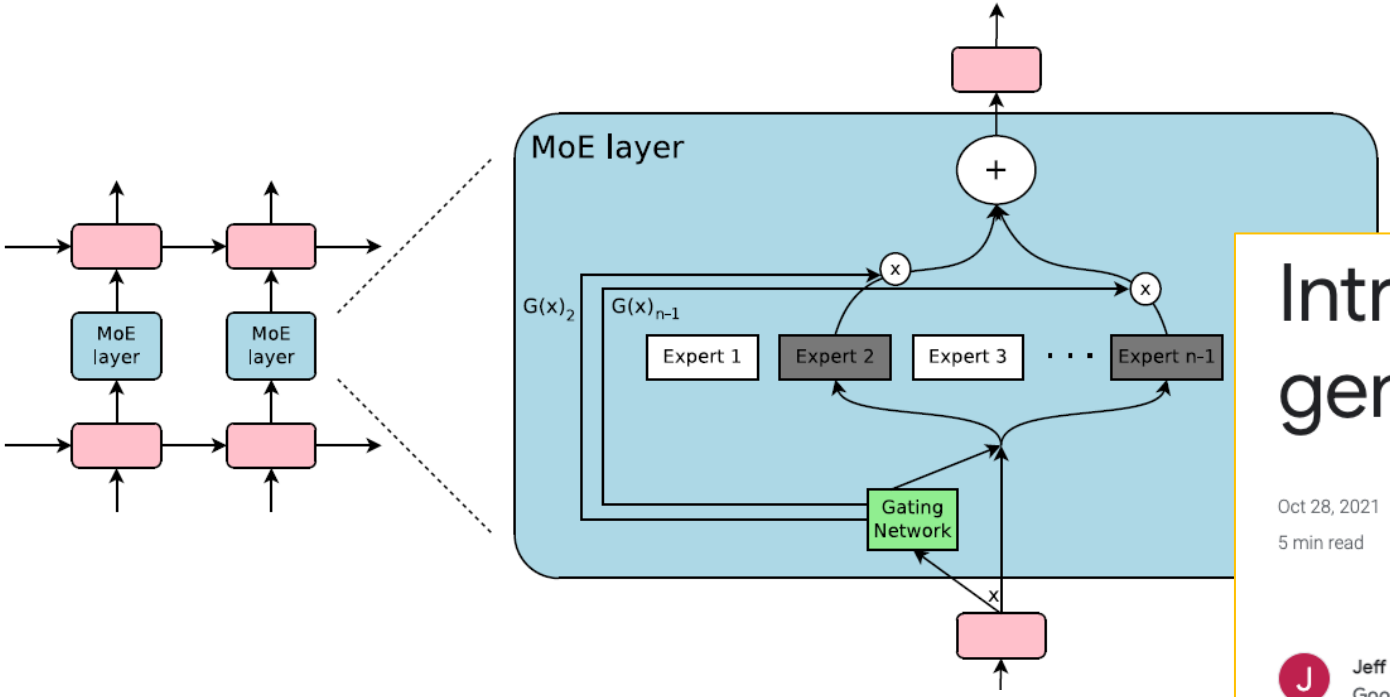
ITOP Hypothesis: The performance of sparse training is highly correlated with the proportion of weights that have been explored.

Evaluation: 95% sparse ResNet-34 on CIFAR-100



Efficiently Scaling with Dynamic Sparsity:
Mixture of Experts (MoEs)





Shazeer M. et. al. “Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts

Introducing Pathways: A next-generation AI architecture

Oct 28, 2021
5 min read

Too often, machine learning systems overspecialize at individual tasks, when they could e
we’re building Pathways—a new AI architecture that will handle many tasks at once, learn
reflect a better understanding of the world.

