Papers about

Lottery Ticket Hypothesis

Yufang Liu

East China Normal University

Background

- LTH
  - suggests the existence of highly trainable sparse networks at random initialization – winning ticket
  - init -> train -> prune -> rewind -> ...
- Why ?
  - why does rewinding help ? (rewind to a later point)
  - what does model learn at ealy training phase ?
  - when and why does model fail to find winning ticket ?

THE EARLY PHASE OF NEURAL NETWORK TRAINING

Jonathan Frankle[†]
MIT CSAIL

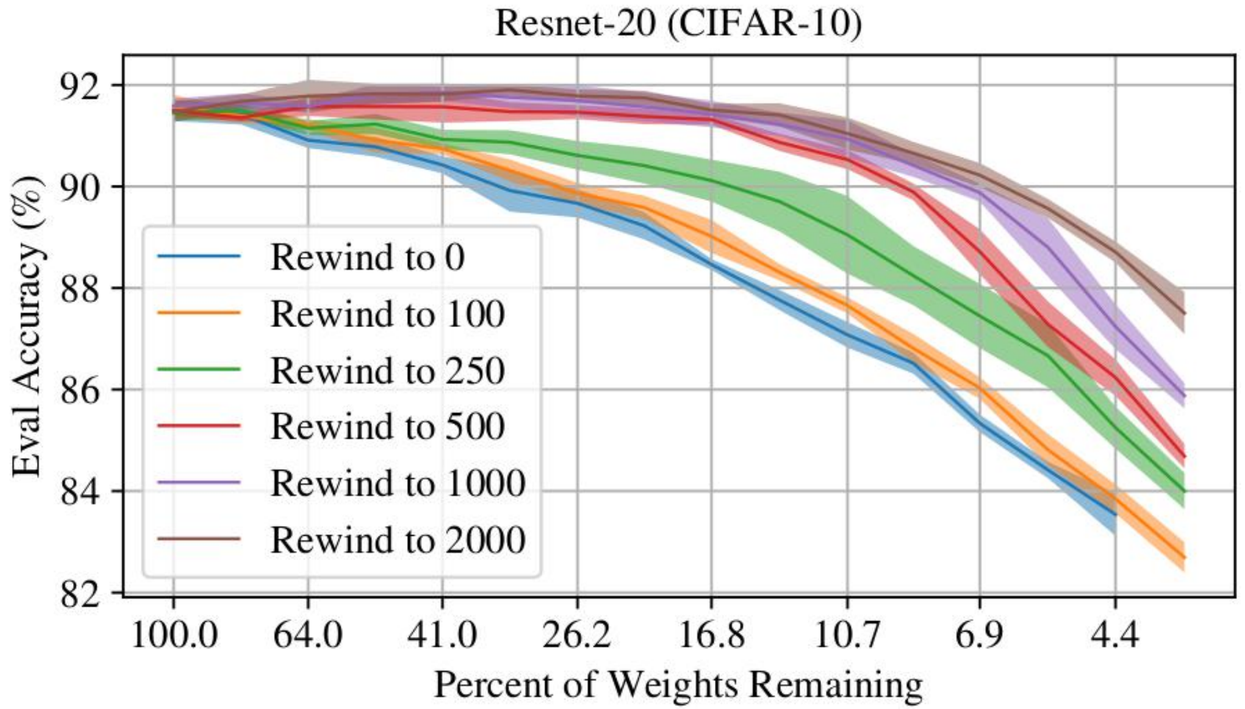
David J. Schwab
CUNY ITS
Facebook AI Research

Ari S. Morcos
Facebook AI Research

ICLR 2020

Background

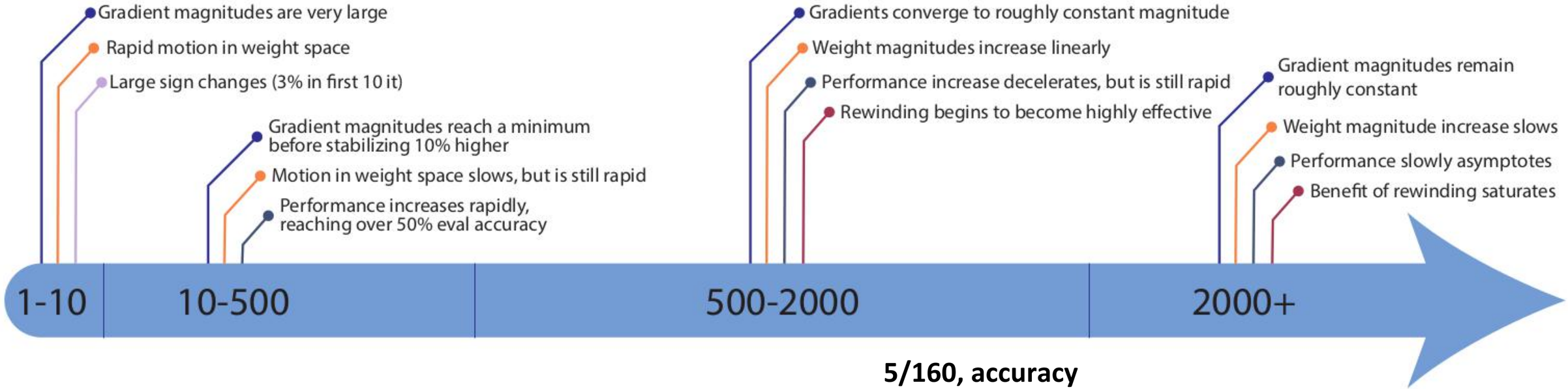
- Lottery ticket rewinding
  - the benefit of rewinding saturates quickly, roughly between 500 and 2000
- Hessian eigenspectrum
  - at init, many large positive and negative eigenvalues
  - a few large eigenvalues emerge, the bulk eigenvalues are close to zero, and the negative eigenvalues become very small.



k > 2000, no further improvement observed

Methods

- Networks
  - mainly on resnet-20 for cifar10
- IMP with rewinding
  - extract sub-networks from various points in training



5/160, accuracy approaches 80%

Methods

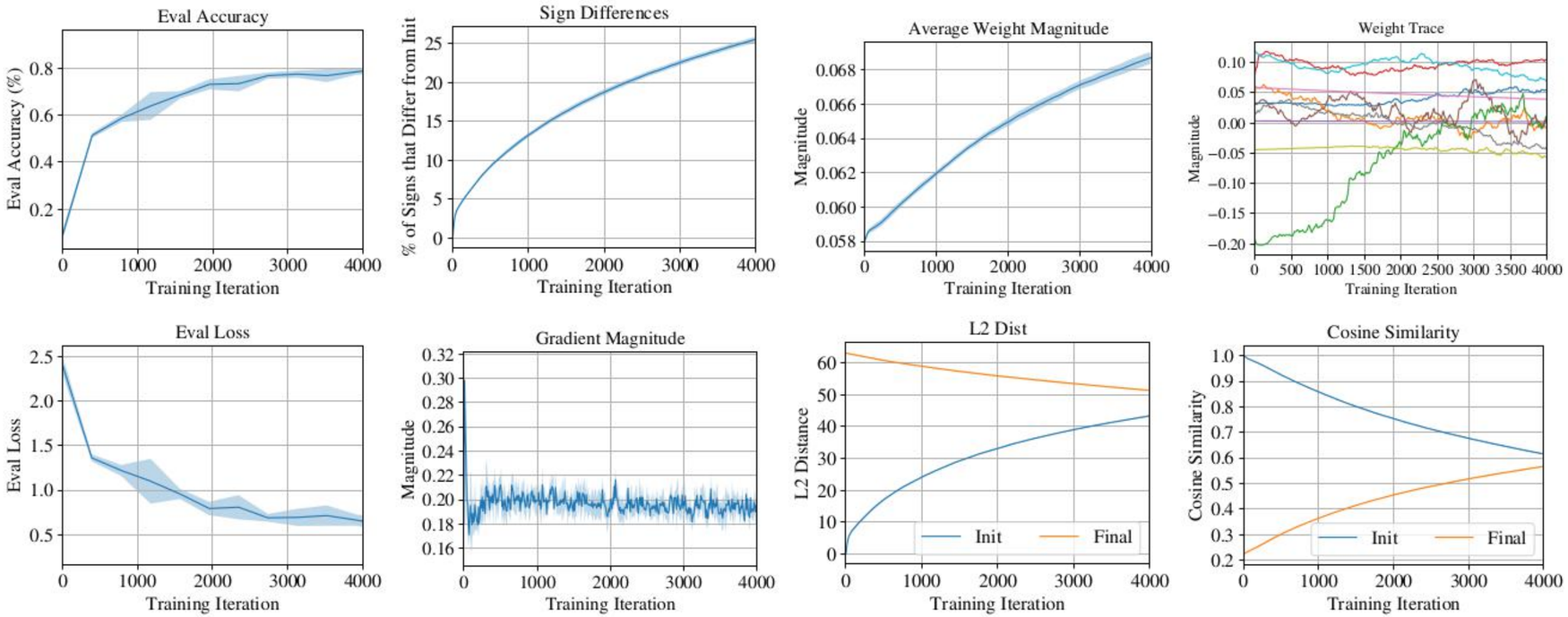


Figure 3: Basic telemetry about the state of ResNet-20 during the first 4000 iterations (10 epochs). Top row: evaluation accuracy/loss; average weight magnitude; percentage of weights that change sign from initialization; the values of ten randomly-selected weights. Bottom row: gradient magnitude; L2 distance of weights from their initial values and final values at the end of training; cosine similarity of weights from their initial values and final values at the end of training.

Perturbing neural networks early in training

- Are Signs all you need ?

perturbation to the network: purple < red

network signs undergo important changes (9%)

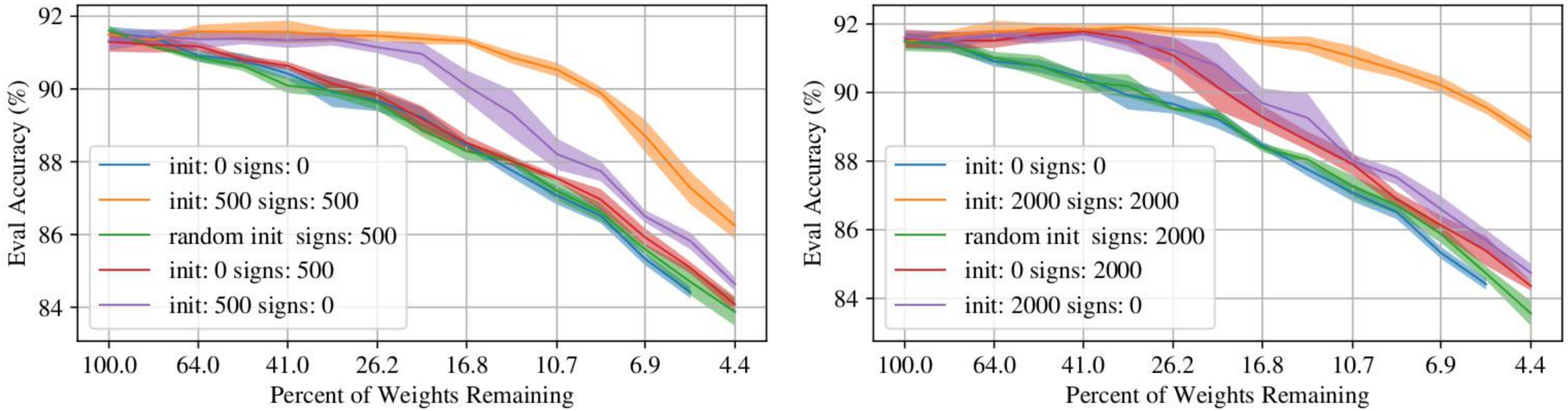


Figure 4: Performance of an IMP-derived sub-network of ResNet-20 on CIFAR-10 initialized to the signs at iteration 0 or k and the magnitudes at iteration 0 or k. Left: $k = 500$. Right: $k = 2000$.

