To the lottery ticket hypothesis and beyond

Can we really make training efficient?

Atelier DSAIDIS optimisation et réseaux de neurones

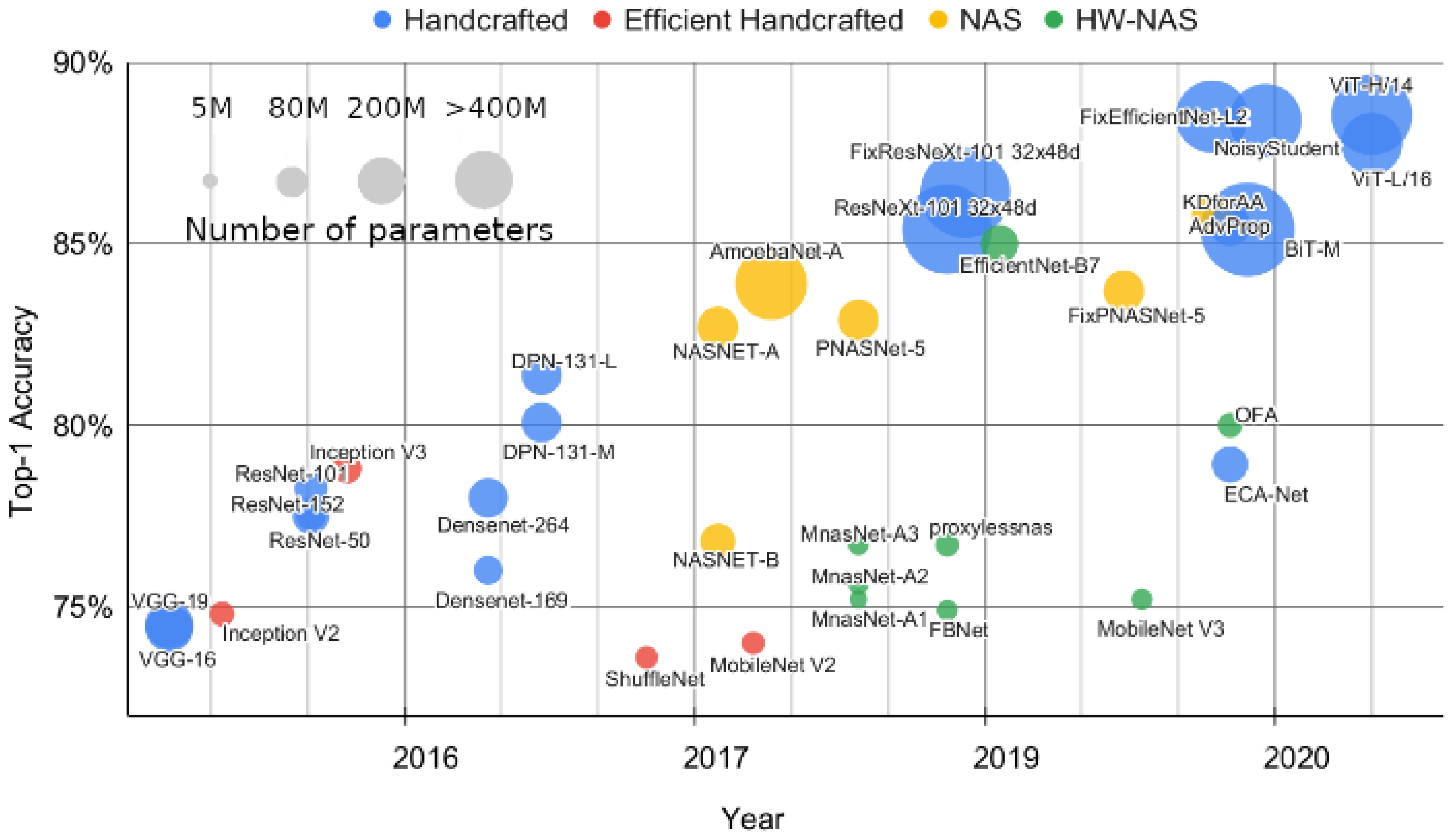
Enzo Tartaglione
Maître de Conférences, équipe MM, dept.IDS, LTCI
Hi!PARIS chair holder

enzo.tartaglione@telecom-paris.fr

Outline

- Pruning 101
- The lottery ticket hypothesis
- Beyond the lottery ticket hypothesis: the rise of the lottery heroes
- Real gain at computation time: neurons at equilibrium
- Conclusion

Recent trends in ANNs



]

Frugality in AI

- **Perform training with little data**
  - Better optimizers
  - New loss functions
  - Ease the features extraction with priors
  - Transfer learning
  - …

Frugality in AI

- **Perform training with little data**
  - Better optimizers
  - New loss functions
  - Ease the features extraction with priors
  - Transfer learning
  - …
- **Use the trained AI models with little computational resources**
  - Pruning & Quantization
  - Knowledge distillation
  - Neural Architecture Search
  - …

Frugality in AI

- **Perform training with little data**
  - Better optimizers
  - New loss functions
  - Ease the features extraction with priors
  - Transfer learning
  - …
- **Use the trained AI models with little computational resources**
  - Pruning & Quantization
  - Knowledge distillation
  - Neural Architecture Search
  - …
- **Train with little computational resources**
  - Ensure fast convergence for models
  - Reduce the training cost
  - …

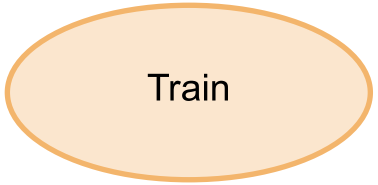
Pruning in deep learning: why?

- Reducing the storage memory (for a compressed model).
- Reducing the memory footprint.
- Reducing the FLOPs at inference time.
- Reducing energy consumption?
- Enhancing generalization?

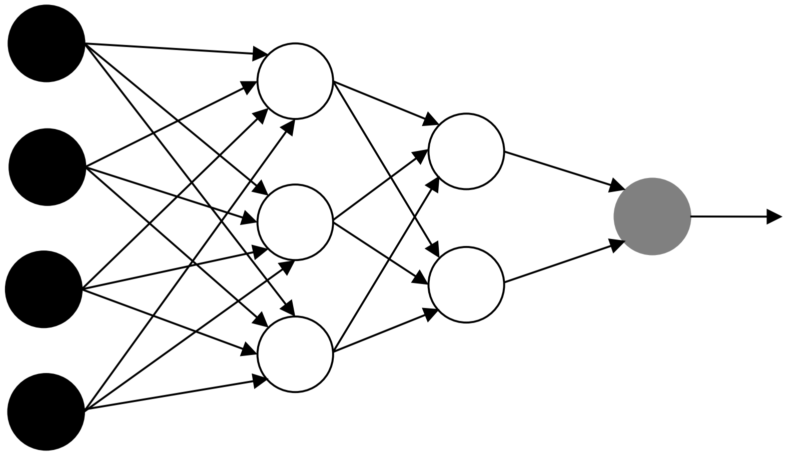
Countless approaches to achieve sparse models…

How to prune a deep model?

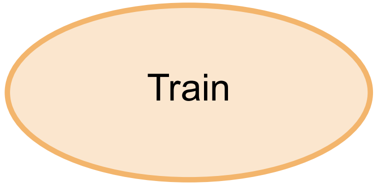
Pruning 101


Train

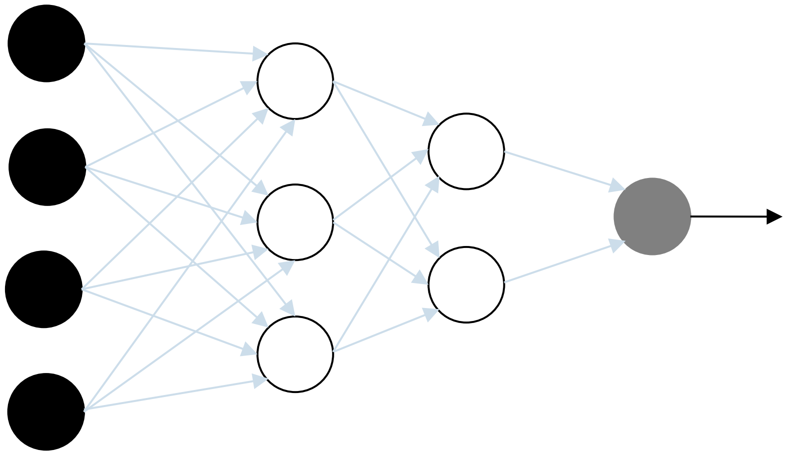
- Parameters are randomy initialized



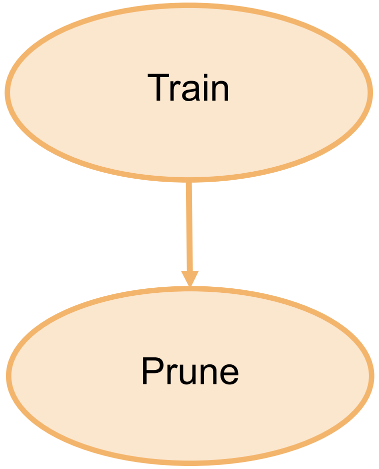
Pruning 101



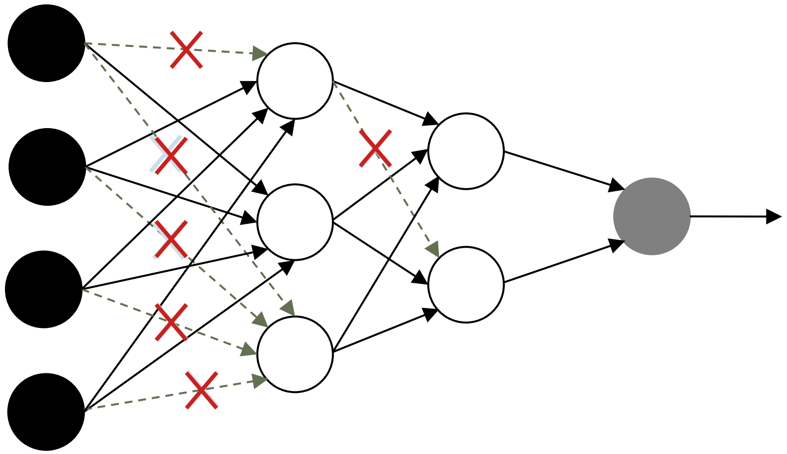
- Parameters are randomy initialized
- Parameters are updated then trained with standard gradient descent until performance is achieved (training stage)