Jeff Dean
Google Senior Fellow and SVP, Google Research

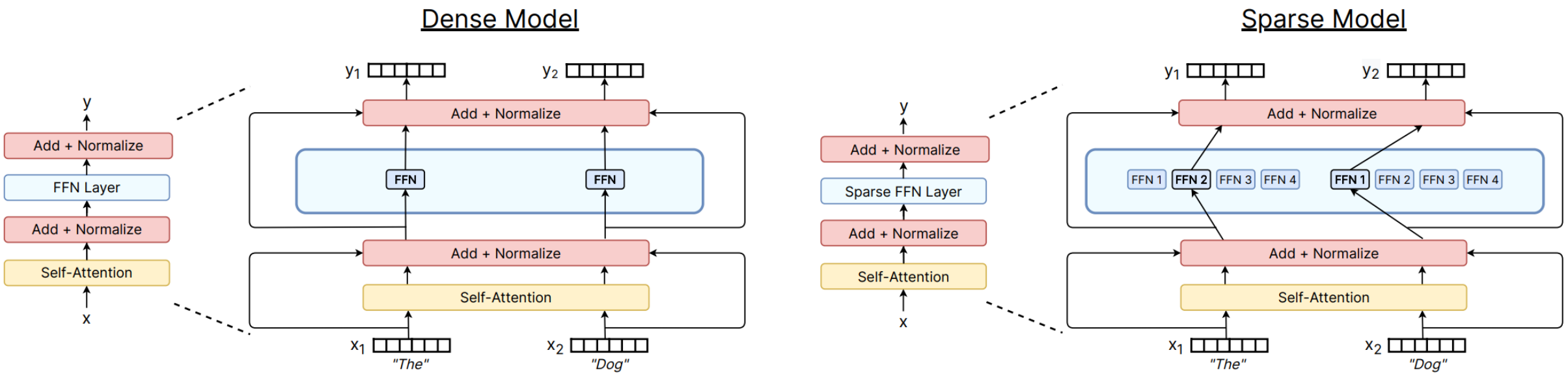
Why MoE?

- MoE is a special type of sparsity (**dynamic, structured, end-to-end**)
  - "Modalized" structure is naturally good for distributed training/parallelism
  - "Block-level" sparsity is hardware-friendly
  - "End-to-end" sparsity keeps the memory /compute low at any point of training

- MoE is also a special type of dynamic inference
  - Dynamically activate an "input-dependent" subnetwork for a new test sample
  - The activation is controlled by a **routing network** (top-*k* classifier, RL, hashing…)

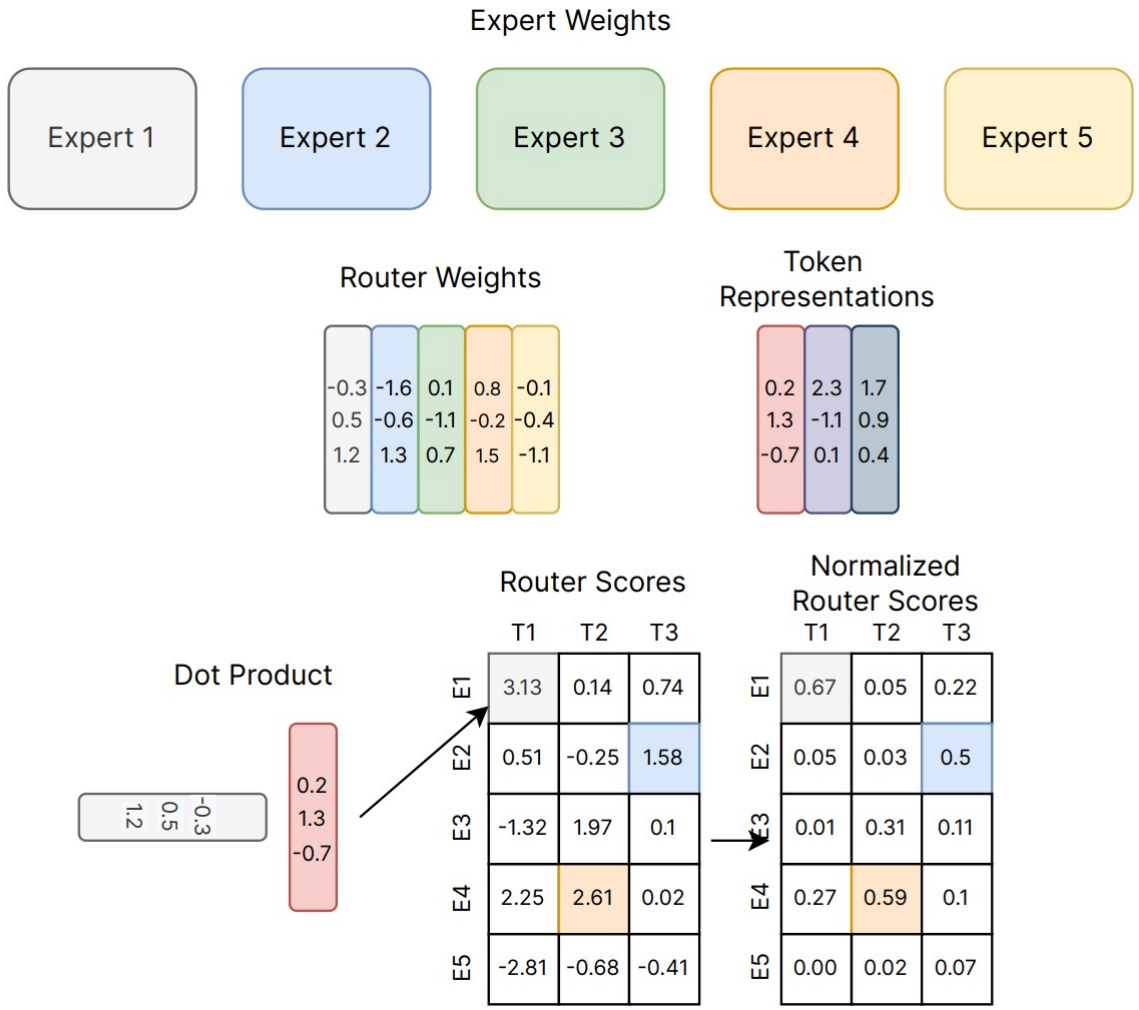
- MoE can be straightforwardly extended to "divide and conquer"…
  - Multi-task learning
  - Multi-modality learning