Perturbing neural networks early in training

- Are Weight Distribution I.I.D. ?

perturbation to the network: purple < red, green

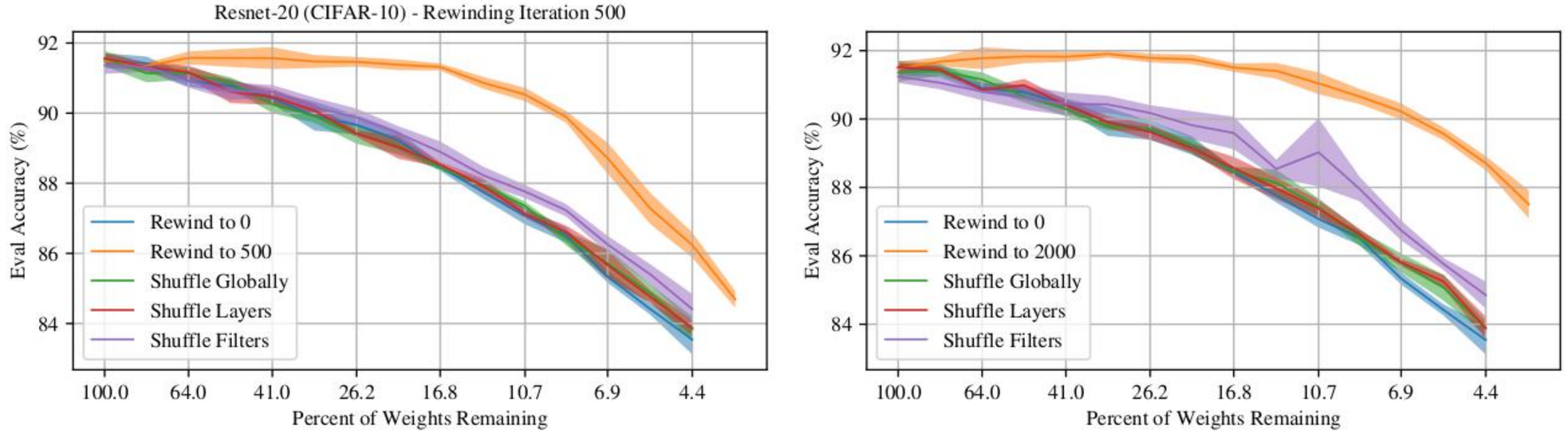


Figure 5: Performance of an IMP-derived ResNet-20 sub-network on CIFAR-10 initialized with the weights at iteration k permuted within various structural elements. Left: $k = 500$. Right: $k = 2000$.

Perturbing neural networks early in training

- Are Weight Distribution I.I.D. ?

signs helps recover the damage, but smaller perturbation

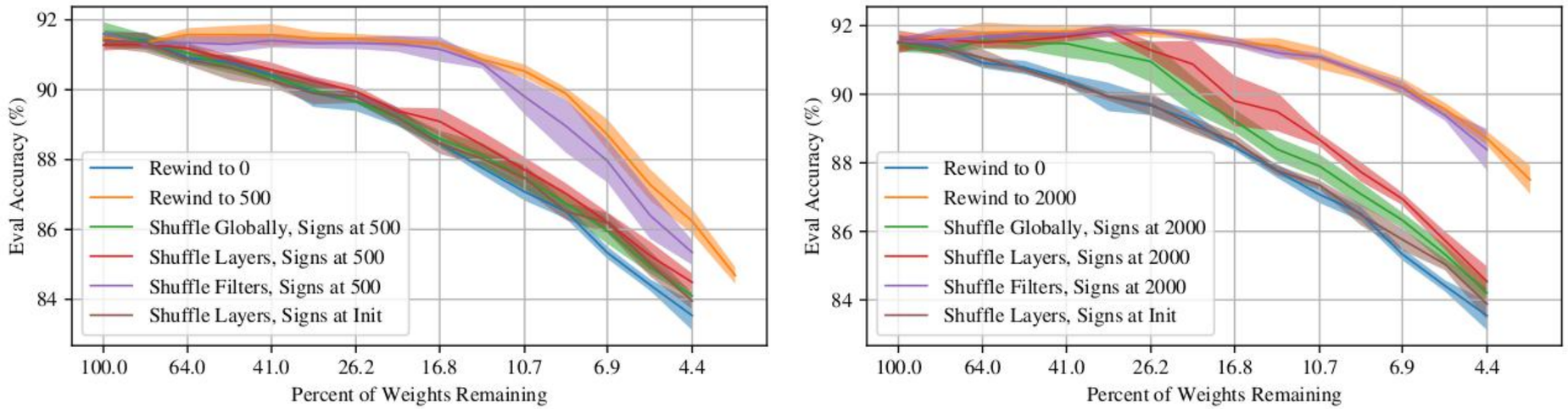


Figure 6: The effect of training an IMP-derived sub-network of ResNet-20 on CIFAR-10 initialized with the weights at iteration k as shuffled within various structural elements where shuffling only occurs between weights with the same sign. Left: $k = 500$. Right: $k = 2000$.

weight distribution after the early phase of training is highly non-independent.

Perturbing neural networks early in training

- Is It All Just Noise ?

tolerate 0.5σ to 1σ of noise

tolerate even 2σ of noise

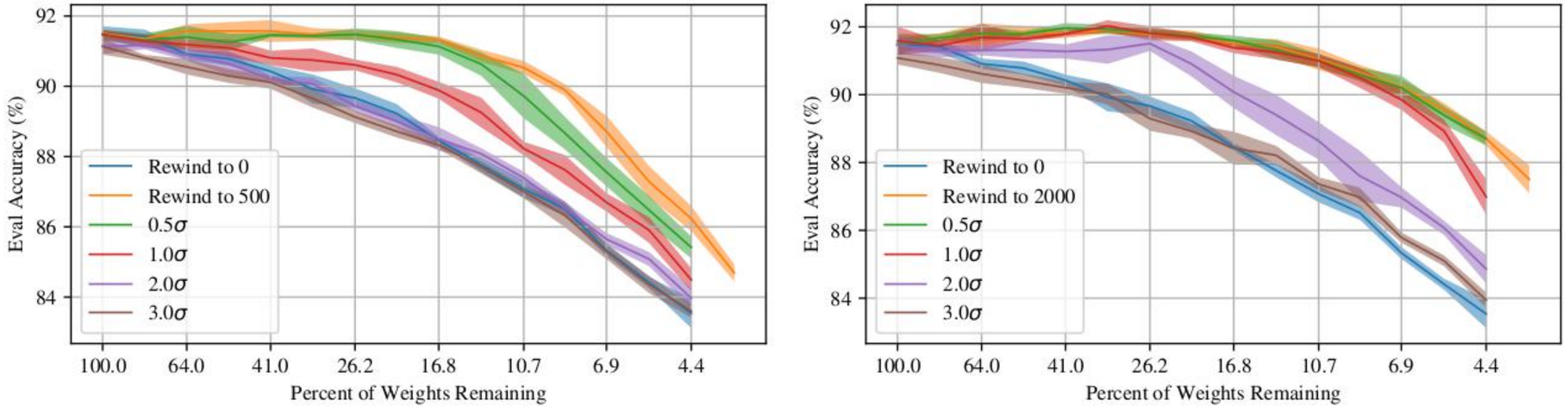


Figure 7: The effect of training an IMP-derived sub-network of ResNet-20 on CIFAR-10 initialized with the weights at iteration k and Gaussian noise of $n\sigma$, where σ is the standard deviation of the initialization distribution for each layer. Left: $k = 500$. Right: $k = 2000$.

Perturbing neural networks early in training

green points aligned least well

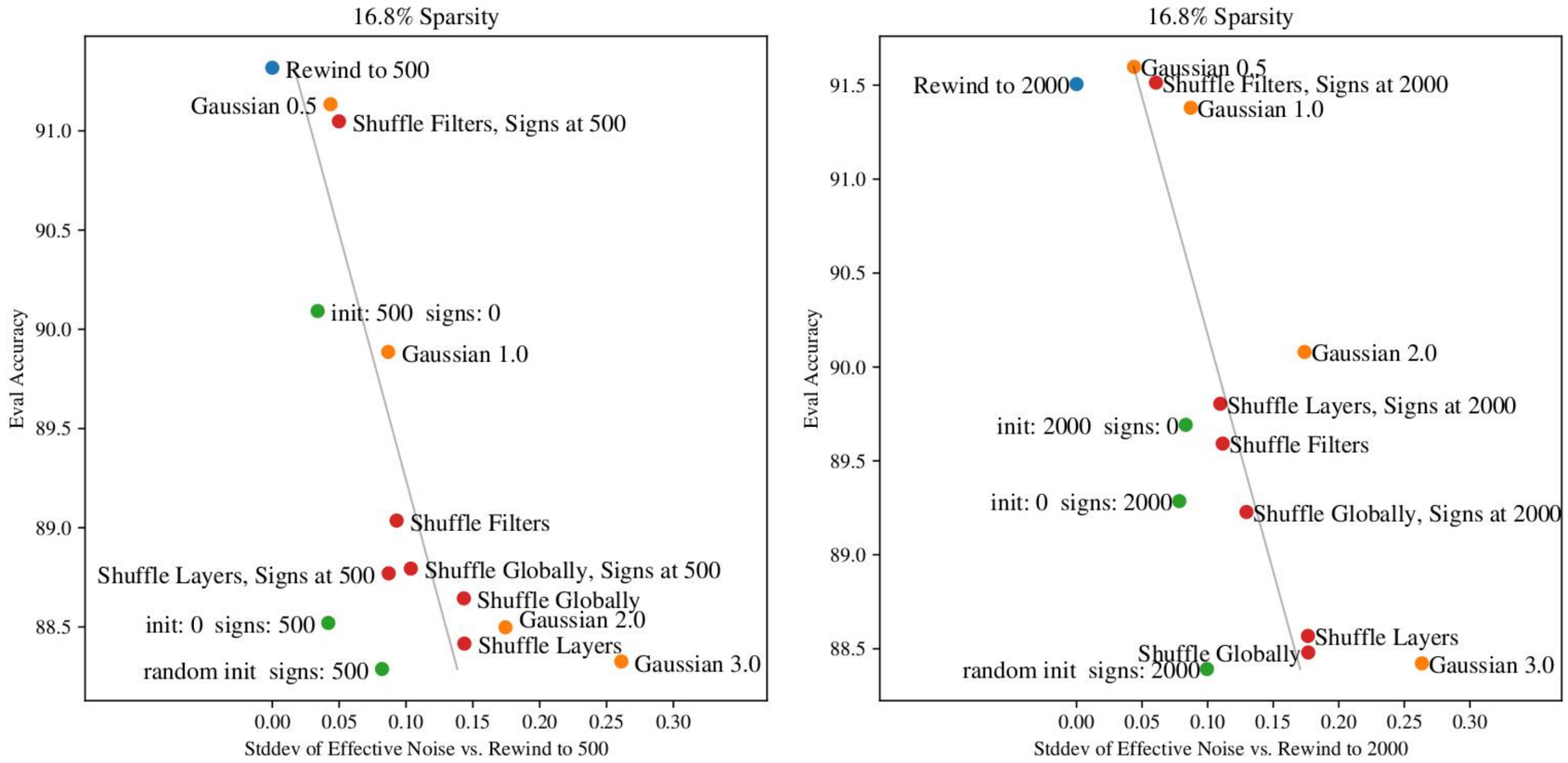


Figure 8: The effective standard deviation of various perturbations as a function of mean evaluation accuracy (across 5 seeds) at sparsity 26.2%. The mean of each perturbation was approximately 0. Left: $k = 500$, $r = -0.672$, $p = 0.008$; Right: $k = 2000$, $r = -0.726$, $p = 0.003$.

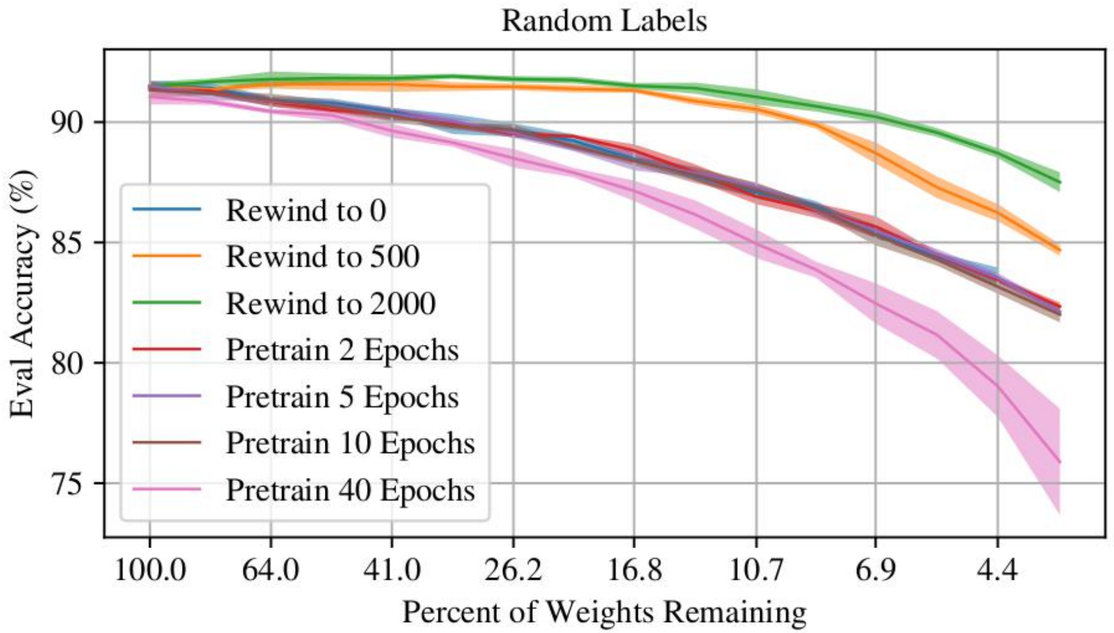
Data-Dependence of Neural Networks

- not due to easily-ascertainable, distributional properties of the network weights and signs
- which various aspects of the data distribution are necessary, p(x) or p(y|x)
- re-create a favorable network state (pretraining as init) using restricted information
- provide misleading labels, ignore labels entirely, or eliminate information