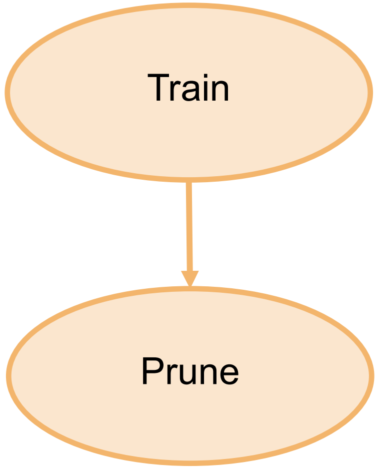
Pruning 101



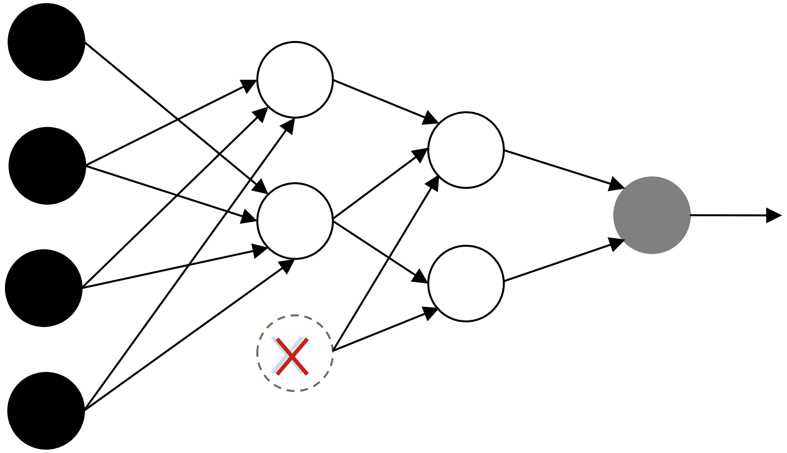
- Parameters are randomy initialized
- Parameters are updated then trained with standard gradient descent until performance is achieved (training stage)
- Parameters below threshold T are removed, pruning connections (parameter sparsification)



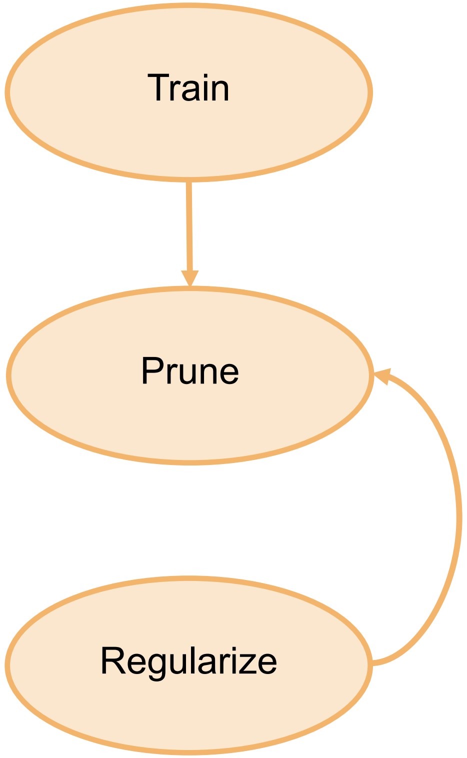
Pruning 101



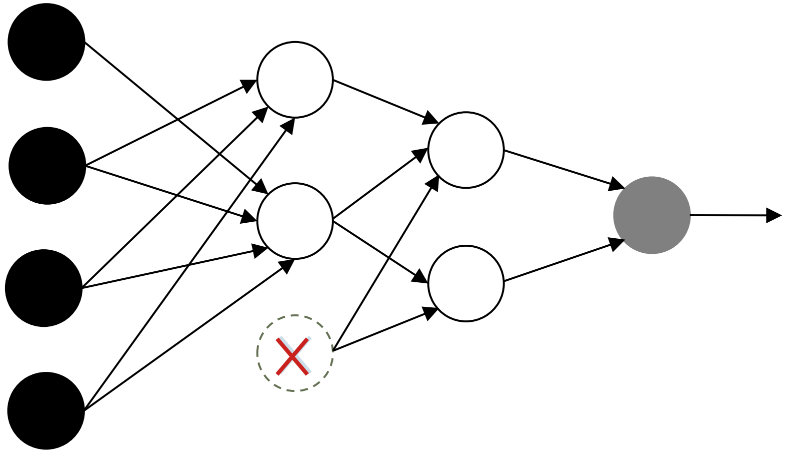
- Parameters are randomy initialized
- Parameters are updated then trained with standard gradient descent until performance is achieved (training stage)
- Parameters below threshold T are removed, pruning connections (parameter sparsification)
- Neurons without input arcs input are pruned from the network (neuron sparsification) ->Degrades network performance



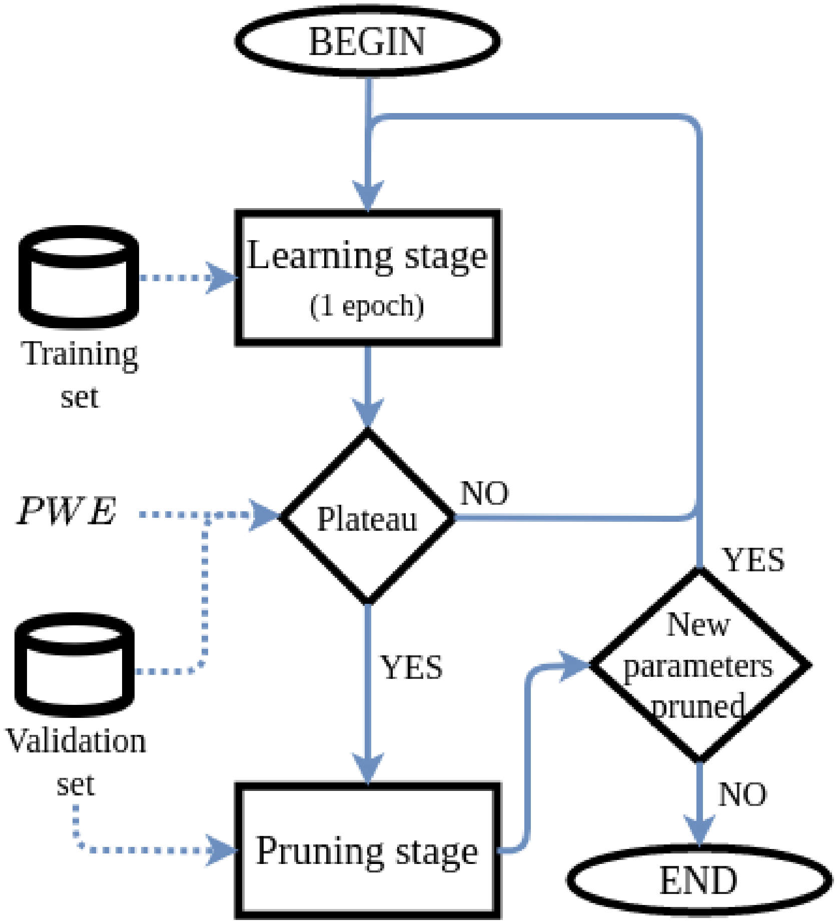
Pruning 101



- Parameters are randomy initialized
- Parameters are updated then trained with standard gradient descent until performance is achieved (training stage)
- Parameters below threshold T are removed, pruning connections (parameter sparsification)
- Neurons without input arcs input are pruned from the network (neuron sparsification) ->Degrades network performance
- Fine-tune the model, recovering the performance and iteratively prune again.



Pruning 101


BEGIN
Learning stage
(1 epoch)
Training set
PWE
Plateau
NO
YES
Validation set
Pruning stage
YES
New parameters pruned
NO
END

The elephant in the room

- Objective: reach the highest sparsity with no task-related performance degradation.
- The most effective approaches are also the ones **more computationally intensive**
  - multiple trainings are required for training one sparse model.
  - One-shot (or few-shot) pruning approaches are in general worse.

- New challenge: achieve sparsity with less computation at training time.

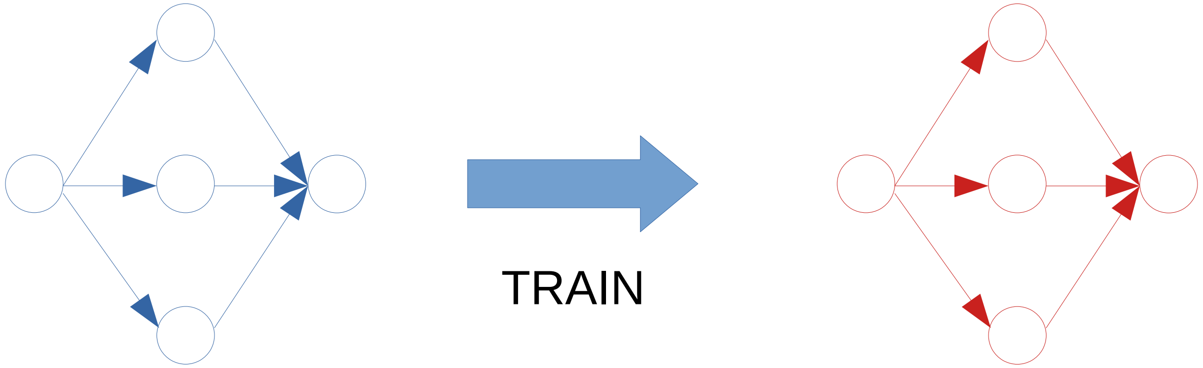
The lottery ticket hypothesis

“A randomly-initialized, dense neural network contains a sub-network that is initialized such that, when trained in isolation, it can match the test accuracy of the original network after training for at most the same number of iterations.” [Frankle and Carbin, 2019]

- The sub-network exists already at initialization: if it can be found, a lot of computation can be saved.
  - Computation for training the model (less parameters to train)
  - Computation for pruning (no iterative pruning anymore, just pruning at initialization, “zero-shot” pruning)

How to identify the lottery winners?

How to identify the lottery winners?