Dense versus Sparse MoE Transformer



Fedus, William, Jeff Dean, and Barret Zoph. "A review of sparse expert models in deep learning." *arXiv preprint arXiv:2209.01667* (2022).

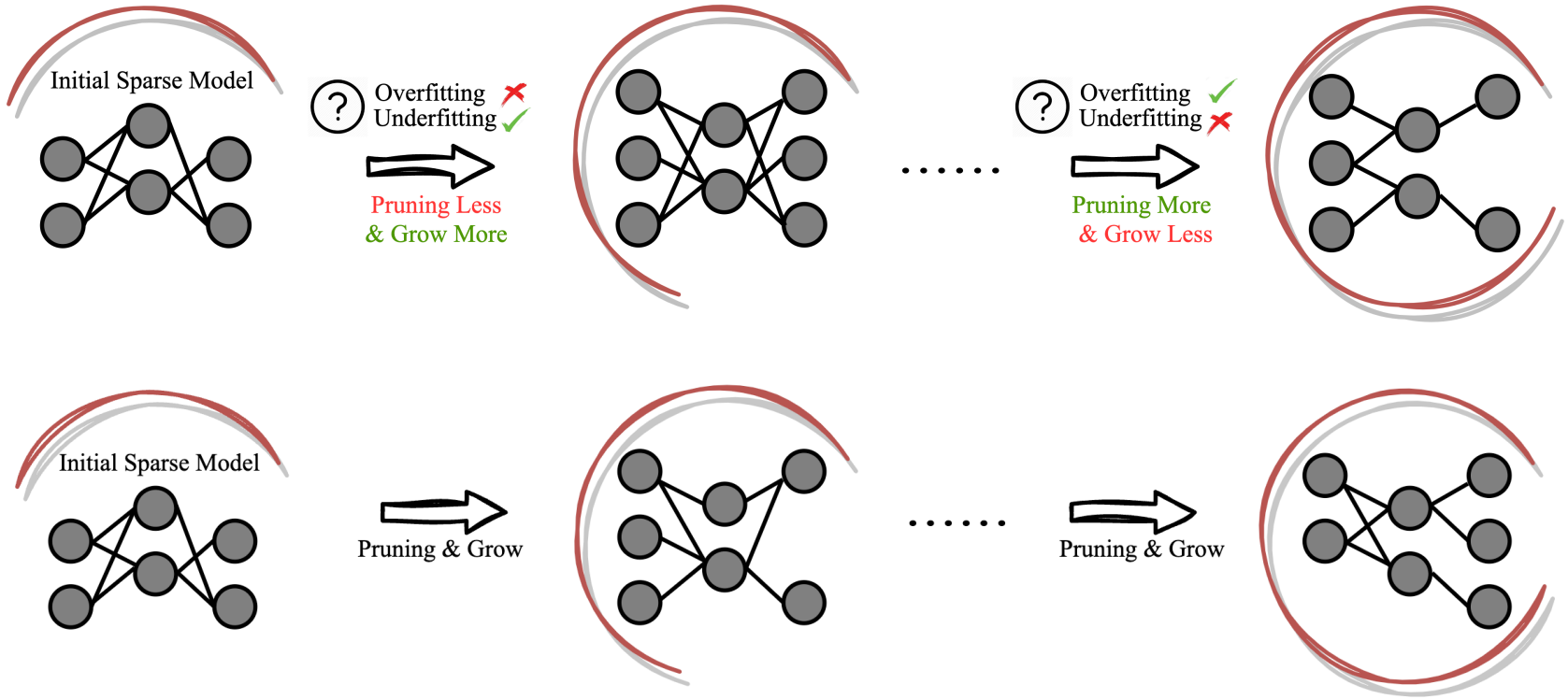
Schematic of Routing Network (using top-k as example)