Data-Dependence of Neural Networks

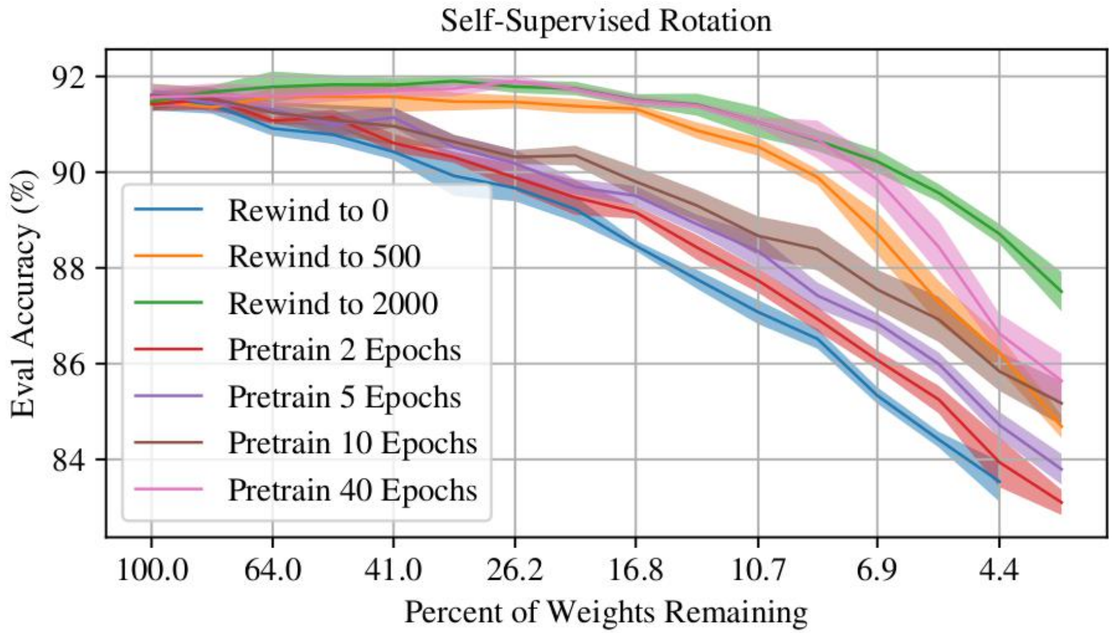
- Random Labels

no improvement for 10 epochs, and hurts for more epochs



- Self-Supervised Rotation

helpful but needs to train more epochs, 1.25 vs 40.
task label can accelarate the learning process

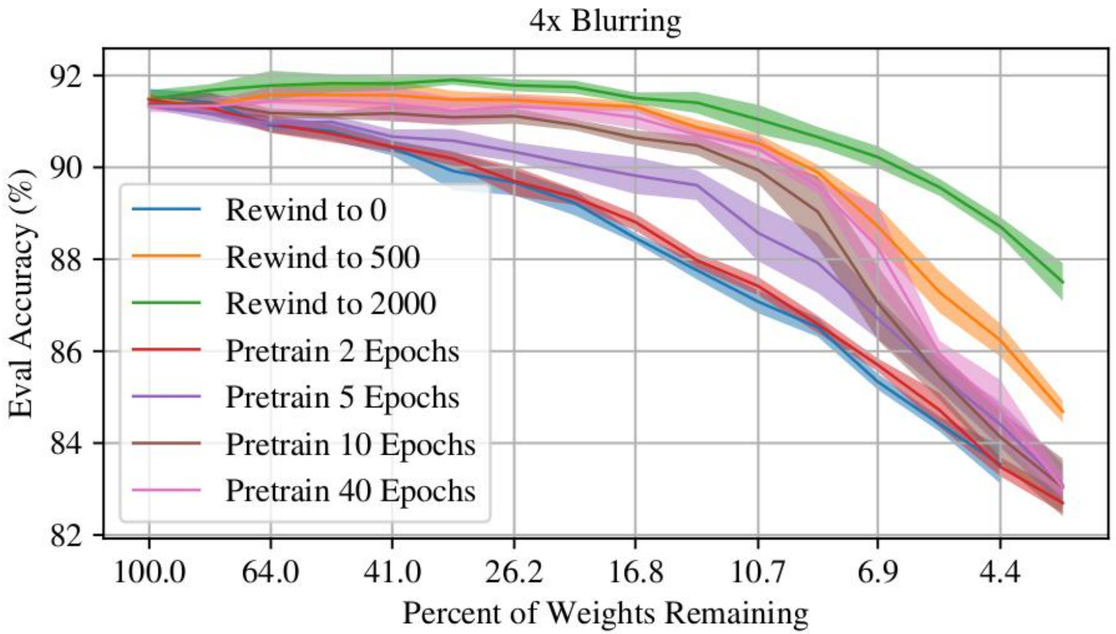


randomly been rotated 90n degrees, n $\in$ {0, 1, 2, 3},
pretrain to predict n

Data-Dependence of Neural Networks

- Blurring Training Examples

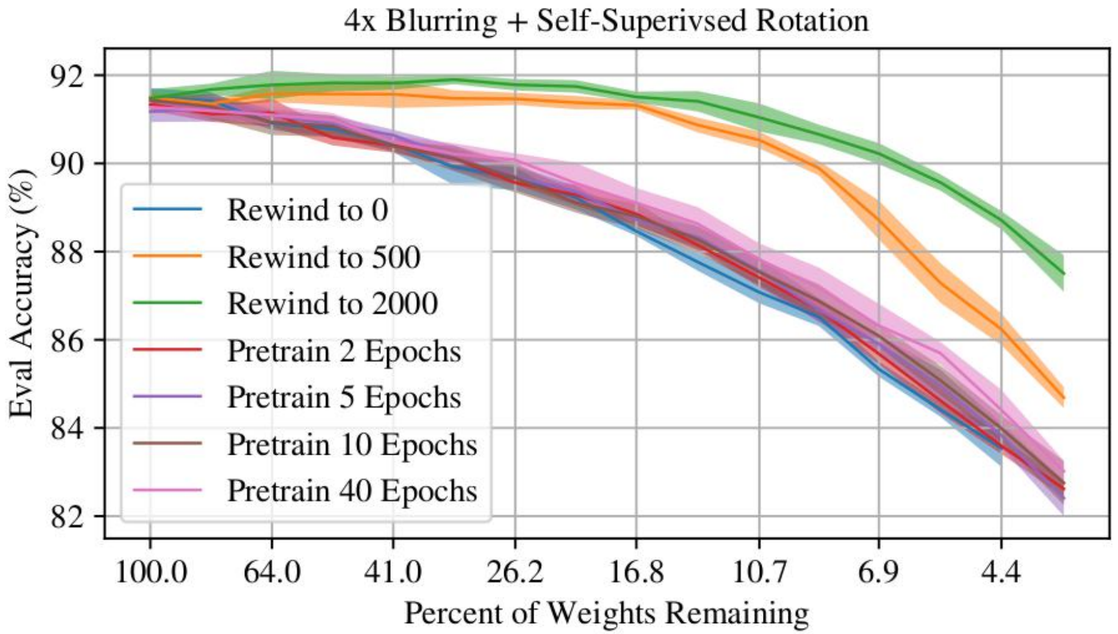
  also succeed, 40 vs 1.25



downsampling by 4x then upsampling back to the full size

- Combine Rotation and Blurring

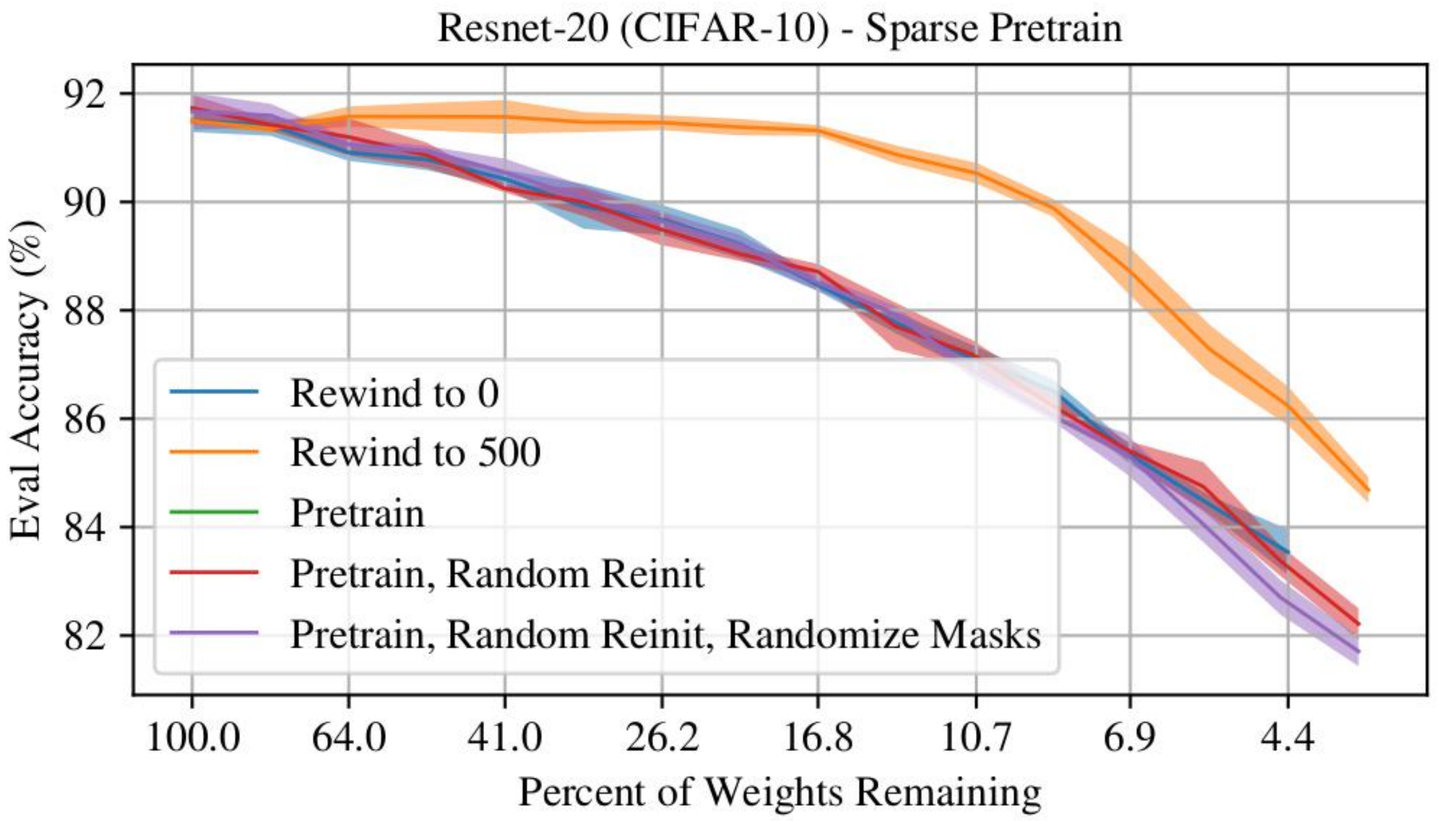
  faild, too little information



Sparse Pretrain

- Pretrain sparse sub-networks on rotation task

  no benefit found



Linear Mode Connectivity and the Lottery Ticket Hypothesis

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

ICML 2020

Motivation

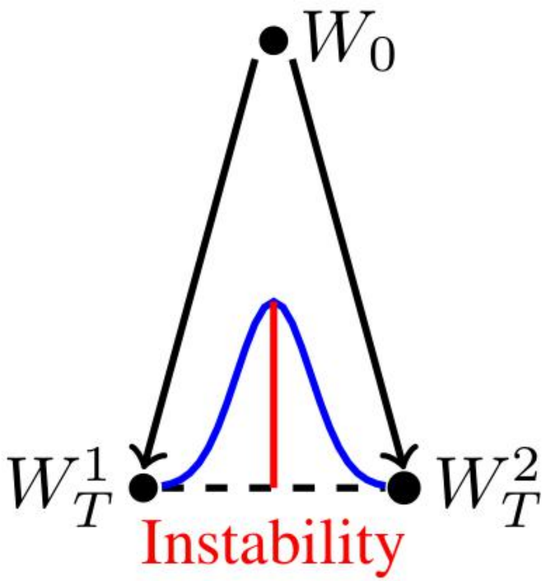
we train neural networks with stochastic gradient descent

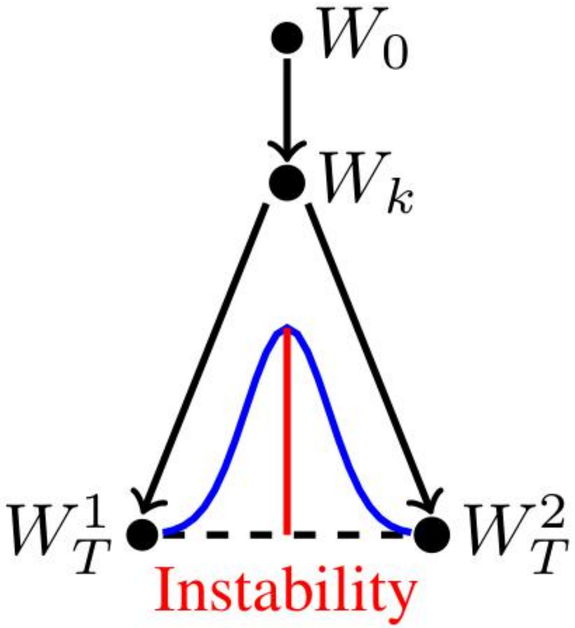
data order, data augmentation

How does this stochatic affect the optimization trajectories of neural networks ?