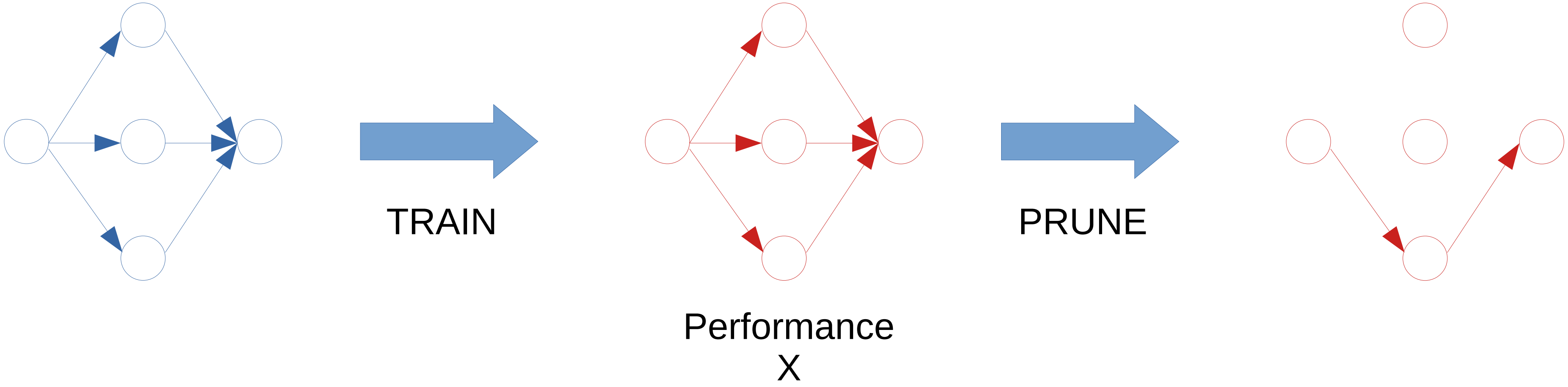
TRAIN

Performance
X

BEFORE TRANING

AFTER TRAINING

How to identify the lottery winners?



TRAIN

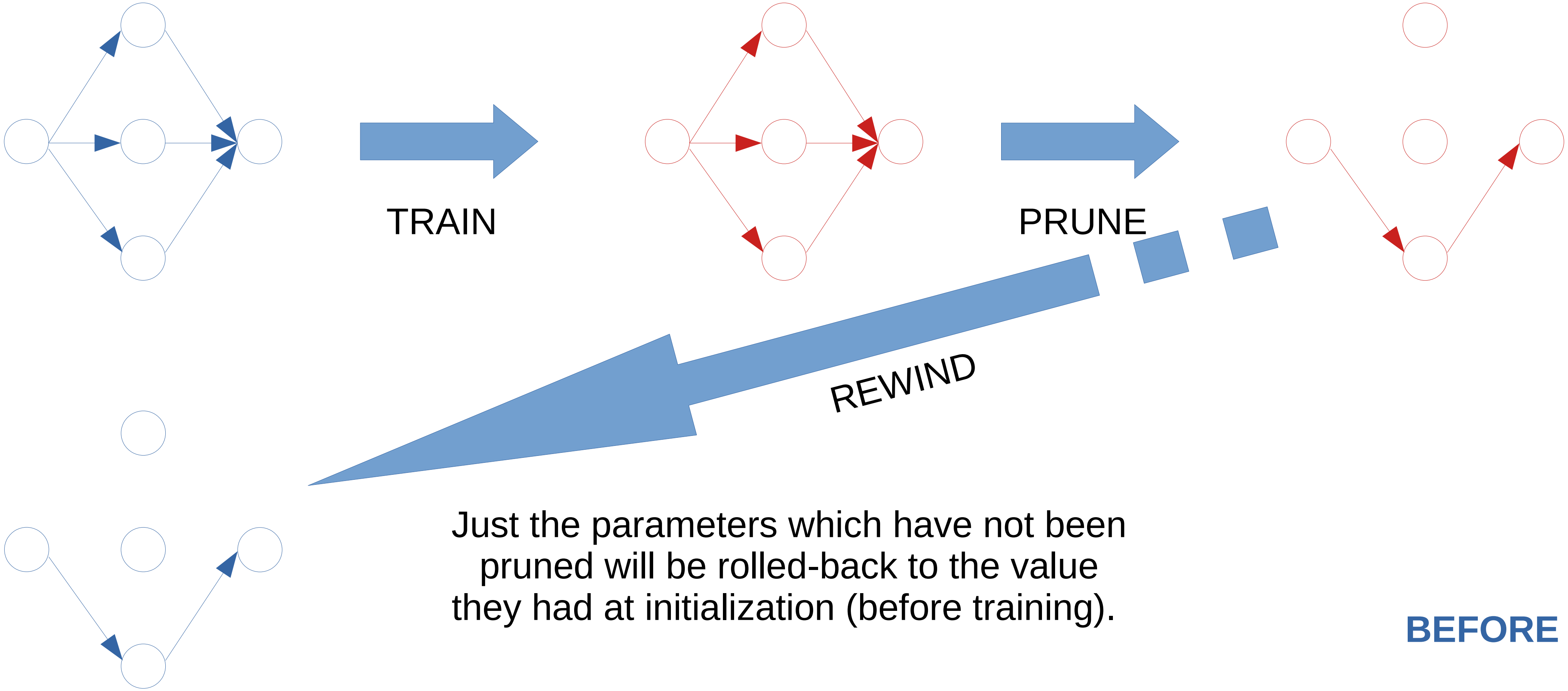
PRUNE

Performance
X

BEFORE TRANING

AFTER TRAINING

How to identify the lottery winners?

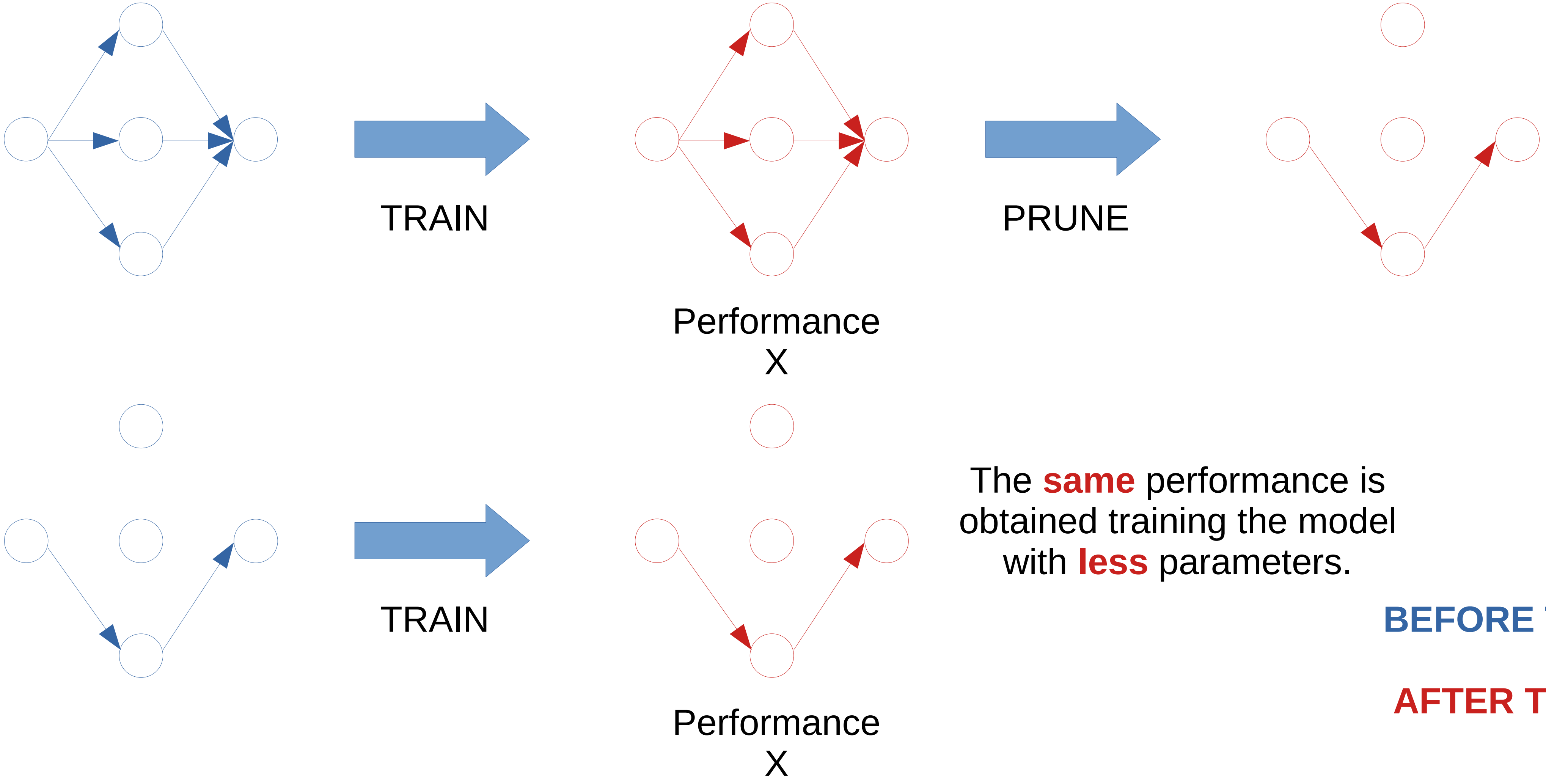


Just the parameters which have not been pruned will be rolled-back to the value they had at initialization (before training).

BEFORE TRANING

AFTER TRAINING

How to identify the lottery winners?