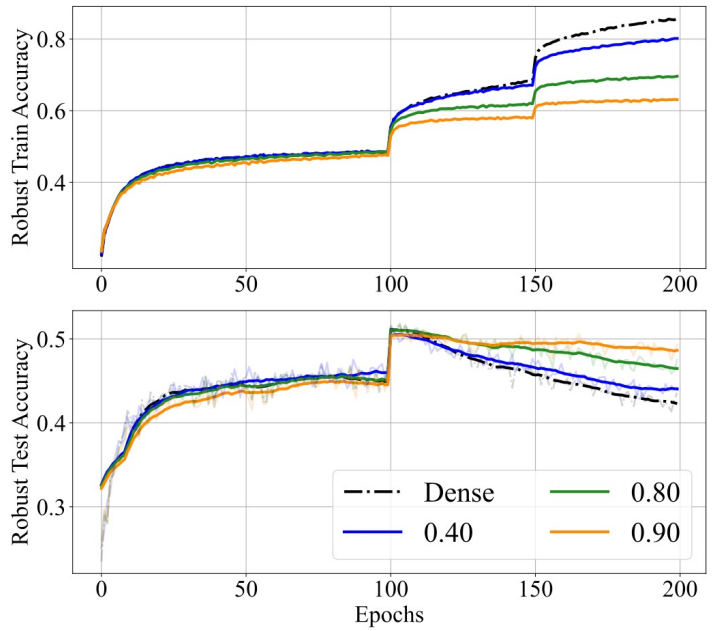


Many open challenges remain on routing!

- Expert load balancing
- Representational Collapse
- "In-situ" change sparsity k?
- .....

Sparsity Helps Robustness? Adversarial/Backdoor/Certified





- T. Chen, Z. Zhang, P. Wang, S. Balachandra, H. Ma, Z. Wang, and Z. Wang, *"Sparsity Winning Twice: Better Robust Generalization from More Efficient Training"*, **ICLR 2022.**

Empirically All Appealing … But Any Theory?

Sparsity meets Deep Learning Theory: A Developing Frontier!

1. Approximation: e.g., whether there exists a sparse NN that approximates its dense NN arbitrarily well ("stronger LTH", etc.)

- E. Malach et. al., "Proving the Lottery Ticket Hypothesis: Pruning is All You Need", **ICML 2020 (just one example, more…)**

2. Optimization: e.g., whether a sparse NN train slower than its dense NN, and if yes how much slower. Also, why sparsity often naturally emerges in NN training?

- H. Yang and Z. Wang, "On the Neural Tangent Kernel Analysis of Randomly Pruned Wide Neural Networks", **AISTATS 2023**

3. Generalization: e.g., does sparse NN generalize better than its dense NN in-distribution, over distribution shifts, or with label noise

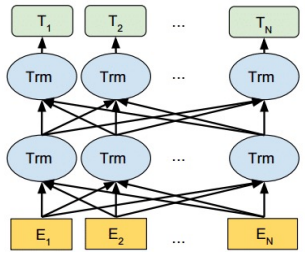
- H. Yang, …, Z. Wang, "Theoretical Characterization of How Neural Network Pruning Affects its Generalization", **arXiv 2023**

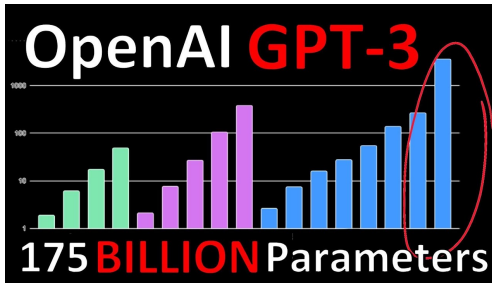
4. Topology (weight-agnostic, often graph theory): e.g., what differentiates between a good sparse mask and a bad one? How to characterize masks changing over time?