Methods

- study whether a neural network optimizes to the same, linearly connected minimum under different samples of SGD noise
- mode connectivity
  - the minima found by two networks are connected by a path of non-increasing error





Methods

- linear interpolation instability
  - error barrier height -> maximum increase in error along path

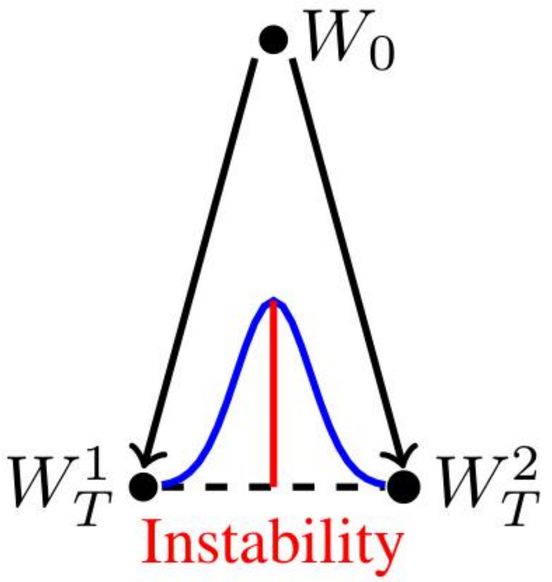
$$\mathcal{E}_\alpha(W_1, W_2) = \mathcal{E}(\alpha W_1 + (1-\alpha)W_2) \qquad \alpha \in [0,1]$$

$$\bar{\mathcal{E}}(W_1, W_2) = \text{mean}(\mathcal{E}(W_1), \mathcal{E}(W_2))$$

instability : $\mathcal{E}_{\text{sup}}(W_1, W_2) - \bar{\mathcal{E}}(W_1, W_2)$

the highest error between interpolating

$< 2\%$ -> stable, otherwise -> unstable



Experiment

- 30 evenly-spaced values of α
- 3 initializations and 3 runs per initialization (9 total)

Network	Variant	Dataset	Params	Train Steps	Batch	Accuracy	Optimizer	Rate	Schedule	Warmup	BatchNorm	Pruned Density	Style
LeNet		MNIST	266K	24K Iters	60	$98.3 \pm 0.1\%$	adam	12e-4	constant	0	No	3.5%	Iterative
ResNet-20	Standard	CIFAR-10	274K	63K Iters	128	$91.7 \pm 0.1\%$	momentum	0.1	10x drop at 32K, 48K	0	Yes	16.8%	Iterative
ResNet-20	Low					$88.8 \pm 0.1\%$		0.01		0		8.6%	
ResNet-20	Warmup					$89.7 \pm 0.3\%$		0.03		30K		8.6%	
VGG-16	Standard	CIFAR-10	14.7M	63K Iters	128	$93.7 \pm 0.1\%$	momentum	0.1	10x drop at 32K, 48K	0	Yes	1.5%	Iterative
VGG-16	Low					$91.7 \pm 0.1\%$		0.01		0		5.5%	
VGG-16	Warmup					$93.4 \pm 0.1\%$		0.1		30K		1.5%	
ResNet-50		ImageNet	25.5M	90 Eps	1024	$76.1 \pm 0.1\%$	momentum	0.4	10x drop at 30,60,80	5 Eps	Yes	30%	One-Shot
Inception-v3		ImageNet	27.1M	171 Eps	1024	$78.1 \pm 0.1\%$	momentum	0.03	linear decay to 0.005	0	Yes	30%	One-Shot

Instability analysis at initialization

- in general, larger-scale image classification networks are unstable at initialization

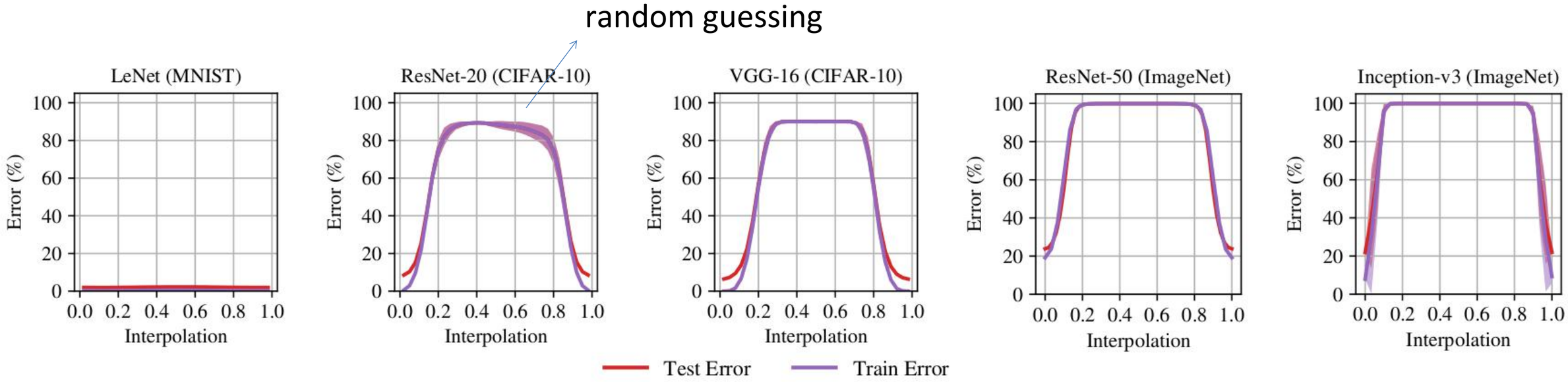


Figure 2. Error when linearly interpolating between networks trained from the same initialization with different SGD noise. Lines are means and standard deviations over three initializations and three data orders (nine samples total). Trained networks are at 0.0 and 1.0.

Instability analysis during training

- investigate when networks become stable
- test set instability decreases as k increases, culminating in stable networks

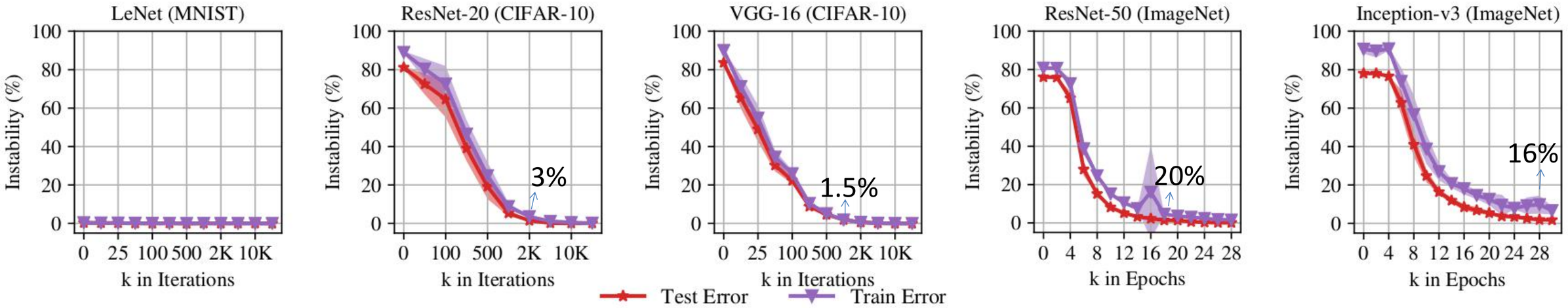


Figure 3. Linear interpolation instability when starting from step k. Each line is the mean and standard deviation across three initializations and three data orders (nine samples in total).

Instability analysis during training

- disentangling instability from training time
- linear mode connectivity vs training times
- reset the learning rate schedule to iteration 0 and train T epochs

training steps did not play a role

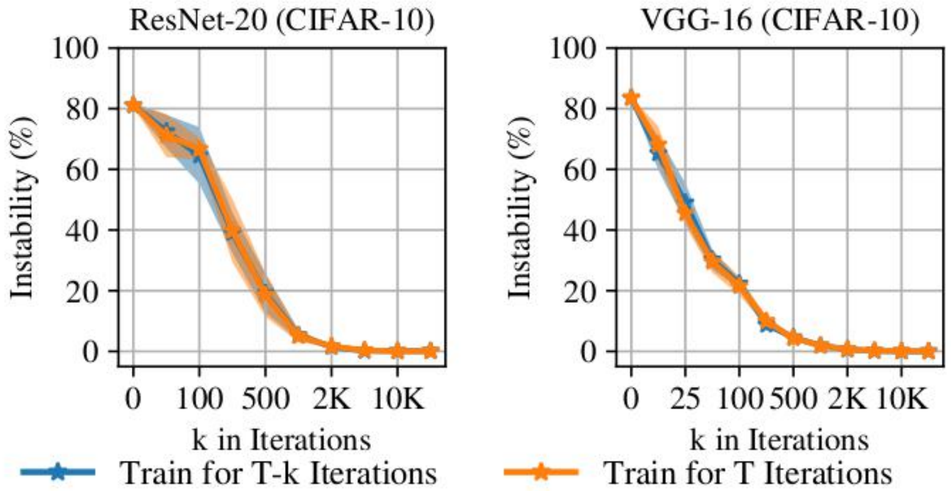


Figure 4. Linear interpolation instability on the test set when making two copies of the state of the network at step k and either (1) training for the remaining $T - k$ steps (blue) or (2) training for T steps with the learning rate schedule reset to step 0 (orange).

Instability Analysis of Lottery Tickets

- in more challenging settings, IMP subnetworks perform no better than subnetworks chosen randomly
- a potential explanation for their successes and failures

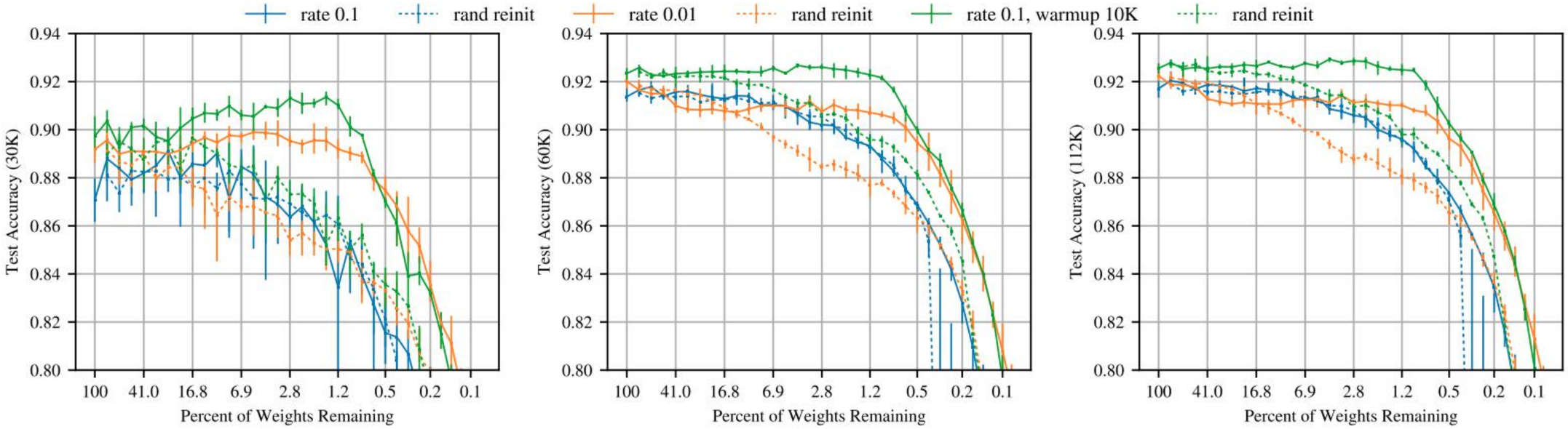


Figure 7: Test accuracy (at 30K, 60K, and 112K iterations) of VGG-19 when iteratively pruned.

Instability Analysis of Lottery Tickets

- Instability analysis of subnetworks at initialization

Network	Full	IMP	Rand Prune	Rand Reinit	Δ IMP	Matching?
LeNet	98.3	98.2	96.7	97.5	0.1	Y
ResNet-20	91.7	88.5	88.6	88.8	3.2	N
ResNet-20 Low	88.8	89.0	85.7	84.7	-0.2	Y
ResNet-20 Warmup	89.7	89.6	85.7	85.6	0.1	Y
VGG-16	93.7	90.9	89.4	91.0	2.8	N
VGG-16 Low	91.7	91.6	90.1	90.2	0.1	Y
VGG-16 Warmup	93.4	93.2	90.1	90.7	0.2	Y
ResNet-50	76.1	73.7	73.1	73.4	2.4	N
Inception-v3	78.1	75.7	75.2	75.5	2.4	N

Table 2. Accuracy of IMP and random subnetworks when rewinding to $k = 0$ at the sparsities in Table 1. Accuracies are means across three initializations. All standard deviations are < 0.2.