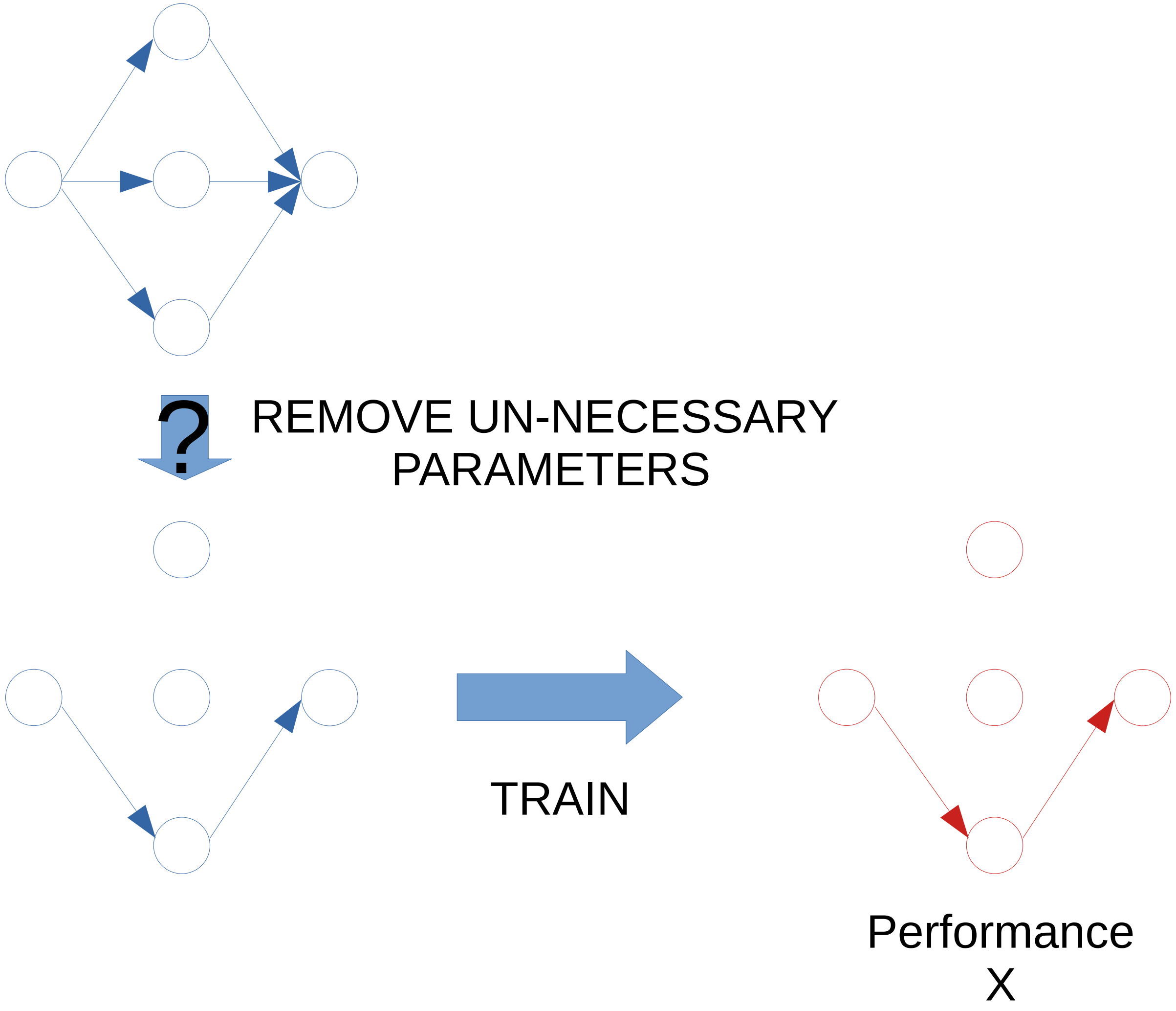


The **same** performance is obtained training the model with **less** parameters.

BEFORE TRANING

AFTER TRAINING

What we would like!



?

REMOVE UN-NECESSARY
PARAMETERS

TRAIN

Performance
X

The **same** performance is
obtained training the model
with **less** parameters.

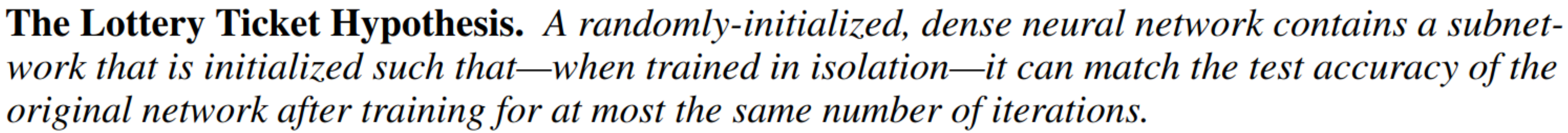
BEFORE TRANING

AFTER TRAINING

How to identify the lottery winners efficiently?

- The original work provides evidence of the existence of the willing tickets, not proposing the efficient algorithm…
  - A model is trained, and depending on the magnitude/variation of the parameters, pruned.
  - The model is rewound to the initial state, but with less parameters, and train/pruning is again repeated…
- Still a lot of research on making the search of the winning ticket efficient is needed!

Lottery winners at initialization?



The Lottery Ticket Hypothesis. *A randomly-initialized, dense neural network contains a subnetwork that is initialized such that—when trained in isolation—it can match the test accuracy of the original network after training for at most the same number of iterations.*

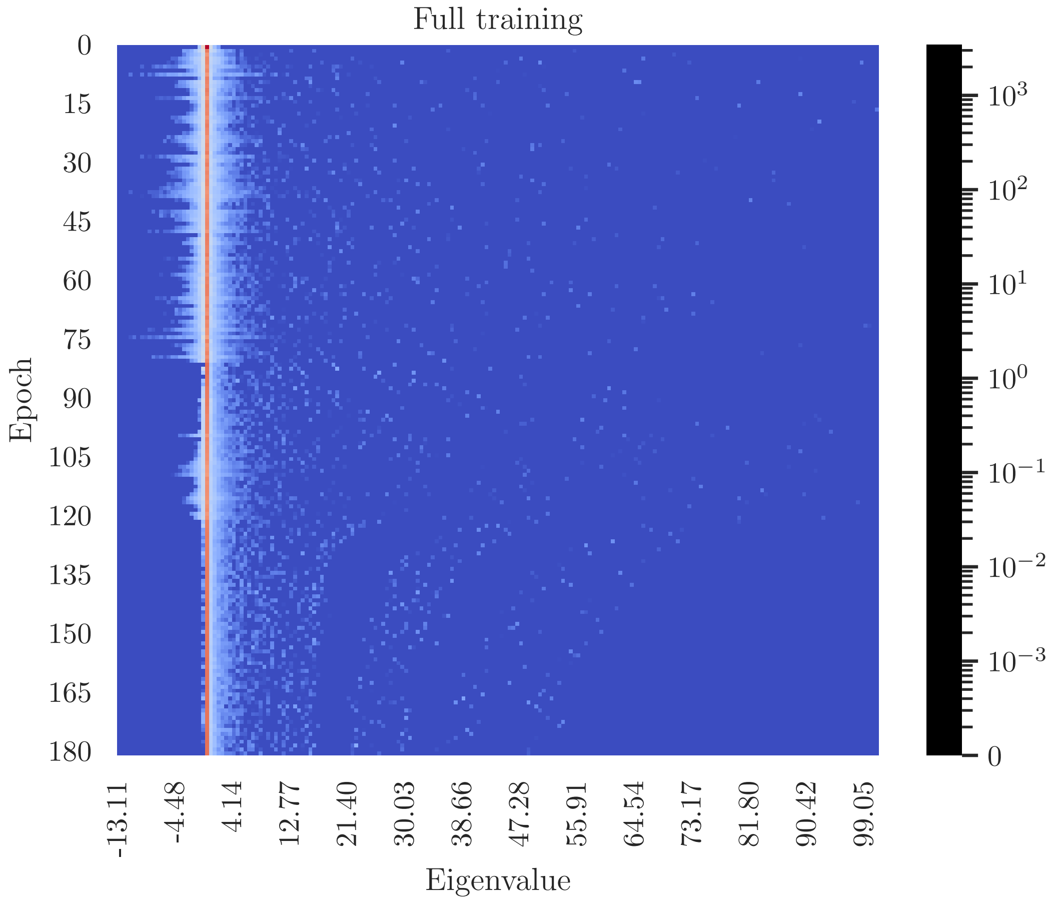
Finding these subnetworks **before or early in the train** can reduce the training cost.

Unfortunately this is an **hard task**:

1. Training a sparse network leads to **subpar results** [Evci et al., 2019]
2. Early tickets are **unstable** [Frankle et al., 2020]
3. The result is **similar to magnitude pruning** applied at the end of the training [Frankle et al., 2021]
4. If rewinding techniques are used, it can **require more computation** than just train the original model

Lottery winners at initialization?

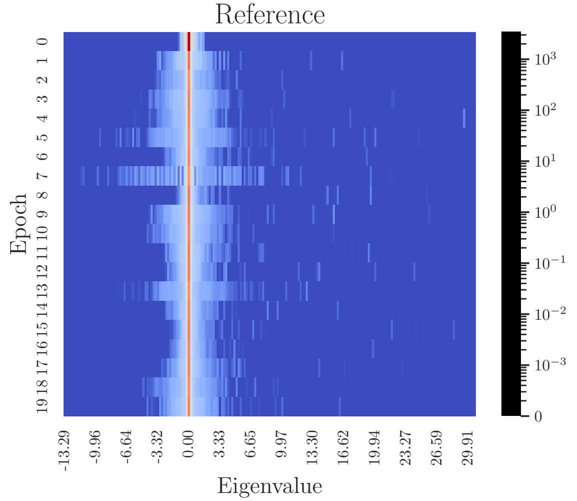
- Case study: ResNet-32 trained on CIFAR-10
  - SGD with momentum 0.9
  - Batch size 100
  - Initial learning rate: 0.1, decayed by factor 10 at epochs 80 and 120
- We visualize the eigenvalues of the hessian for the full training set (50k images), computed using the `PyHessian` library.
- Very complex loss landscape with high lr, which tends to be more and more “convex” as the lr decreases.