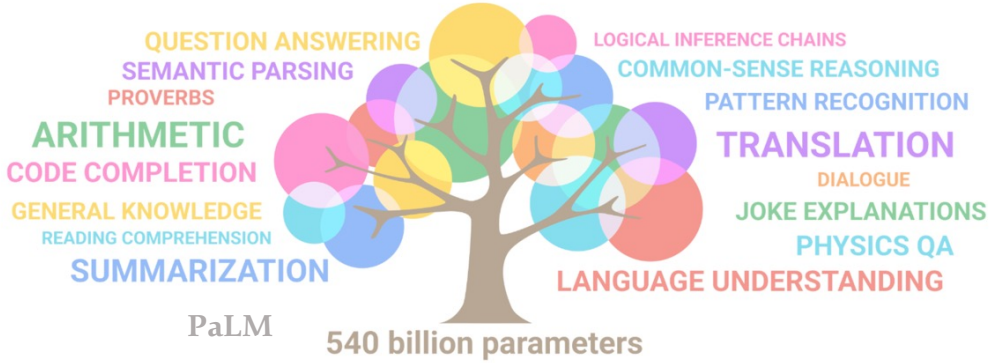
- D. Hoang, …, Z. Wang, "Revisiting Pruning at Initialization Through the Lens of Ramanujan Graph", **ICLR 2023**

Sparse Transfer: An NLP Example

Efficiency

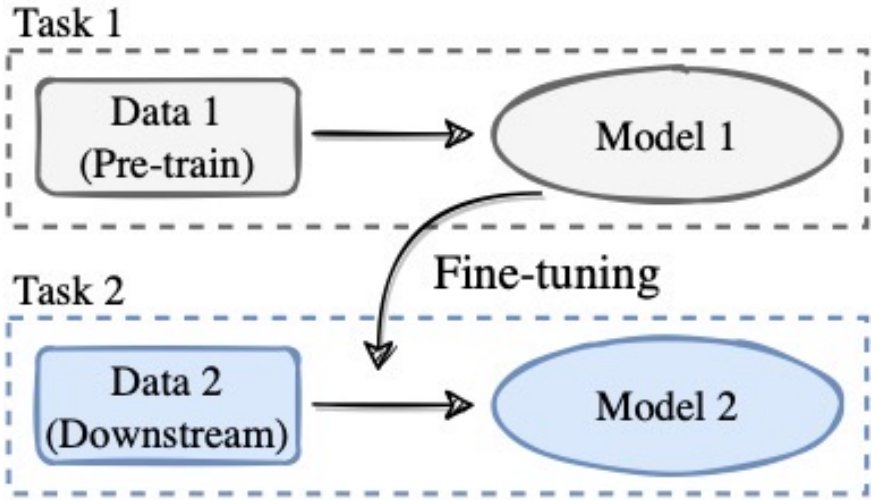






Reliability

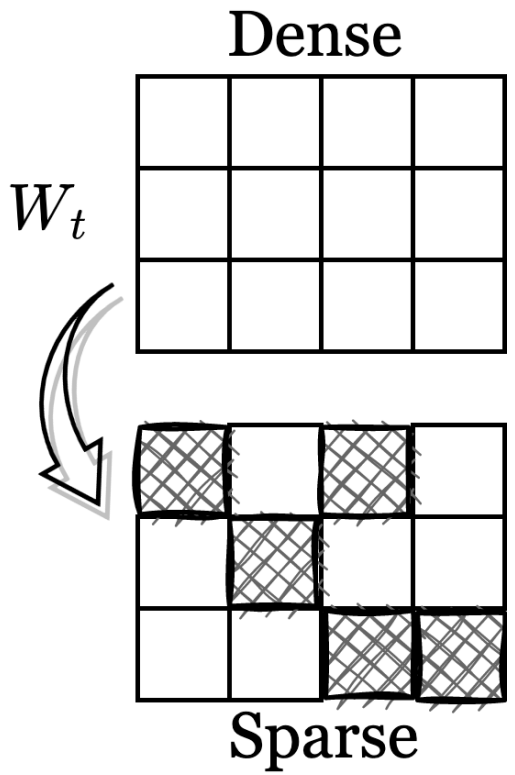
- Task Discrepancy
- Data Discrepancy
- Data Scarcity in Downstream