Instability Analysis of Lottery Tickets

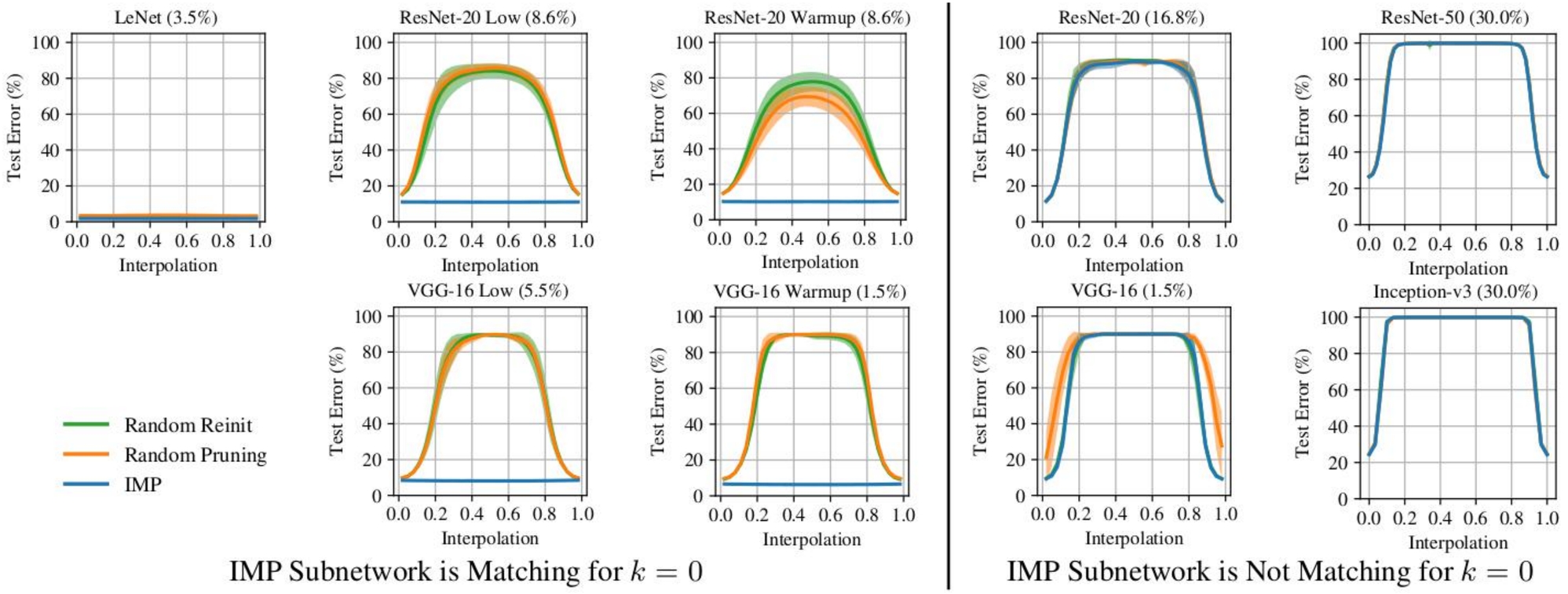


Figure 5. Test error when linearly interpolating between subnetworks trained from the same initialization with different SGD noise. Lines are means and standard deviations over three initializations and three data orders (nine in total). Percents are weights remaining.

Instability Analysis of Lottery Tickets

- Instability analysis of subnetworks during training.

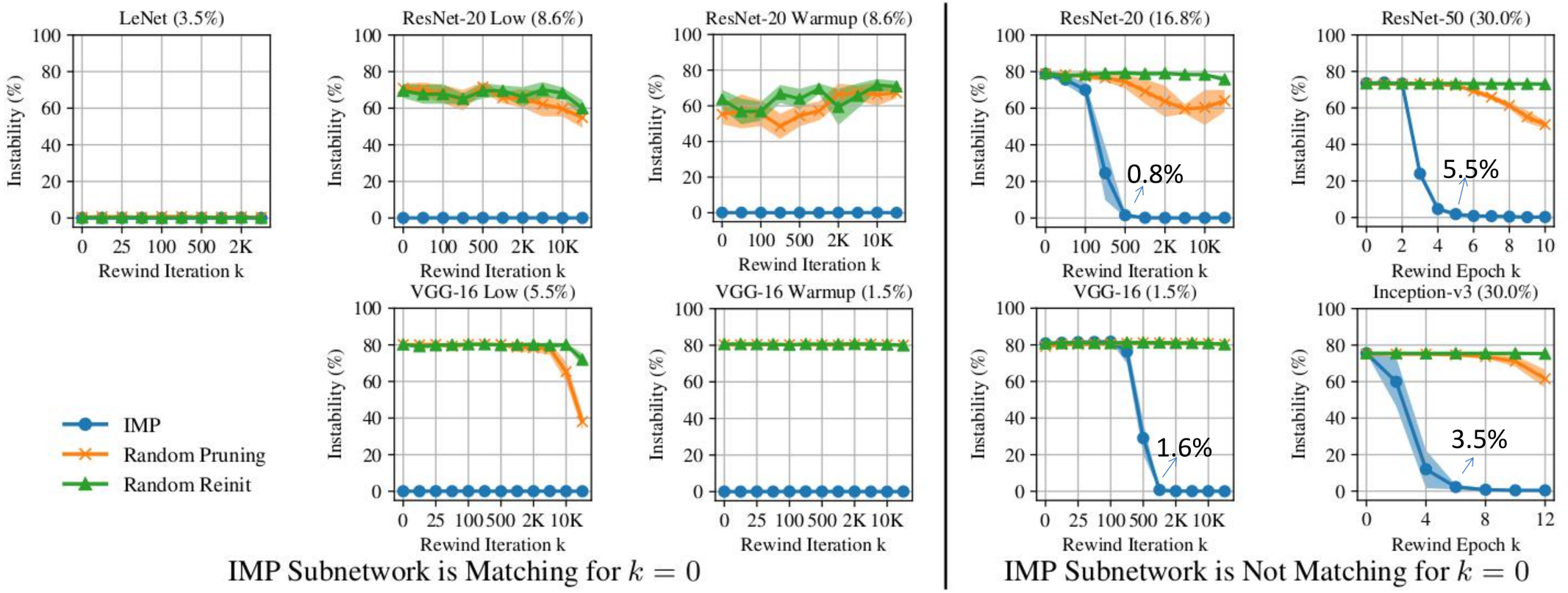


Figure 6. Linear interpolation instability of subnetworks created using the state of the full network at step k and applying a pruning mask. Lines are means and standard deviations over three initializations and three data orders (nine in total). Percents are weights remaining.

Instability Analysis of Lottery Tickets

- Instability analysis of subnetworks during training.

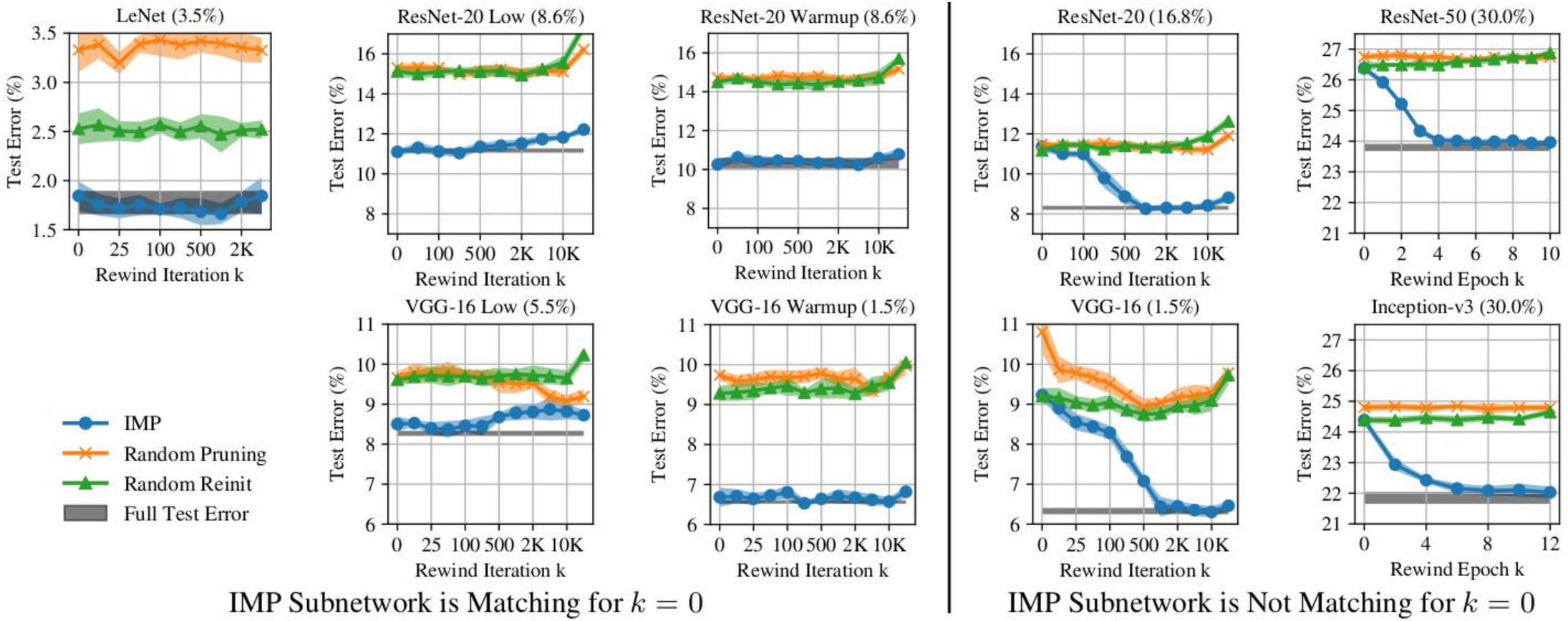


Figure 7. Test error of subnetworks created using the state of the full network at step k and applying a pruning mask. Lines are means and standard deviations over three initializations and three data orders (nine in total). Percents are weights remaining.

Results at Other Sparsity Levels

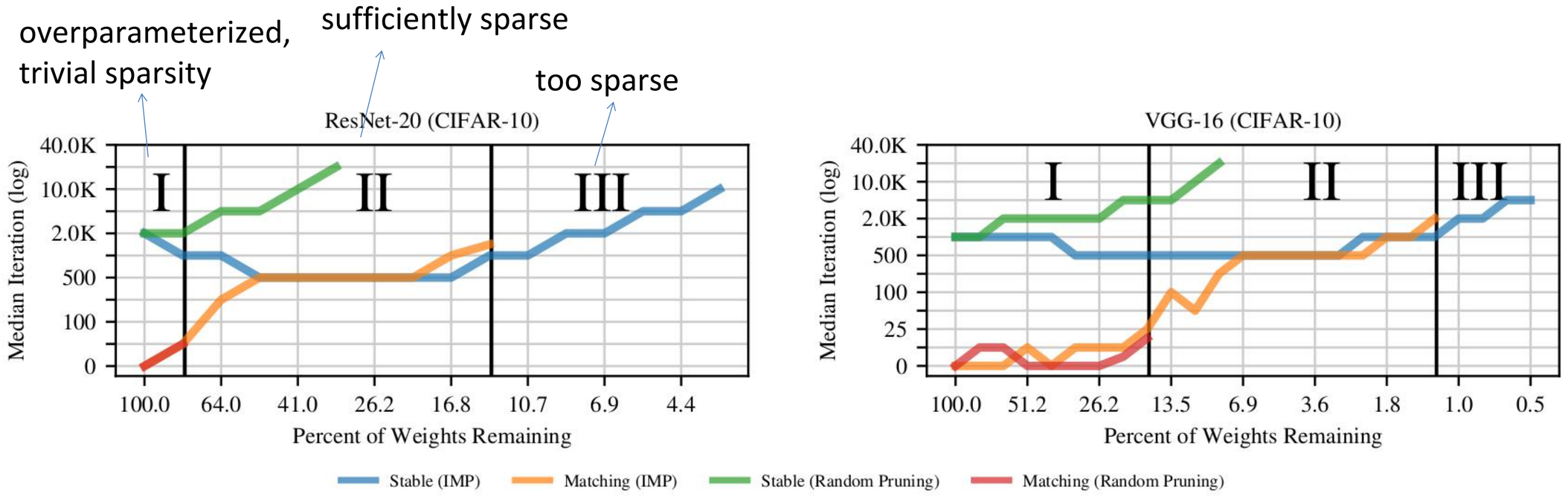


Figure 9. The median rewinding iteration at which IMP subnetworks and randomly pruned subnetworks of ResNet-20 and VGG-16 become stable and matching. A subnetwork is stable if instability $< 2\%$. A subnetwork is matching if the accuracy drop $< 0.2\%$; we only include points where a majority of subnetworks are matching at any rewinding iteration. Each line is the median across three initializations and three data orders (nine samples in total).