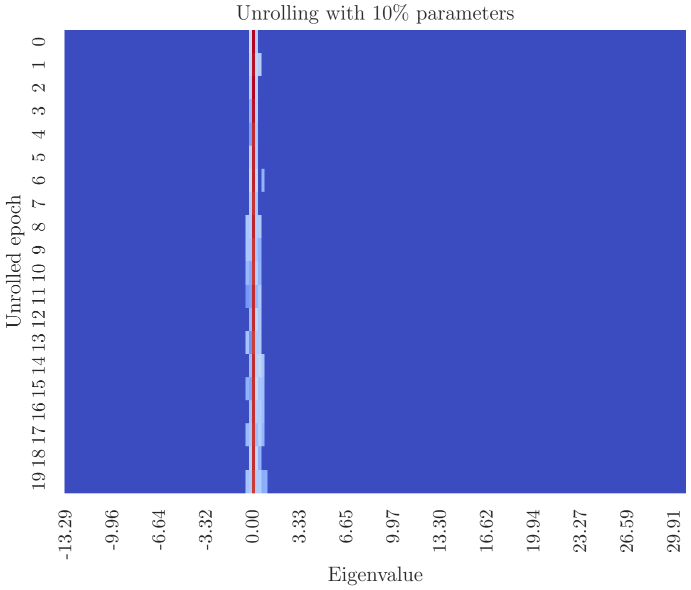


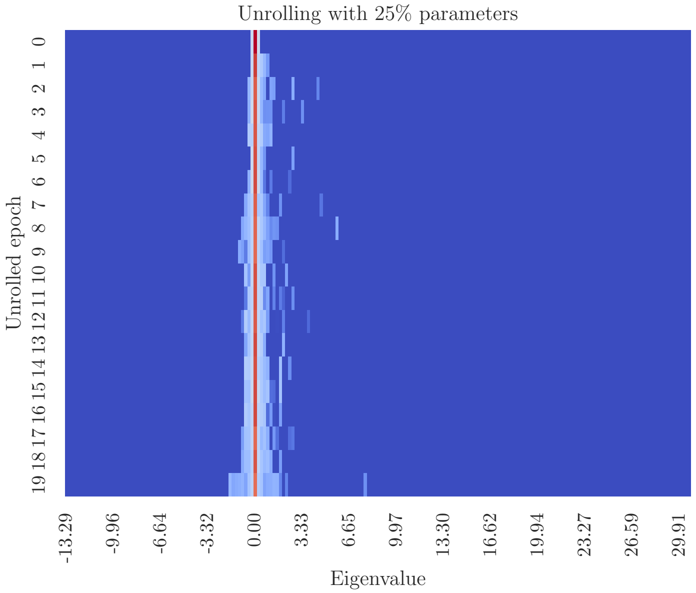
E. Tartaglione, "The Rise of the Lottery Heroes: Why Zero-Shot Pruning is Hard," 2022 IEEE International Conference on Image Processing (ICIP), 2022, pp. 2361-2365, doi: 10.1109/ICIP46576.2022.9897223.

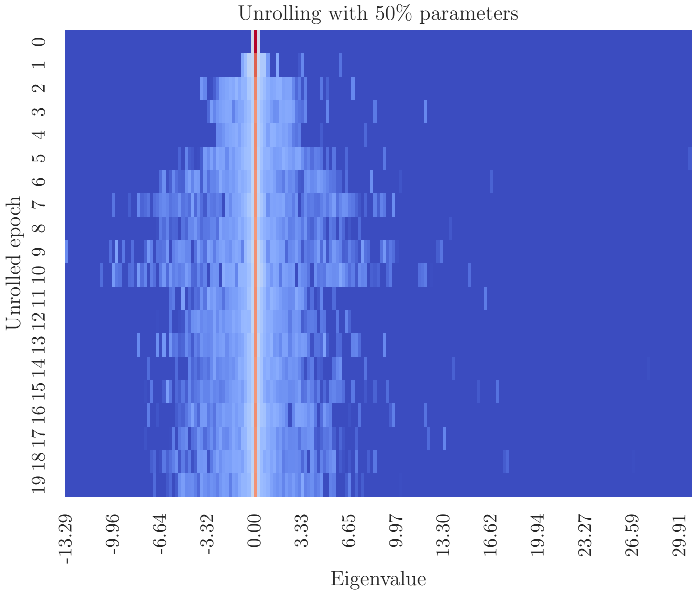
Lottery winners at initialization? (II)

- We compare the first 20 epochs with unrolling at many pruning rates.
- The landscape changes dramatically, and training can not match good performance!


Reference


Unrolling with 10% parameters


Unrolling with 25% parameters


Unrolling with 50% parameters

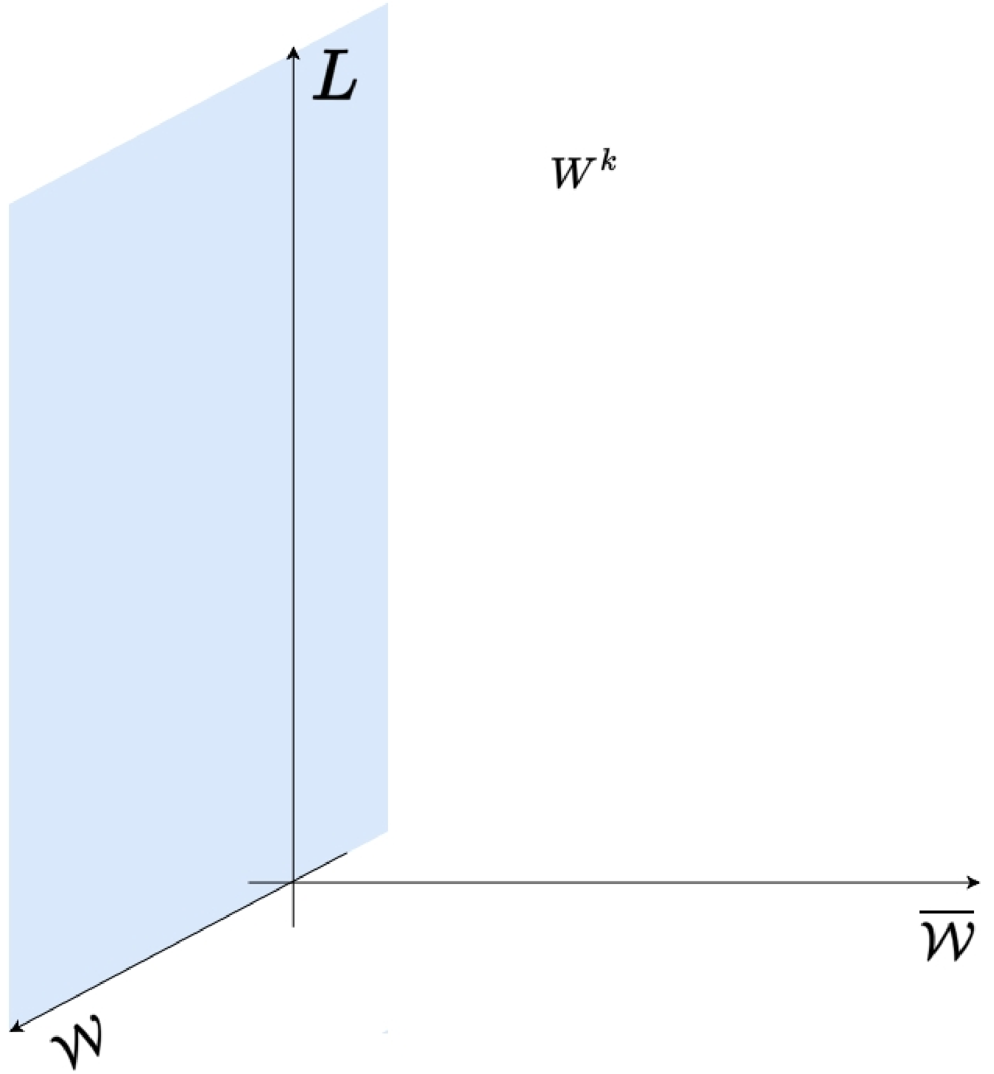
E. Tartaglione, "The Rise of the Lottery Heroes: Why Zero-Shot Pruning is Hard," 2022 IEEE International Conference on Image Processing (ICIP), 2022, pp. 2361-2365, doi: 10.1109/ICIP46576.2022.9897223.

Why is it so hard?

Let W^k be the initialization of a model.

We want to prune a subset of its parameters (namely, $\overline{\mathcal{W}}$).

This means that a subset $\mathcal{W}$ of parameters survives, and that we project in the blue hyper-plane.

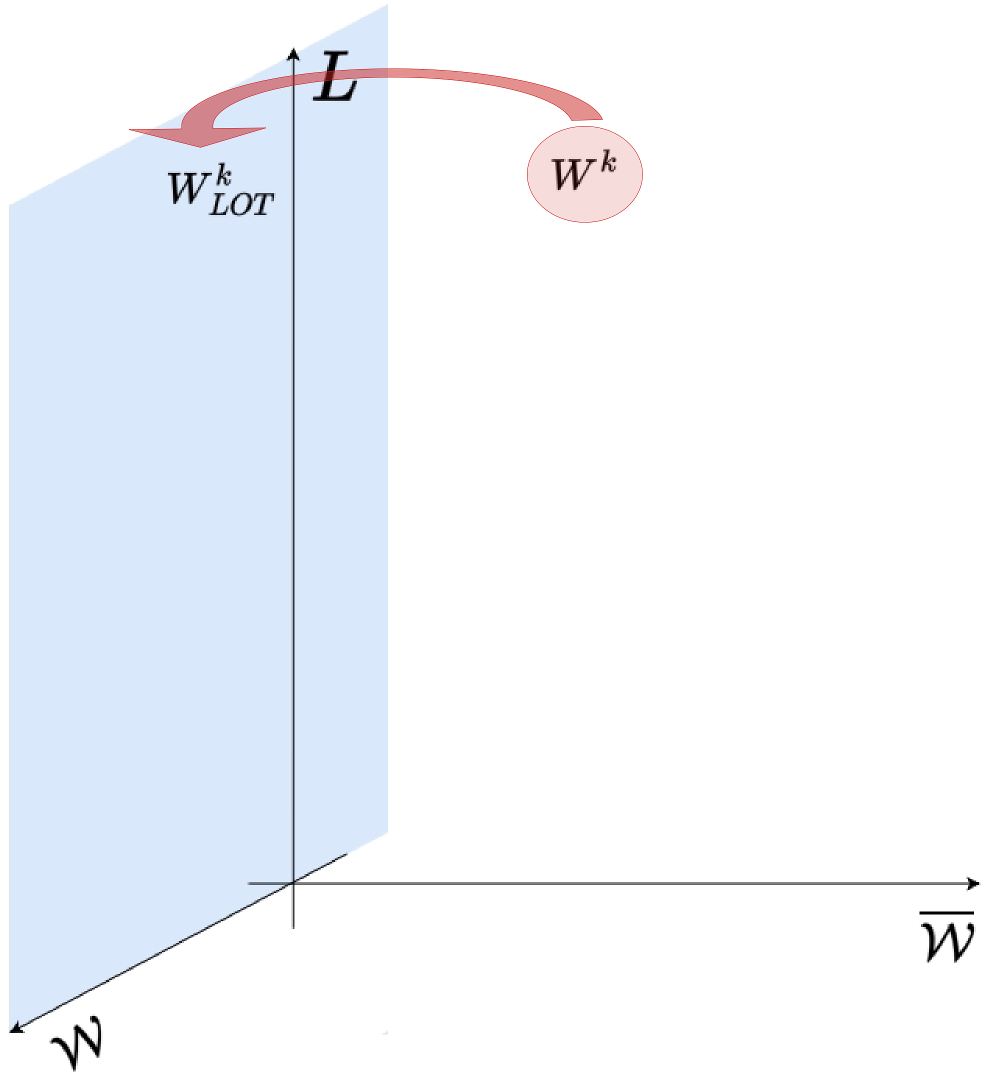


E. Tartaglione, "The Rise of the Lottery Heroes: Why Zero-Shot Pruning is Hard," 2022 IEEE International Conference on Image Processing (ICIP), 2022, pp. 2361-2365, doi: 10.1109/ICIP46576.2022.9897223.