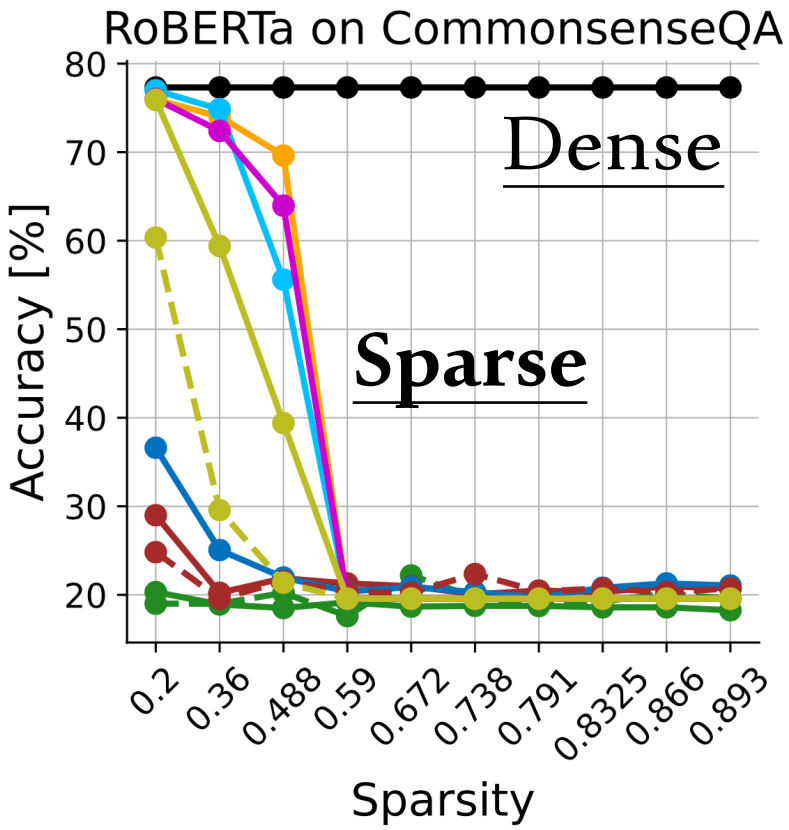


Sparse Transfer: An NLP Example

Sparsification

Weights $W_t = W_{t-1} + \Delta W$



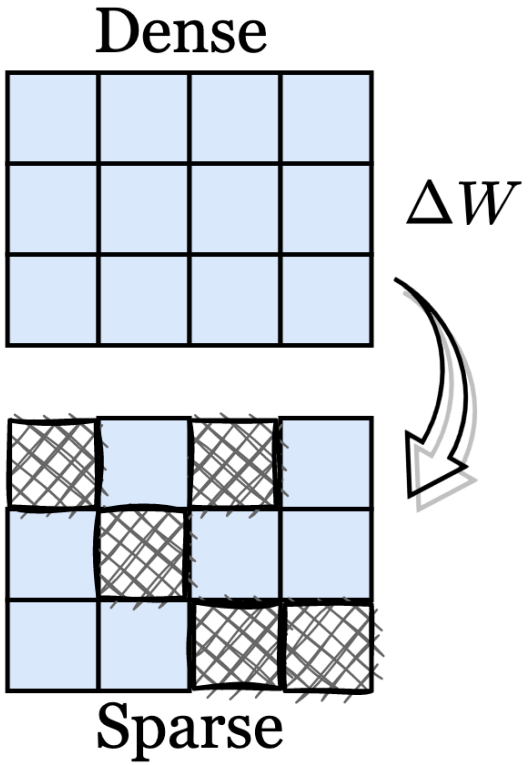


Sparse Transfer: An NLP Example

Sparsification

$$W_t = W_{t-1} + \Delta W$$ Weight Updates

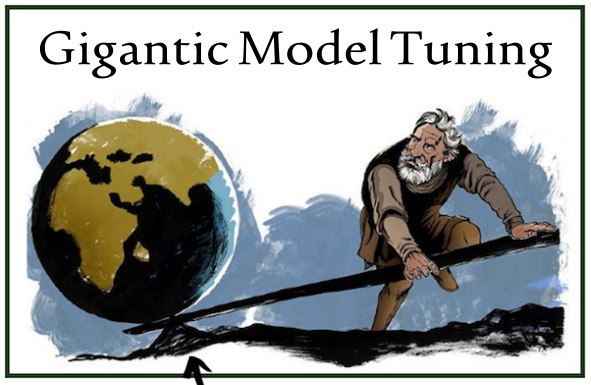
"Diff Pruning"
[Guo et al. ACL'21]