Conclusion

- model learns a lot at begining and rewinding is helpful
- stability emerges early in training
- changing aspects of optimization once the network becomes stable to improve performance ?

Sanity Checks for Lottery Tickets: Does Your Winning Ticket Really Win the Jackpot?

Xiaolong Ma[1,†], **Geng Yuan**[1,†], **Xuan Shen**[1], **Tianlong Chen**[2], **Xuxi Chen**[2], **Xiaohan Chen**[2], **Ning Liu**[3], **Minghai Qin**, **Sijia Liu**[4], **Zhangyang Wang**[2], **Yanzhi Wang**[1]

[1] Northeastern University, [2] University of Texas at Austin
[3] Midea Group, [4] Michigan State University
{ma.xiaol,yanz.wang}@northeastern.edu

NeurIPS 2021

Motivation

- key training hyperparameters such as learning rate and training epochs were not scrutinized nor exhaustively tuned
  - lr, training epochs ...
- What makes the comprehensive condition to define the lottery ticket hypothesis?
- Do winning tickets exist across the major DNN architectures under such definition?
- What are the intrinsic reasons for their existence or non-existence?

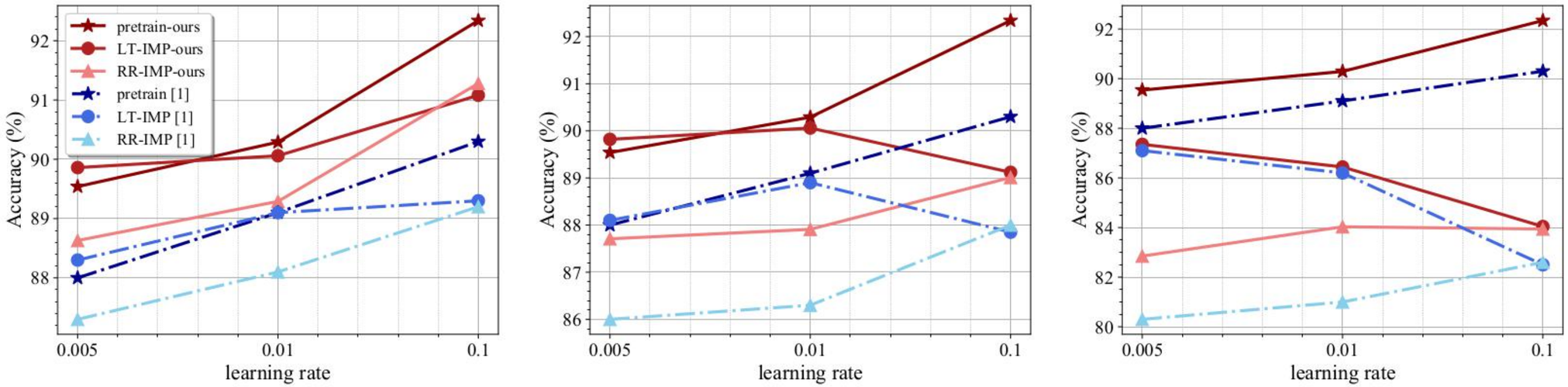
Origin Definition

- $f(x;\theta_0 \odot m) \longrightarrow f(x;(\theta_0 \odot m)_T) \approx f(x;\theta_T)$
  - $f(x;(\theta_0 \odot m)_T) \; \rangle \; f(x;(\theta_0' \odot m)_T)$

- well-trained
  - appropriate lr, sufficient training epochs ...

Origin Definition

- the “winning ticket” exists in smaller lr but not larger lr (a, b)
- at same lr, insufficient training recipe can be misleading (c)



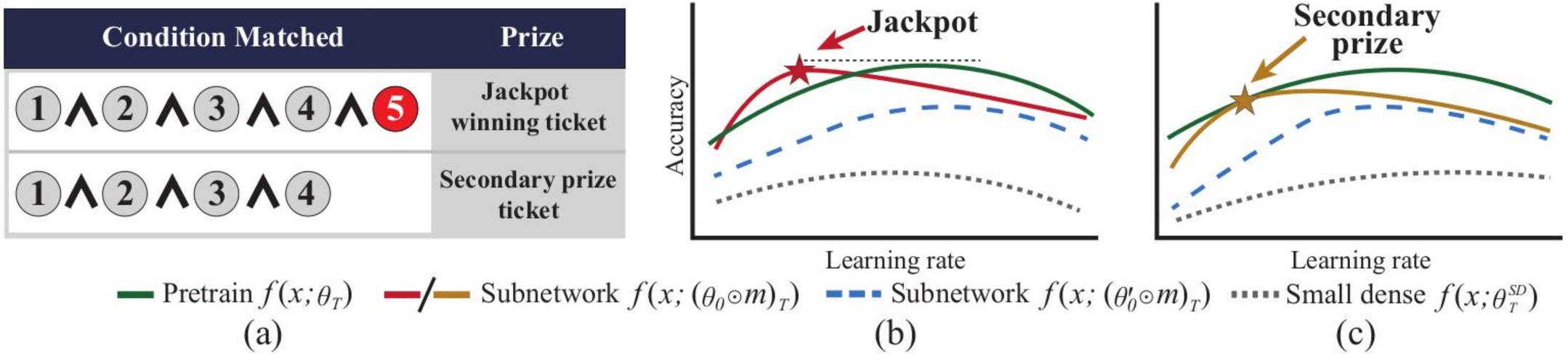
(a) sparsity ratio $s = 0.59$ (b) sparsity ratio $s = 0.832$ (c) sparsity ratio $s = 0.914$

Figure 1: Preliminary results of ResNet-20 on CIFAR-10 dataset with different learning rates and sparsity ratios. We train the network using 160 epochs, while [1] uses 78 epochs. Please refer to [1] and Appendix A for the full results of all sparsity levels.

RR: random reinitialization

Rigorous Definition

- A non-trivial sparsity ratio s and a sufficient training epochs T
- clear accuracy advantage over a small dense ntework
- There exists a lr such that $f(x;(\theta_0 \odot m)_T)$ > $f(x;(\theta'_0 \odot m)_T)$ with any lr (>0.5%)
- There exists a lr such that $f(x;(\theta_0 \odot m)_T)$ ≥ $f(x;\theta_T)$ at same lr
- There exists a lr such that $f(x;(\theta_0 \odot m)_T)$ ≥ $f(x;\theta_T)$ well-trained



Experiment

Table 2: Dataset and network we evaluate using the re-definition of the lottery ticket hypothesis.

Dataset	CIFAR-10		CIFAR-100			Tiny-ImageNet		ImageNet-1K	
#Images	50K/10K		50K/10K			100K/10K		1.28M/50K	
#Classes	10		100			200		1000	
Img Size	32×32		32×32			64×64		224×224	
Network	RN-20	RN-32	MBNet-v1	RN-18	VGG-16	RN-18	RN-50	RN-18	RN-50
#Params.	0.27M	1.86M	3.21M	11.22M	14.72M	11.68M	25.56M	11.69M	25.56M

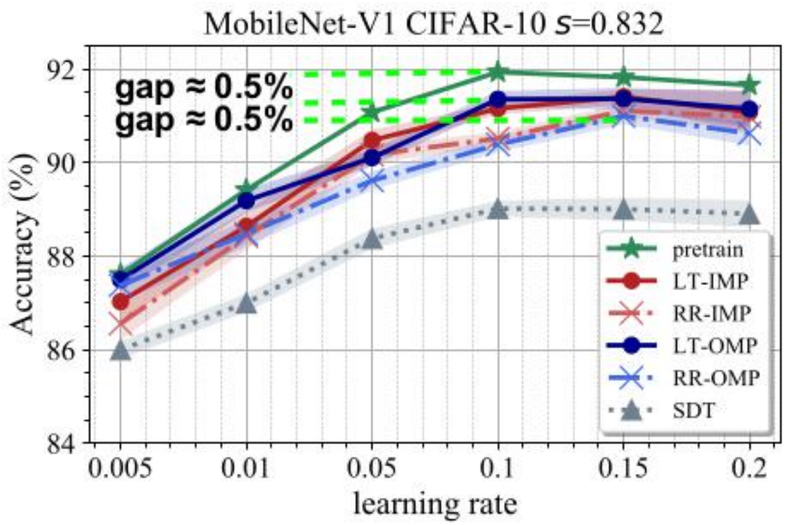
Table 3: Summary of the observations of all experiments.

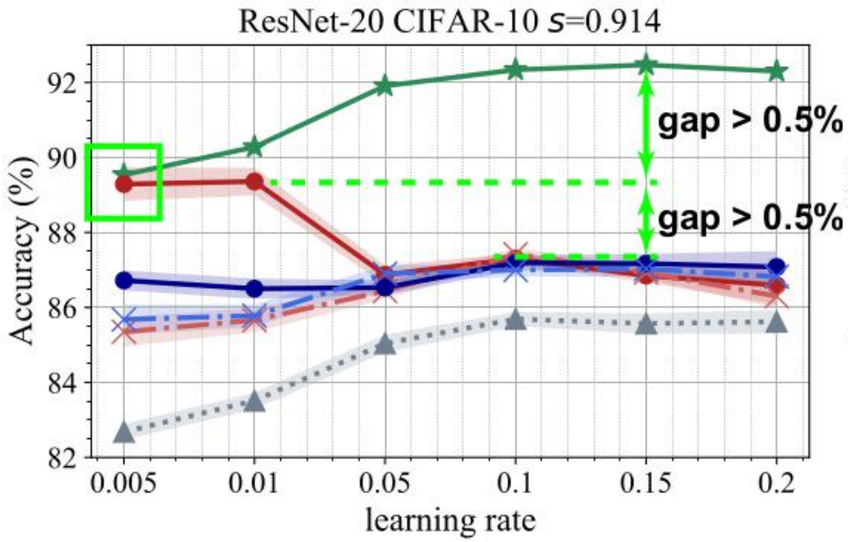
Dataset	RN20	RN32	MBNet-v1	RN18	VGG-16	Dataset	RN18	RN50
CIFAR-10	✗ ✓ ✓ ✓	✗ ✓ ✓ ✓	△ ✓ ✗ ✗	✗ ✓ ✓ ✗	✗ ✗ ✗ ✗	Tiny-ImageNet	✗ ✗ ✗ ✗	✗ ✗ ✗ ✗
CIFAR-100	✗ ✓ ✓ ✓	✗ ✓ ✓ ✓	✗ ✗ ✗ ✗	△ ✓ ✓ ✓	✗ ✗ ✗ ✗	ImageNet-1K	✗ ✗ ✗ ✗	✗ ✗ ✗ ✗

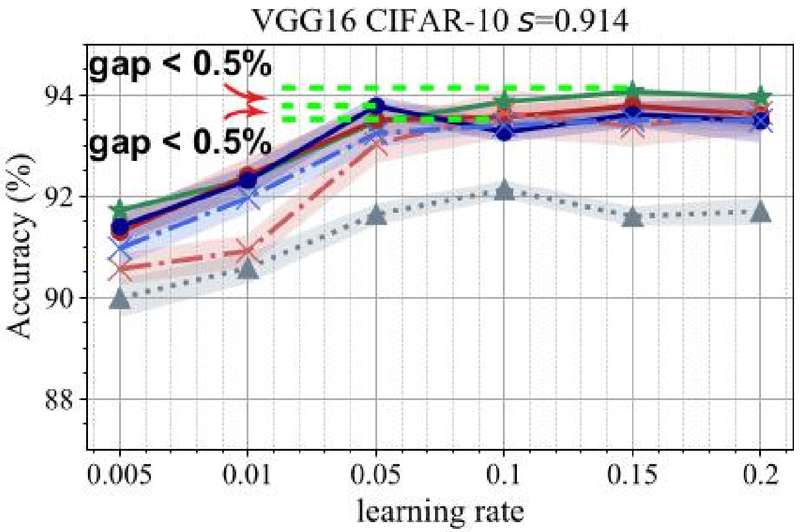
Jackpot Secondary Prefer small lr Prefer IMP ✓ Yes ✗ No △ At boundary

Experiment

- no clear Jackpot winning tickets are found
- secondary prize tickets exist in most of the networks on small datasets
  - biggger model, larger sparsity
  - no clear secondary prize tickets found for larger dataset, maybe larger model