Why is it so hard? (II)

We project in the subspace, finding the configuration W^k_{LOT}.

- When we project to a subspace, the loss landscape can change **drastically**!
- We need to perform the optimization in the blue subspace only!



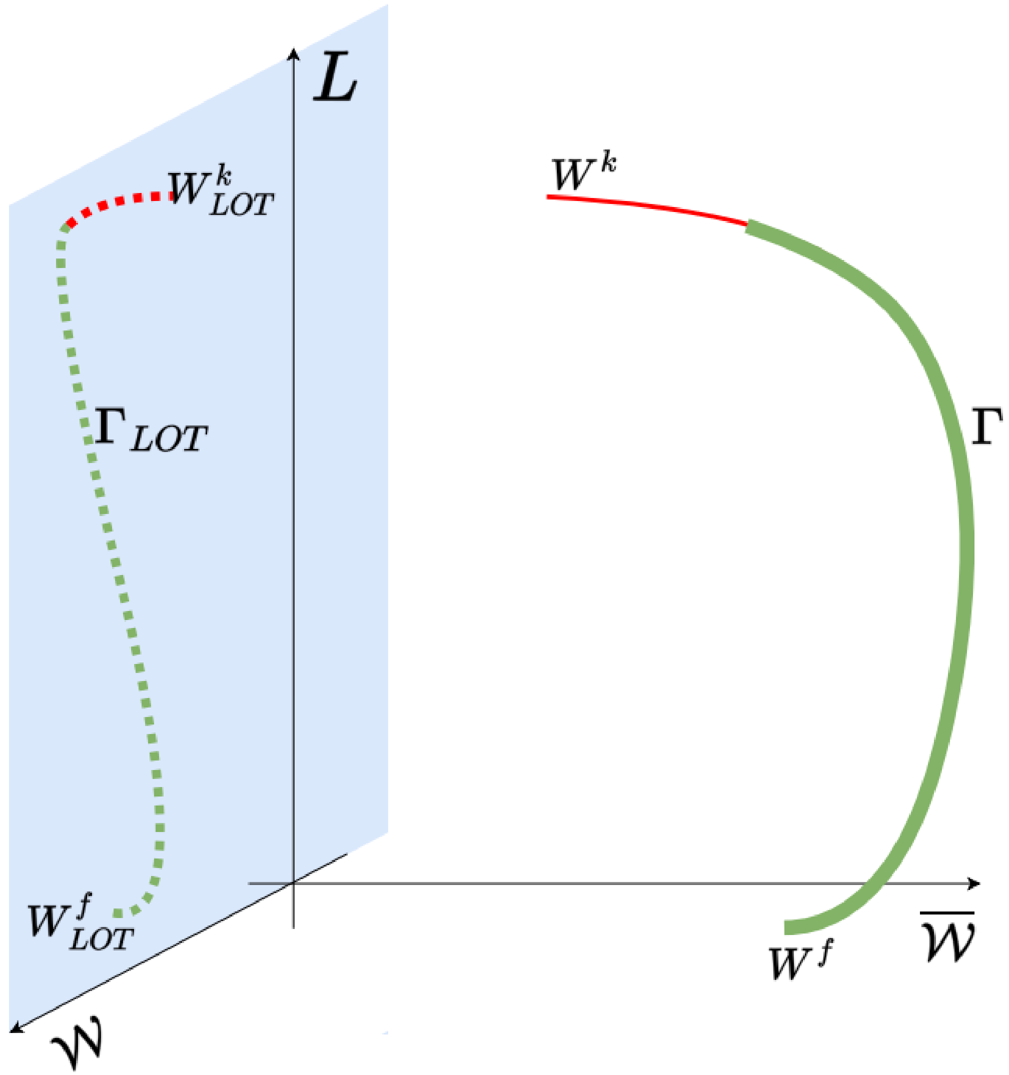
E. Tartaglione, "The Rise of the Lottery Heroes: Why Zero-Shot Pruning is Hard," 2022 IEEE International Conference on Image Processing (ICIP), 2022, pp. 2361-2365, doi: 10.1109/ICIP46576.2022.9897223.

Why is it so hard? (III)

Despite an optimization path, leading to a comparable loss as W^f might exist, it can be very difficult to find.

Empirically, it becomes more and more difficult to find as we prune more and more parameters.

In a **few-shot** (or even zero-shot) scenario, it becomes **extremely difficult** to find the **winning tickets**!



E. Tartaglione, "The Rise of the Lottery Heroes: Why Zero-Shot Pruning is Hard," 2022 IEEE International Conference on Image Processing (ICIP), 2022, pp. 2361-2365, doi: 10.1109/ICIP46576.2022.9897223.

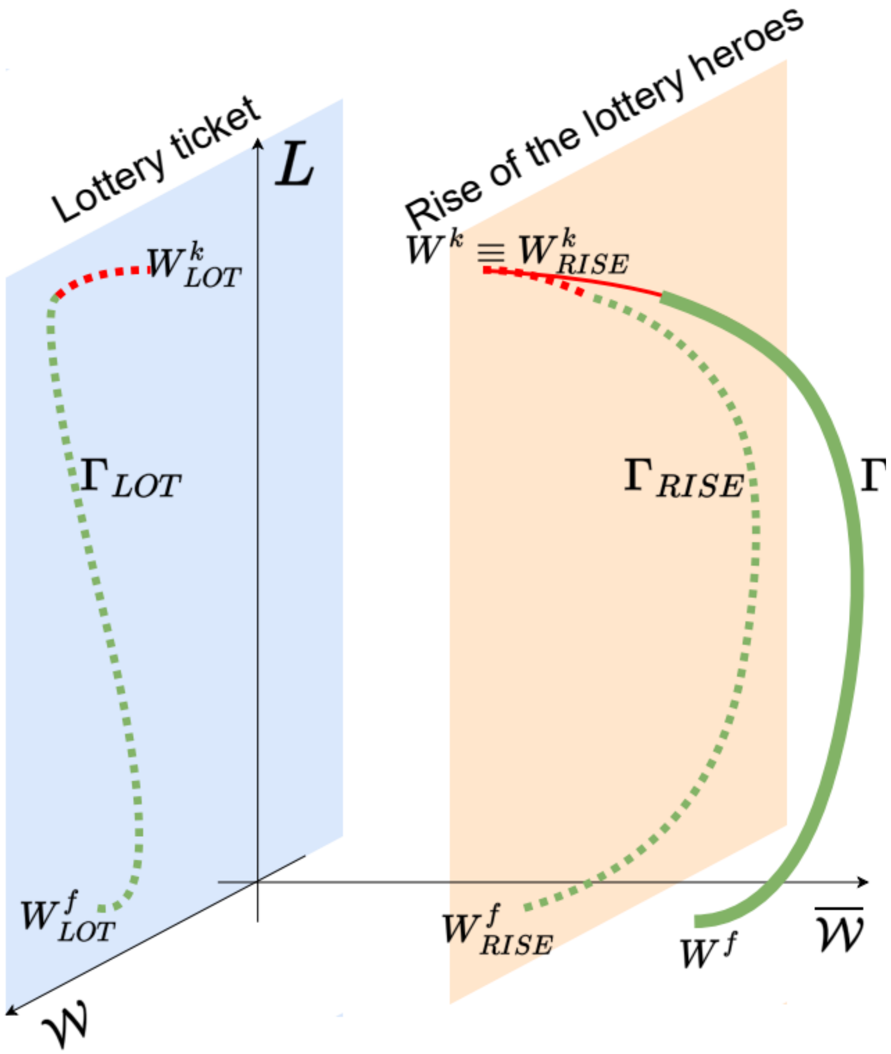
The rise of the lottery heroes

ICIP2022 BORDEAUX

What if we **do NOT project** (hence, no perturbation in the loss landscape at initialization due to the pruning)?

Instead of setting the parameters to zero, we simply do not update them all along the training!

Hence, we constrain the optimization problem in the orange subspace.



E. Tartaglione, "The Rise of the Lottery Heroes: Why Zero-Shot Pruning is Hard," 2022 IEEE International Conference on Image Processing (ICIP), 2022, pp. 2 361-2365, doi: 10.1109/ICIP46576.2022.9897223.

Are we pruning here?

E. Tartaglione, "The Rise of the Lottery Heroes: Why Zero-Shot Pruning is Hard," 2022 IEEE International Conference on Image Processing (ICIP), 2022, pp. 2
361-2365, doi: 10.1109/ICIP46576.2022.9897223.

Are we pruning here?

We move from pruning parameters to pruning the update of parameters.

E. Tartaglione, "The Rise of the Lottery Heroes: Why Zero-Shot Pruning is Hard," 2022 IEEE International Conference on Image Processing (ICIP), 2022, pp. 2361-2365, doi: 10.1109/ICIP46576.2022.9897223.

Are we pruning here?

We move from pruning parameters to pruning the update of parameters.

Lottery ticket hypothesis:
finding a sub-network which, when trained in isolation, matches the performance of the full model. Very tiny networks can be found at considerable computational effort with iterative pruning + rewinding.

E. Tartaglione, "The Rise of the Lottery Heroes: Why Zero-Shot Pruning is Hard," 2022 IEEE International Conference on Image Processing (ICIP), 2022, pp. 2 361-2365, doi: 10.1109/ICIP46576.2022.9897223.

Are we pruning here?