Sparse Transfer: An NLP Example

Sparsification

$$W_t = W_{t-1} + \Delta W$$ Weight Updates

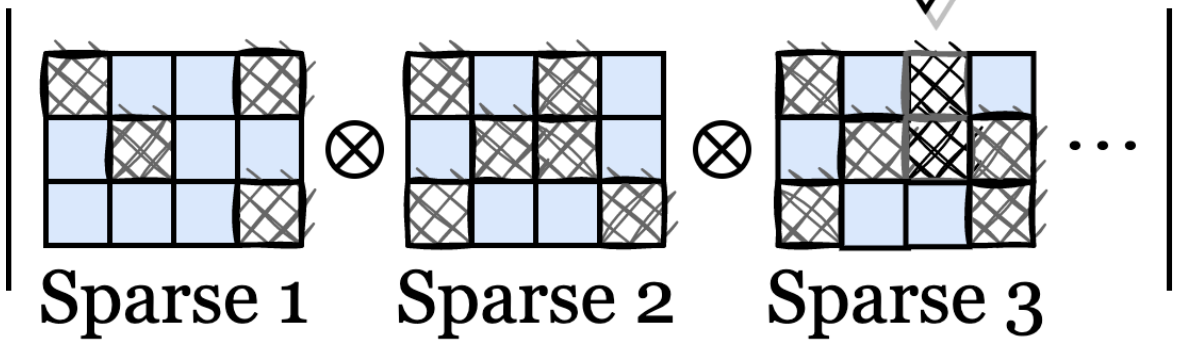


Sparsity

Dense

ΔW

Ours



The Multiplication Results are **Dense!**

Sparse Transfer: An NLP Example

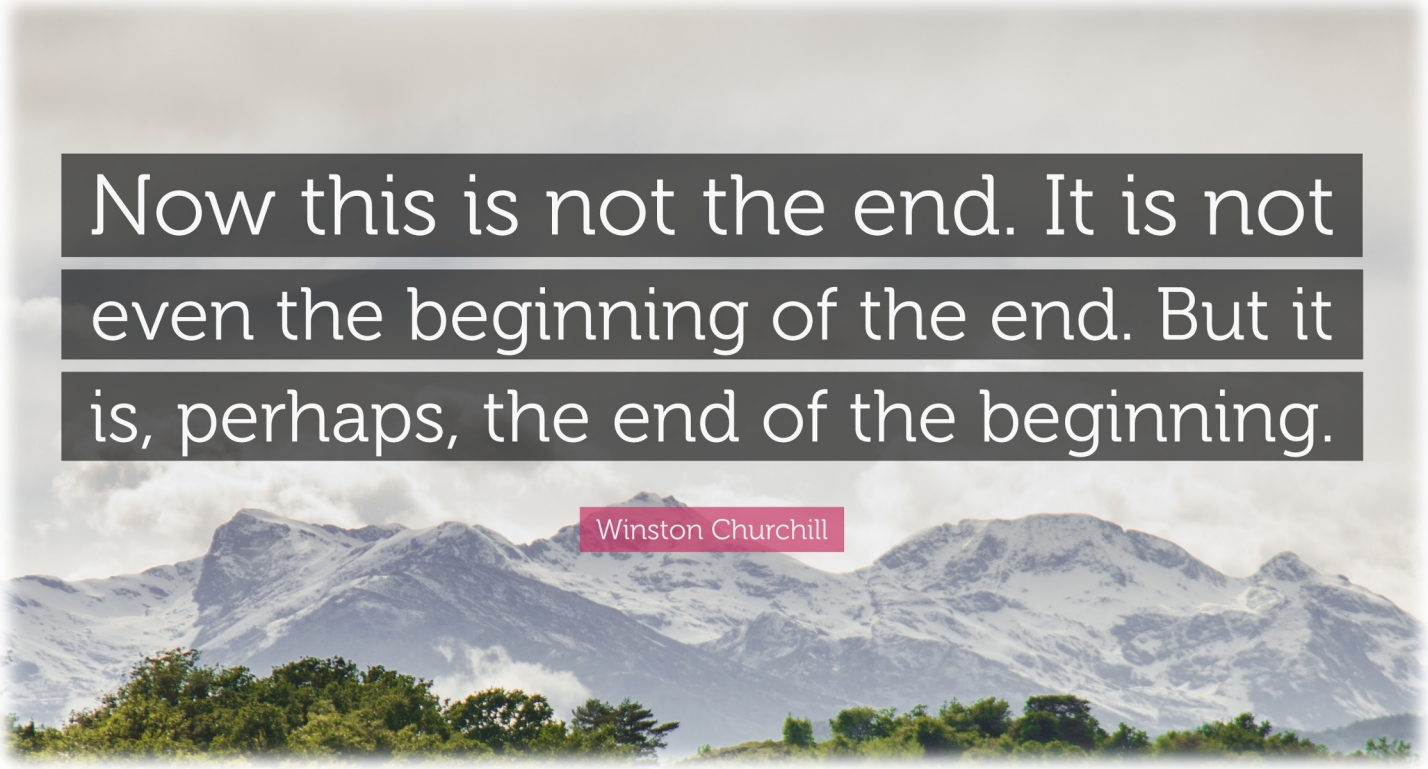
Sparsification

$$W_t = W_{t-1} + \Delta W$$ Weight Updates

GPT-2	#Trainable Parameters	BLEU	NIST
Full Fine-tune	354.92 M	68.2	8.62
Adapters	11.48 M	68.9	8.71
Ours	**0.39M**	**69.4**	**8.78**