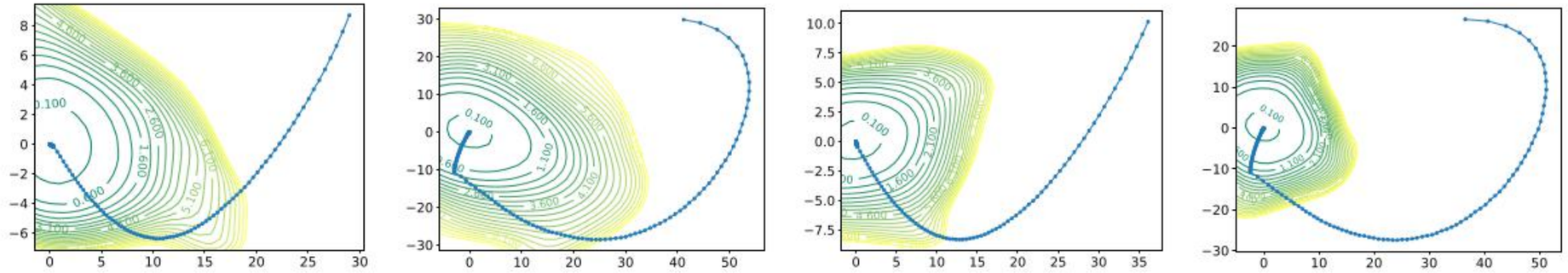






IMP vs OMP

when residual connections exist, IMP > OMP, otherwise no advantage

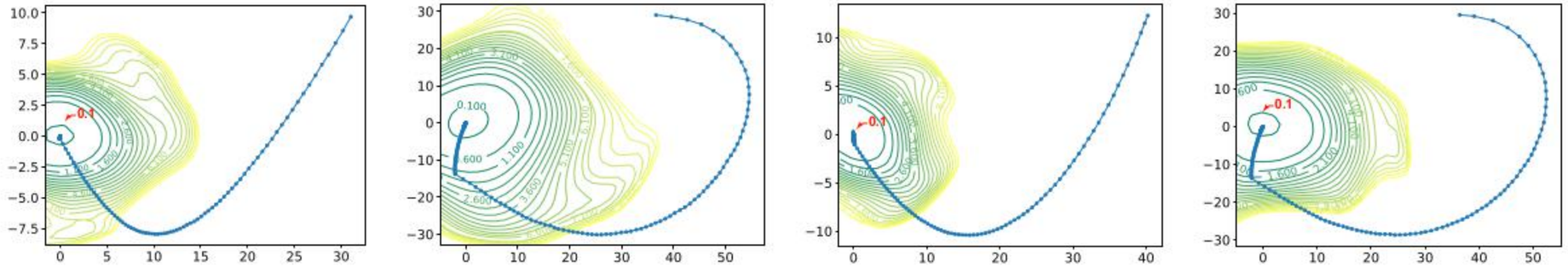


(a) IMP, lr=0.01, Acc=92.9% (b) IMP, lr=0.1, Acc=91.4% (c) OMP, lr=0.01, Acc=91.8% (d) OMP, lr=0.1, Acc=91.6%

Figure 4: Training trajectories along the loss surface contours of ResNet-32 on CIFAR-10 at sparsity ratio of 0.945.

contour is smoother and close-to-convex

the optimization becomes too difficult, neither IMP nor OMP can find a smooth route



(a) IMP, lr=0.01, Acc=89.4% (b) IMP, lr=0.1, Acc=89.6% (c) OMP, lr=0.01, Acc=89.7% (d) OMP, lr=0.1, Acc=90.7%

Figure 5: Training trajectories along the loss contours of ResNet-32-like network without residual connections on CIFAR-10 at sparsity ratio of 0.945.

Experiment

- different lr for subnetwork training may be helpful

Table 4: Ablation results using ResNet-20 on CIFAR-10 at sparsity 0.914. The shaded area indicates the learning rate that finds the better subnetwork accuracy.

Pretraining lr (Acc %): 0.01 (90.3)			Pretraining lr (Acc %): 0.1 (92.4)		
LT lr	IMP Acc (%)	OMP Acc (%)	LT lr	IMP Acc (%)	OMP Acc (%)
0.001	87.5	83.3	0.01	85.3	85.8
0.005	**89.7**	85.3	0.05	86.6	**87.4**
0.01	89.4	86.5	0.1	87.3	87.2
0.05	87.9	87.2	0.15	86.7	87.3

Guideline for LTH

- small dataset using networks with residual connections, IMP > OMP, no residual connections -> no advantage
- small dataset using networks with residual connections, small lr is better, when no residual connection, prefer larger lr
- when the network is redundant, the sparsity can be higher
- when using larger lr, using different initialization yields the similar accuracy in subnetwork training

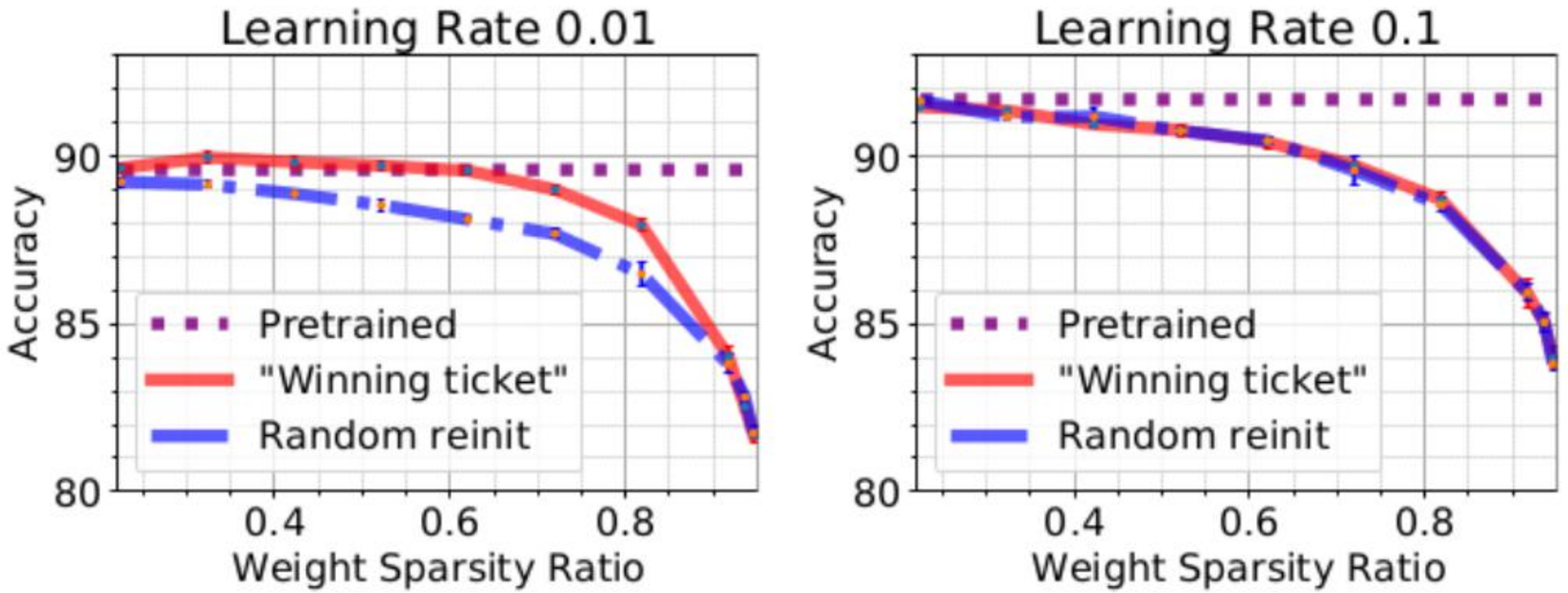
Lottery Ticket Preserves Weight Correlation: Is it Desirable or Not?

Ning Liu [1 *] **Geng Yuan** [2 *] **Zhengping Che** [3] **Xuan Shen** [2] **Xiaolong Ma** [2] **Qing Jin** [2] **Jian Ren** [4] **Jian Tang** [1 †]
Sijia Liu [5] **Yanzhi Wang** [2 †]

ICML 2021

Motivation

- investigate the precise condition when winning property exists



(a) Iterative pruning at learning rate of 0.01.

(b) Iterative pruning at learning rate of 0.1.

Method

- hypothesis: correlation between initialized weights and final-trained weights when the lr is not sufficiently large
- Weight Correlation Indicator

$$R_p(\theta, \theta') = \frac{\sum_l \mathbf{card}\left(T_p\left((\theta)^l\right) \cap T_p\left((\theta')^l\right)\right)}{p \cdot \sum_l N_l}$$

- $T_p\left((\theta)^l\right)\ (p \in [0,1])$ top-p largest-magnitude weights in the l-layer

$R_p(\theta, \theta') > p$ positive correlation

$R_p(\theta, \theta') \approx p$ no correlation

$R_p(\theta, \theta') < p$ negative correlation

Method

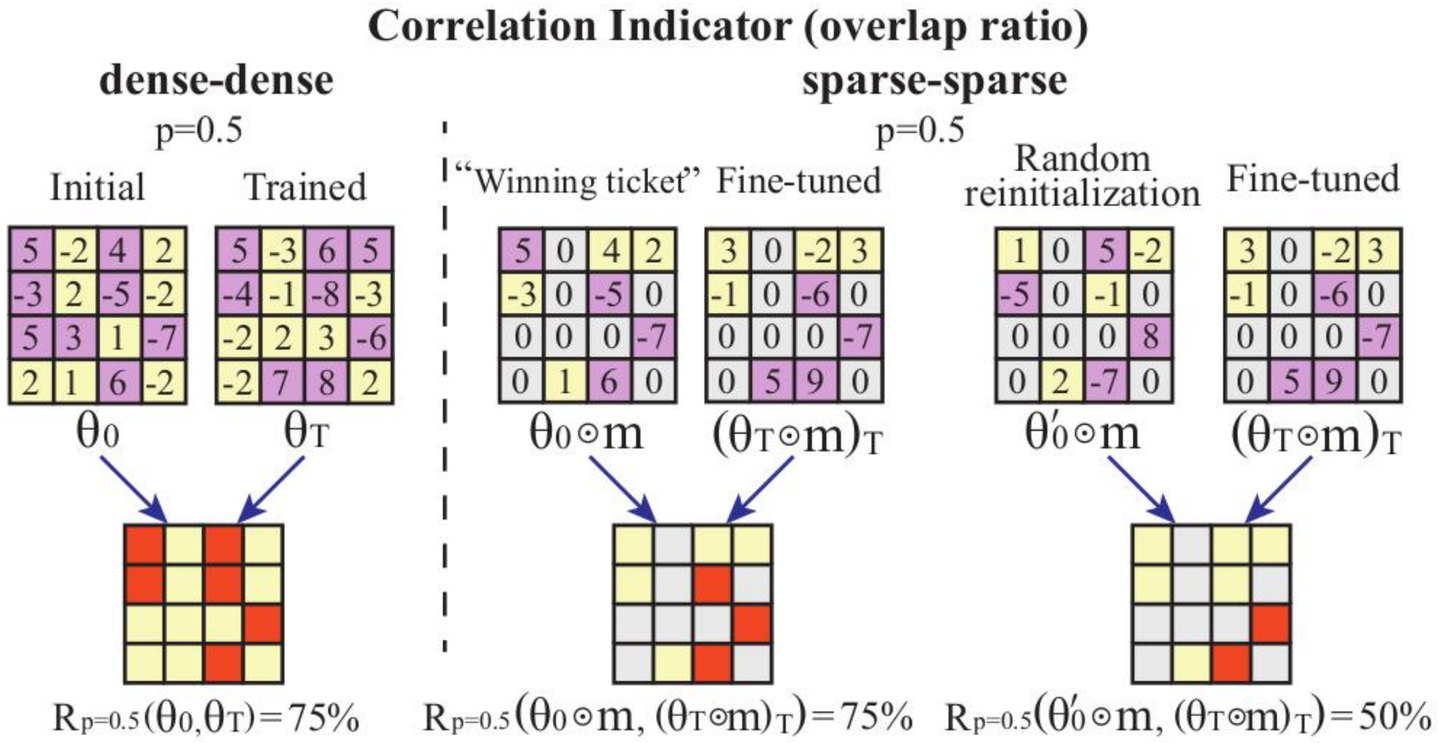


Figure 3. Scenarios for quantitative analysis of the weight correlation with an example of $sparsity\ ratio = 50\%$ and $p = 0.5$. This example is one DNN layer, while our actual computation is on the whole DNN.