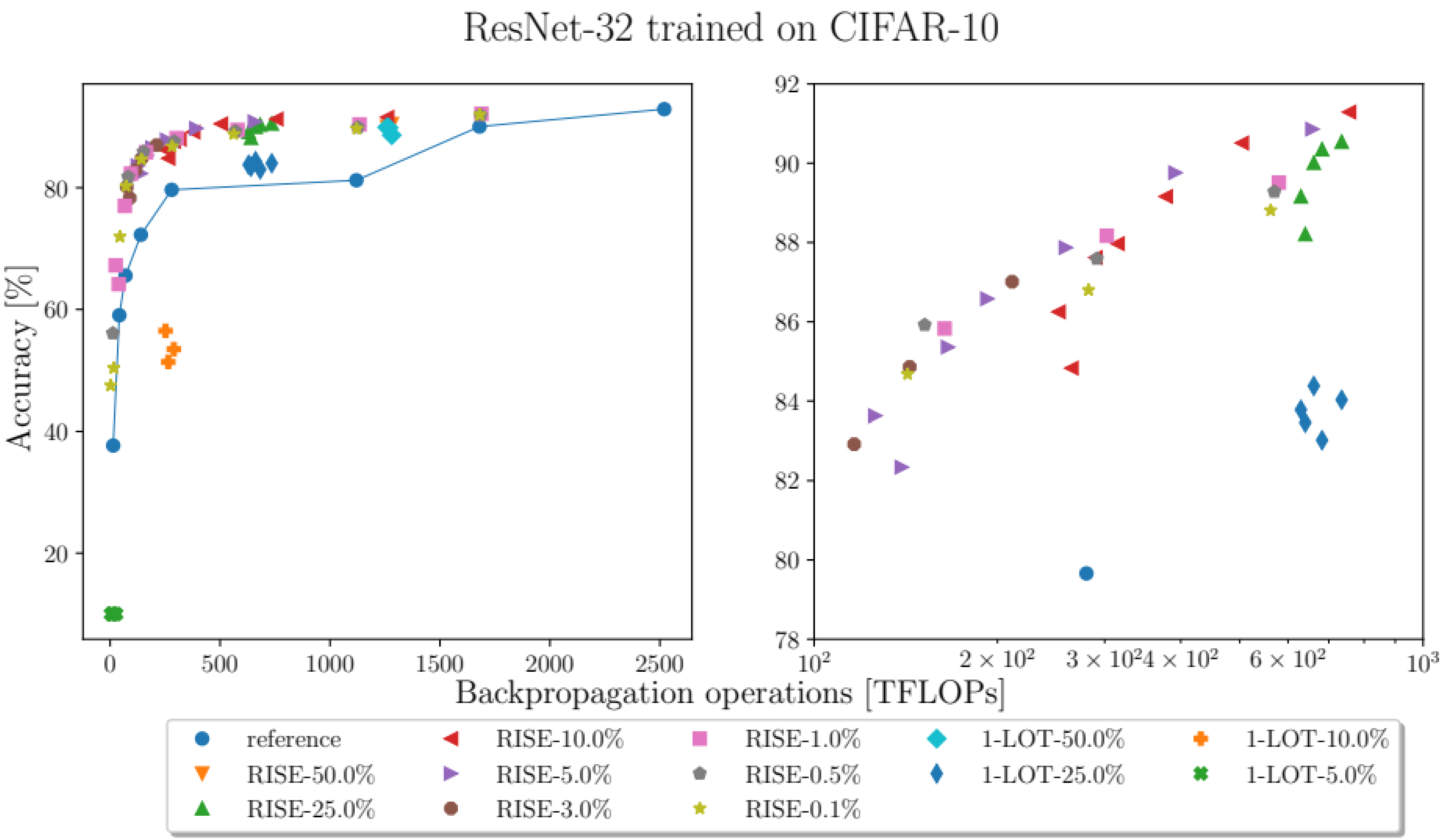
We move from pruning parameters to pruning the update of parameters.

Lottery ticket hypothesis:
finding a sub-network which, when trained in isolation, matches the performance of the full model. Very tiny networks can be found at considerable computational effort with iterative pruning + rewinding.

Rise of the lottery heroes:
finding a sub-network which needs to be updated to match the performance of the full model. Significant computational complexity can be saved in zero-shot scenarios (during the original training!)

E. Tartaglione, "The Rise of the Lottery Heroes: Why Zero-Shot Pruning is Hard," 2022 IEEE International Conference on Image Processing (ICIP), 2022, pp. 2361-2365, doi: 10.1109/ICIP46576.2022.9897223.

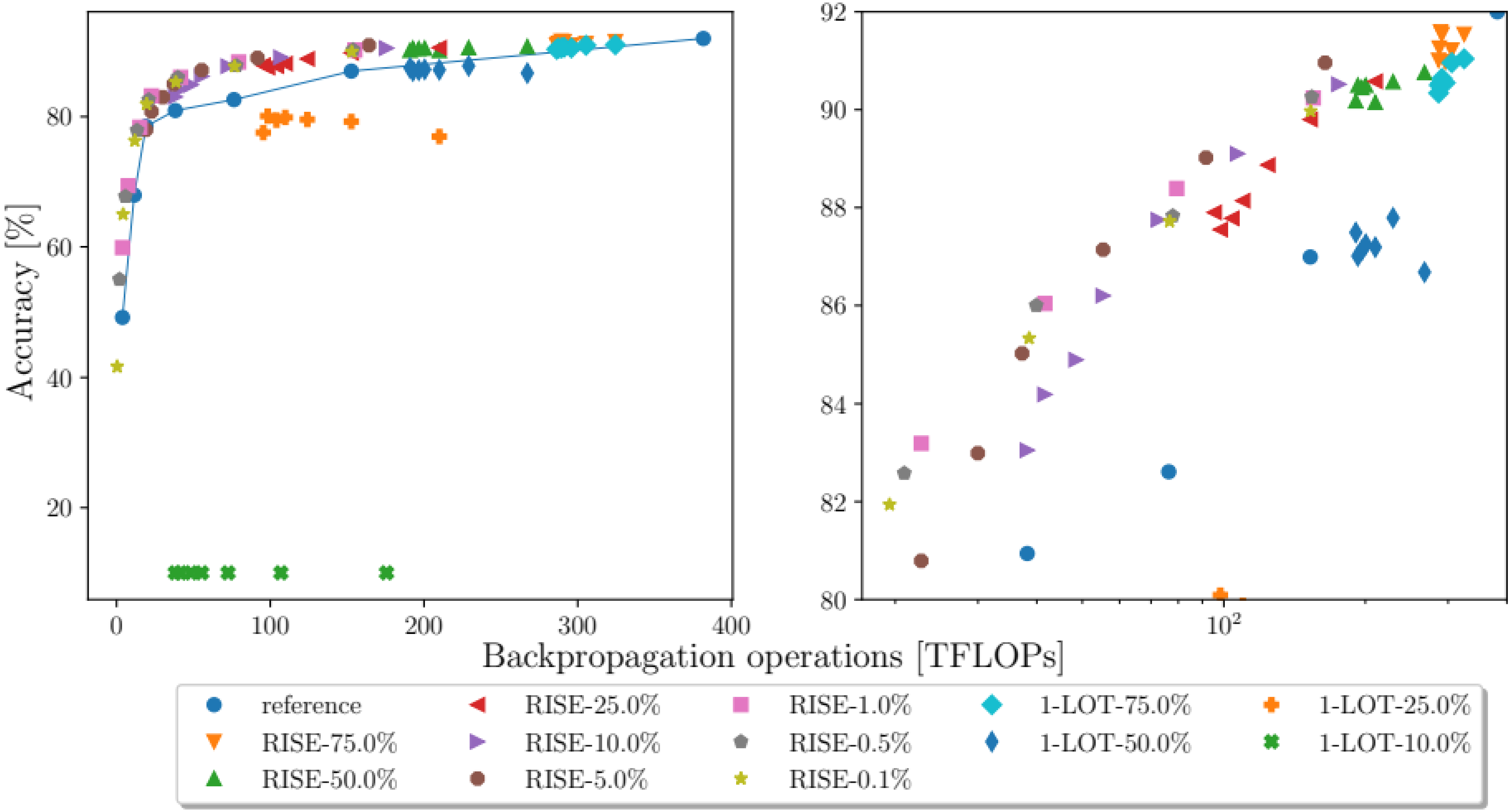
Experiments



E. Tartaglione, "The Rise of the Lottery Heroes: Why Zero-Shot Pruning is Hard," 2022 IEEE International Conference on Image Processing (ICIP), 2022, pp. 2361-2365, doi: 10.1109/ICIP46576.2022.9897223.