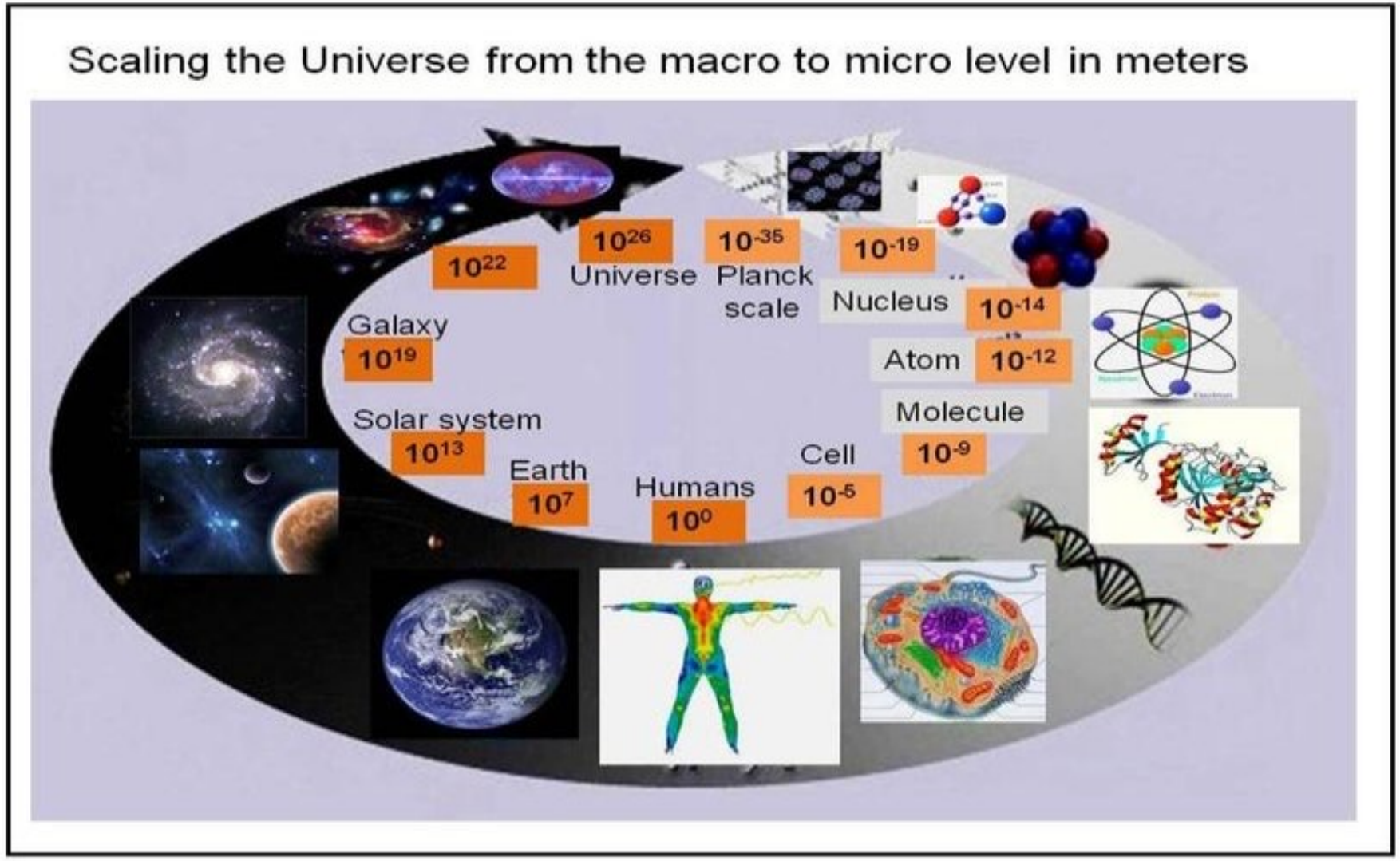
- X. Chen, T. Chen, Y. Cheng, W. Chen, Z. Wang, and A. Awadallah "*DSEE: Dually Sparsity-embedded Efficient Tuning of Pre-trained Language Models*?", ACL 2023

A Look Into the Future of Sparsity?



- Algorithm
- Theory
- Application
- System
- (Neuroscience)

A Look Into the Future of Sparsity?



In the "ML model" universe, sparsity will have key roles at both extreme ends:

- **"Trimming down"**
- **"Scaling up"**

The University of Texas at Austin
Electrical and Computer Engineering
Cockrell School of Engineering