Weight Correlation in DNN Pre-Training

- a notably higher correlation compared to larger lr

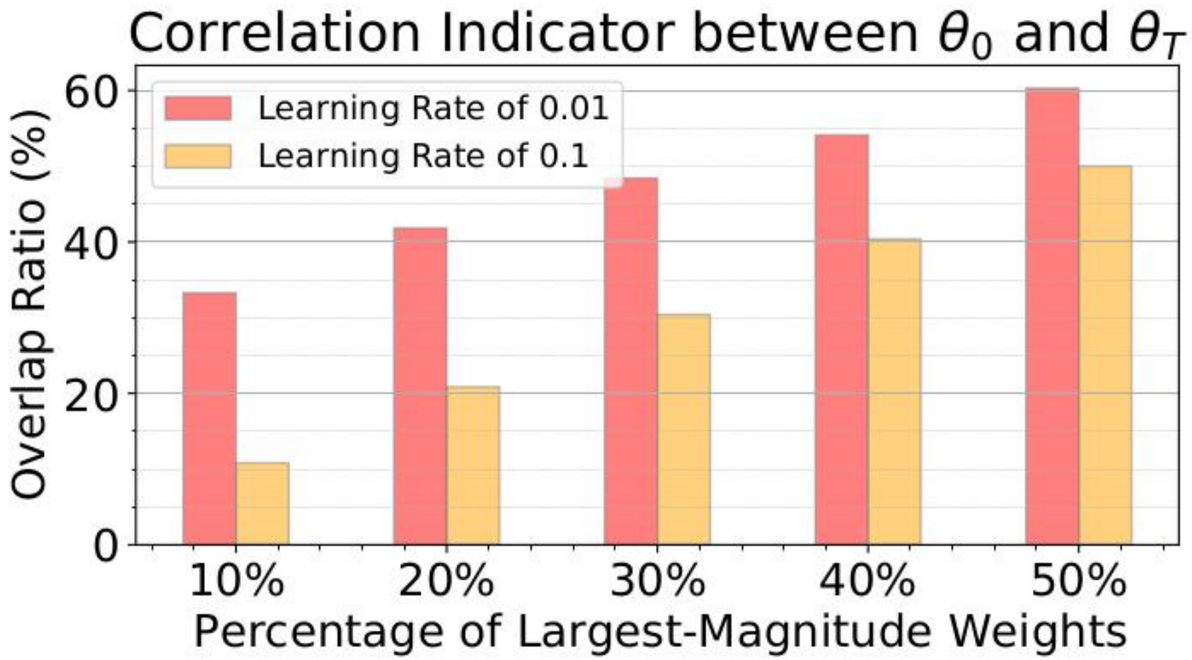


Figure 4. The overlap ratios (when $p = 10\%, 20\%, 30\%, 40\%$ and 50%) between the initial weights θ_0 and the pretrained weights θ_T at learning rate of 0.01 and 0.1.

training epoch is 150

Condition of the Winning Property

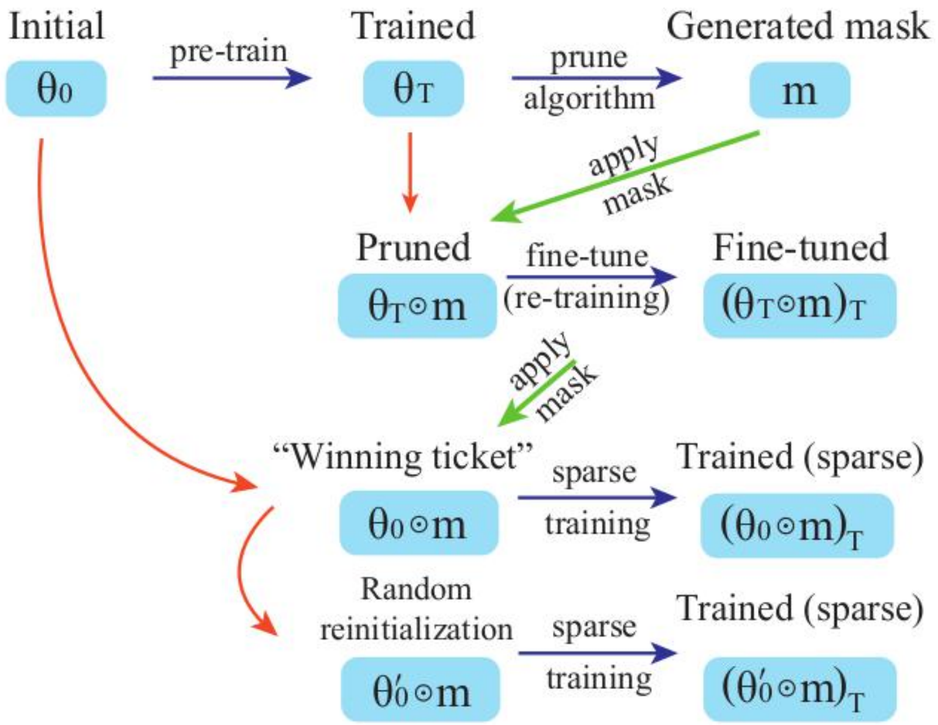
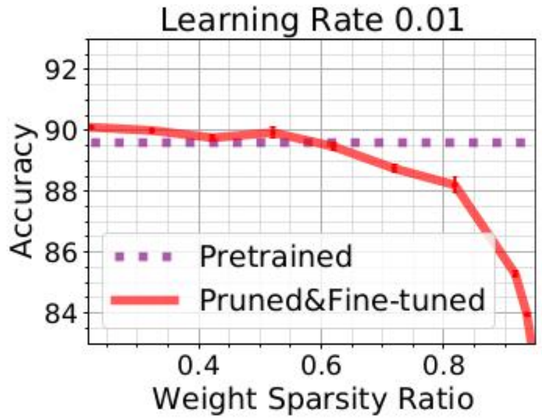
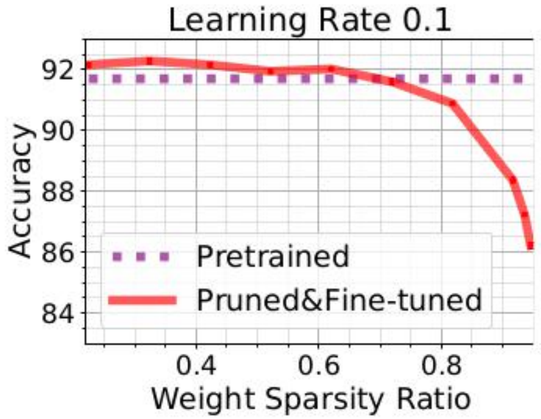


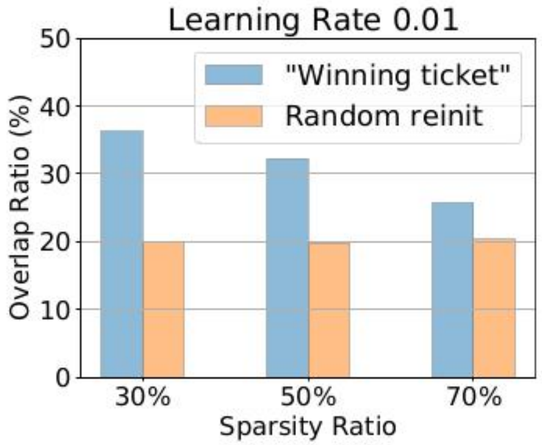
Figure 1. Illustration of notations: "pre-training", "pruning" (mask generation), "sparse training", and "pruning & fine-tuning".



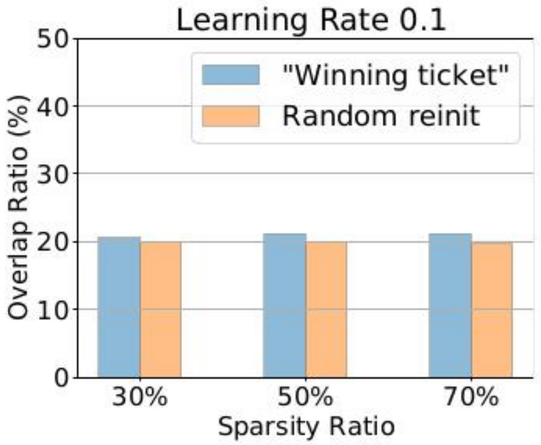
(a) Pruning&fine-tuning at learning rate of 0.01.



(b) Pruning&fine-tuning at learning rate of 0.1.

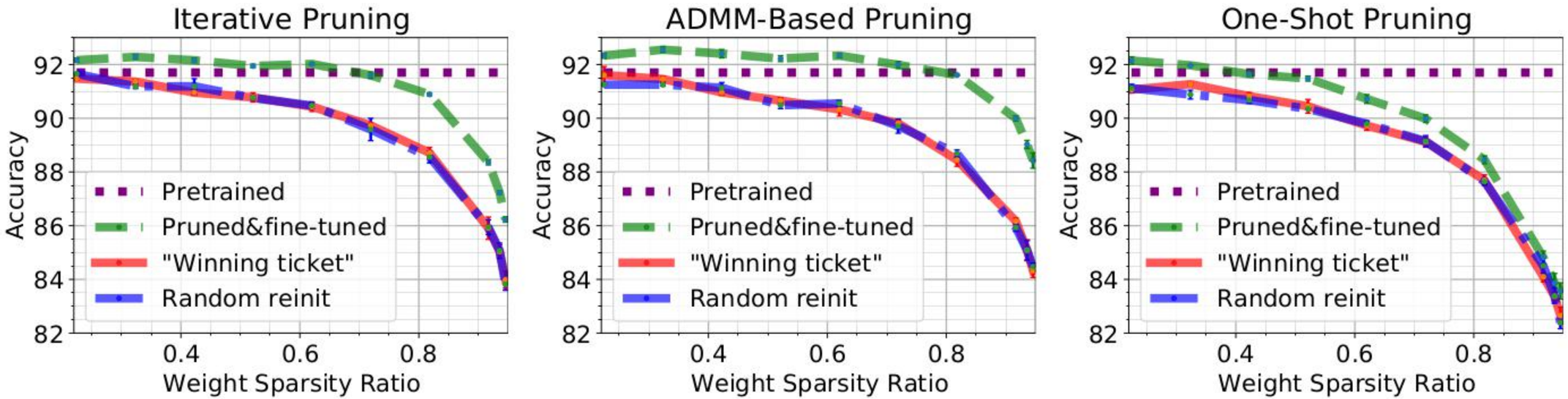


(c) The overlap ratio comparison at learning rate 0.01.



(d) The overlap ratio comparison at learning rate 0.1.

Comparing with LTH



(a) The iterative pruning at learning rate of 0.1.

(b) The ADMM-based pruning at learning rate of 0.1.

(c) The least magnitude one-shot pruning at learning rate of 0.1.

Figure 6. Accuracy performance of pruning & fine-tuning vs. two sparse training cases ("winning ticket" and random reinitialization). Three pruning algorithms are utilized for mask generation: iterative pruning (a), ADMM-based pruning (Zhang et al., 2018) (b), and one-shot pruning (c).

Resnet-20 on Cifar10

Comparing with LTH

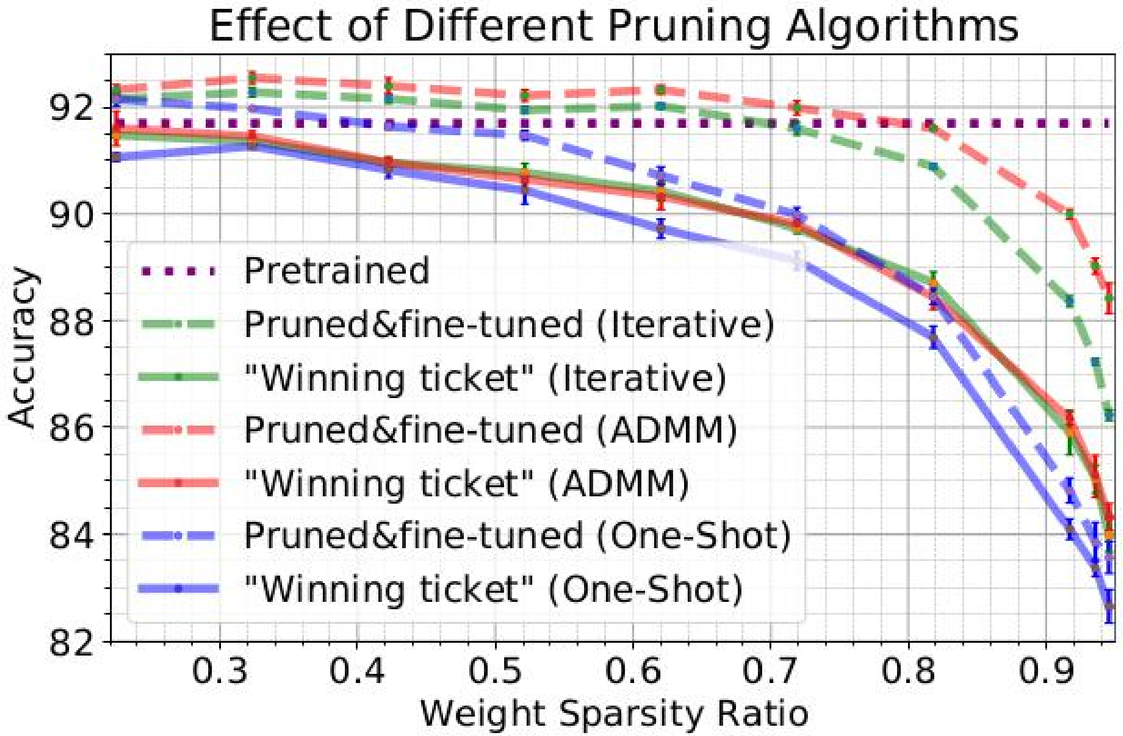


Figure 7. Accuracy performances of pruning & fine-tuning and sparse training ("winning ticket" case), under all three pruning algorithms (iterative pruning, ADMM-based pruning, and one-shot pruning) for mask generation.