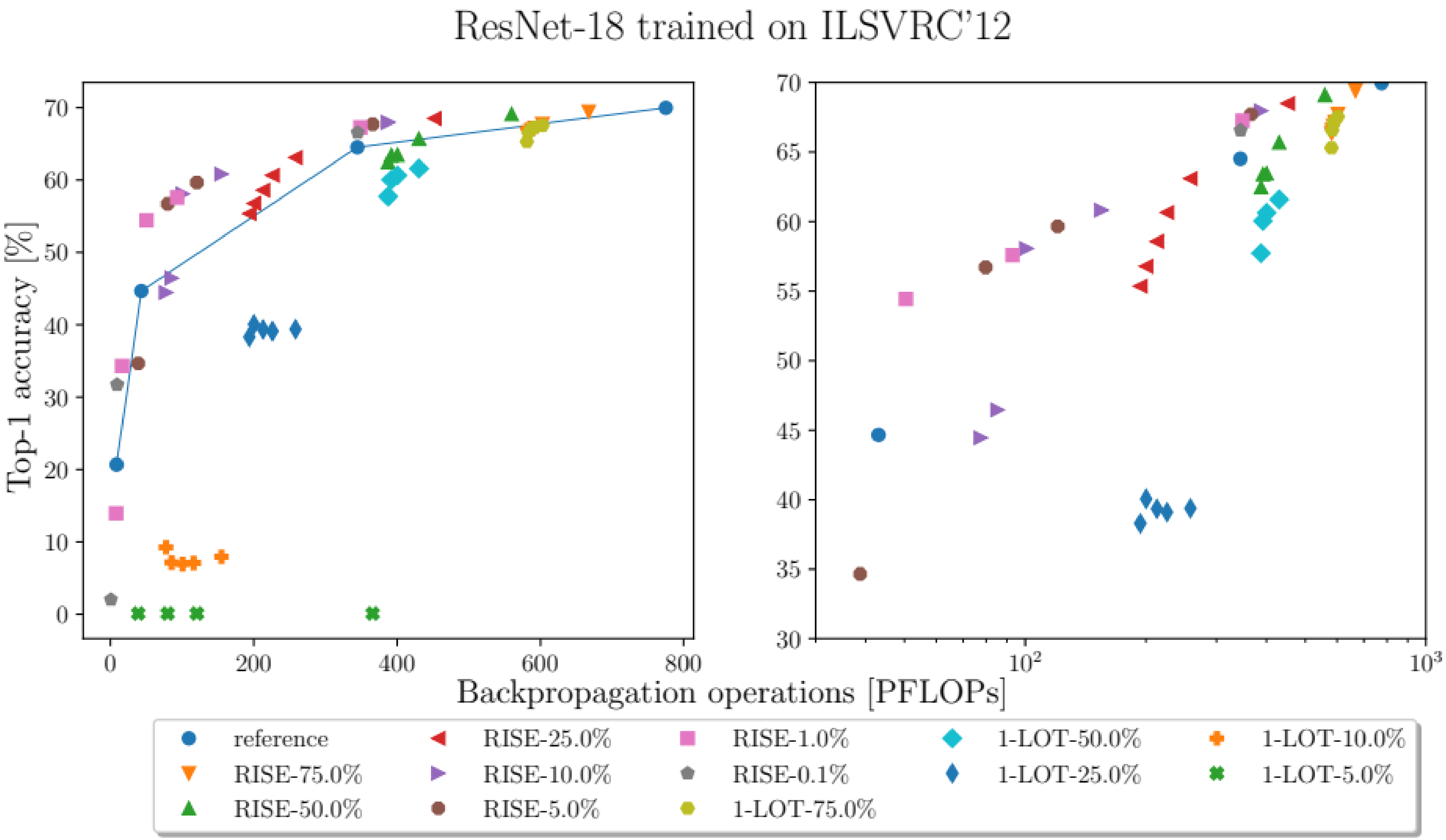
Experiments (II)

MobileNet-v3 small trained on CIFAR-10



E. Tartaglione, "The Rise of the Lottery Heroes: Why Zero-Shot Pruning is Hard," 2022 IEEE International Conference on Image Processing (ICIP), 2022, pp. 2 361-2365, doi: 10.1109/ICIP46576.2022.9897223.

Experiments (III)



E. Tartaglione, "The Rise of the Lottery Heroes: Why Zero-Shot Pruning is Hard," 2022 IEEE International Conference on Image Processing (ICIP), 2022, pp. 2361-2365, doi: 10.1109/ICIP46576.2022.9897223.

Interlude

- **Zero-shot pruning is very hard**: the pruning operation projects and constraints the optimization in sub-spaces with (potentially) very different loss landscape from the one at initialization.

- A much more promising direction is to "**prune**" **the back-propagation graph**, finding a small subset of parameters which need to be updated to reach the baseline performance.

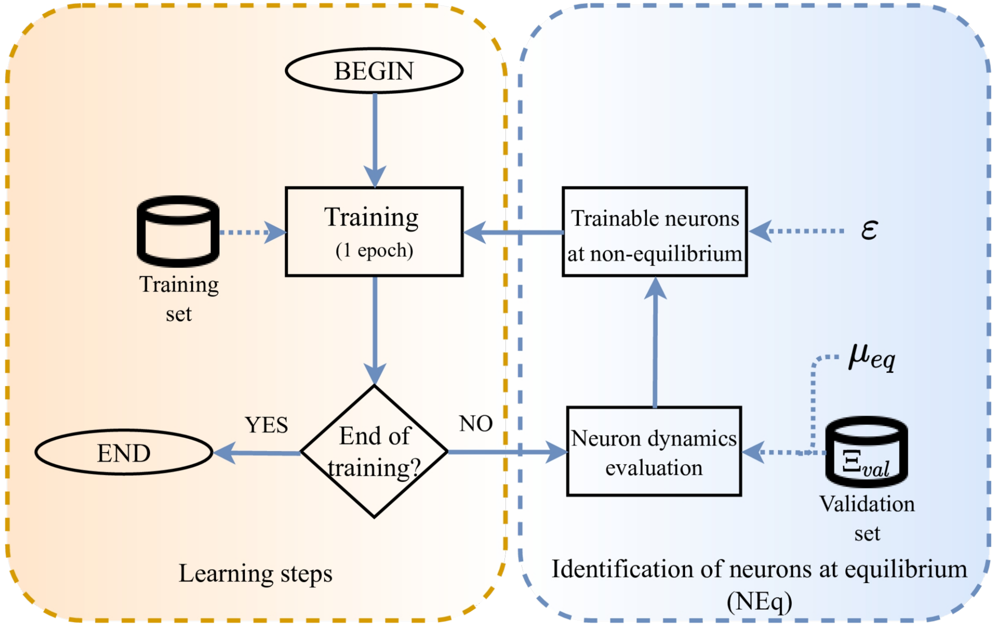
- A **trade-off** between the **complexity** deployed for **training** VS **final performance** is empirically observed!

- In this work, even though pruning after one training only, we use some information (virtually) inaccessible before training…

- Now comes the challenge to really gain: how do we know what to prune?

Neq – neurons at equilibrium

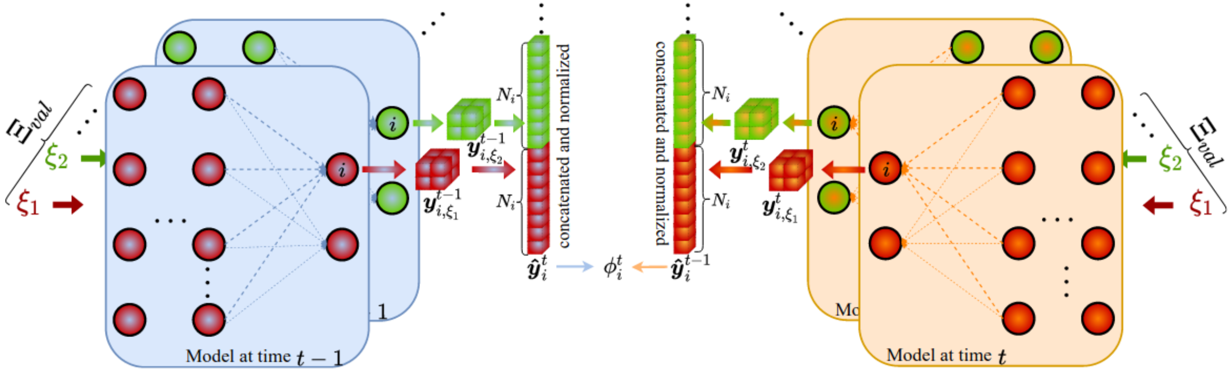
- **Not a pruning technique**.
- Focus on **entire neurons**.
- **No prior knowledge** required, is agnostic to the training strategy.
- **No rewinding** strategies.
- Each epoch, neurons at **equilibrium** are not updated.
- **Saves** computation at training time.
- How to evaluate the dynamics?



Andrea Bragagnolo, Enzo Tartaglione, Marco Grangetto. To update or not to update? Neurons at equilibrium in deep models. 36th Conference on Neural Information Processing Systems (NeurIPS 2022), Nov 2022, New Orleans, United States.

Neq – neurons at equilibrium

- We identify neurons whose output does not change on the validation set.
- We compute the cosine distance ϕ_i^t between the features extracted from the i-th neuron, computed on the same input (from the validation set).
- This value is in range [-1; +1]: +1 means it remains exactly the same.


Ξ_{val}
ξ_2
ξ_1
i
$\boldsymbol{y}_{i,\xi_2}^{t-1}$
$\boldsymbol{y}_{i,\xi_1}^{t-1}$
N_i
concatenated and normalized
$\hat{\boldsymbol{y}}_i^t$
ϕ_i^t
$\hat{\boldsymbol{y}}_i^{t-1}$
$\boldsymbol{y}_{i,\xi_2}^{t}$
$\boldsymbol{y}_{i,\xi_1}^{t}$
1
Mo
Model at time $t-1$
Model at time t

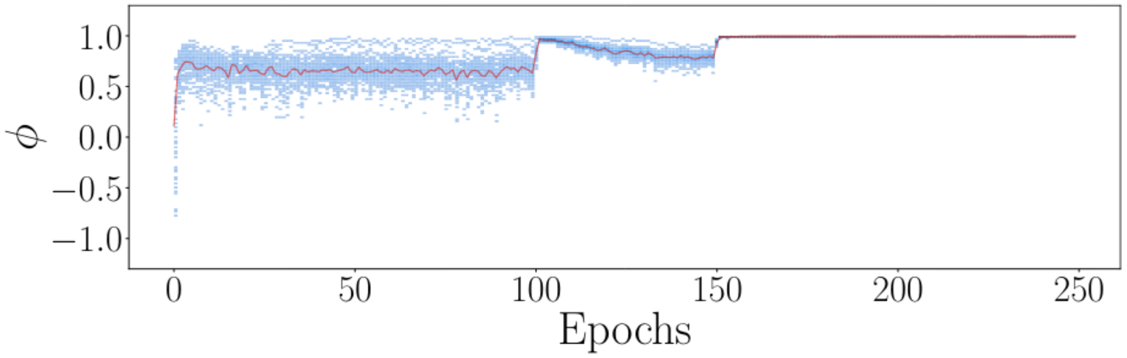
$$\phi_i^t = \sum_{\xi\in\Xi_{val}} \sum_{n=1}^{N_i} \hat{y}_{i,n,\xi}^t \cdot \hat{y}_{i,n,\xi}^{t-1}.$$

Andrea Bragagnolo, Enzo Tartaglione, Marco Grangetto. To update or not to update? Neurons at equilibrium in deep models. 36th Conference on Neural Information Processing Systems (NeurIPS 2022), Nov 2022, New Orleans, United States.

Neq – neurons at equilibrium

- Interestingly we observe that, in phases where the learning rate is very high (eg. until epoch 100 in fig.) the average value of ϕ_i^t is stable to a constant which is not +1…

- That's still equilibrium!
- We detect neurons at equilibrium if the variation of similarities $\Delta\phi_i^t$ is zero.
- Since it is important also to smoothen this variation term across noise in high learning rate regimes, we introduce a velocity term (similar to momentum in SGD) $v^t_{\Delta\phi_i}$



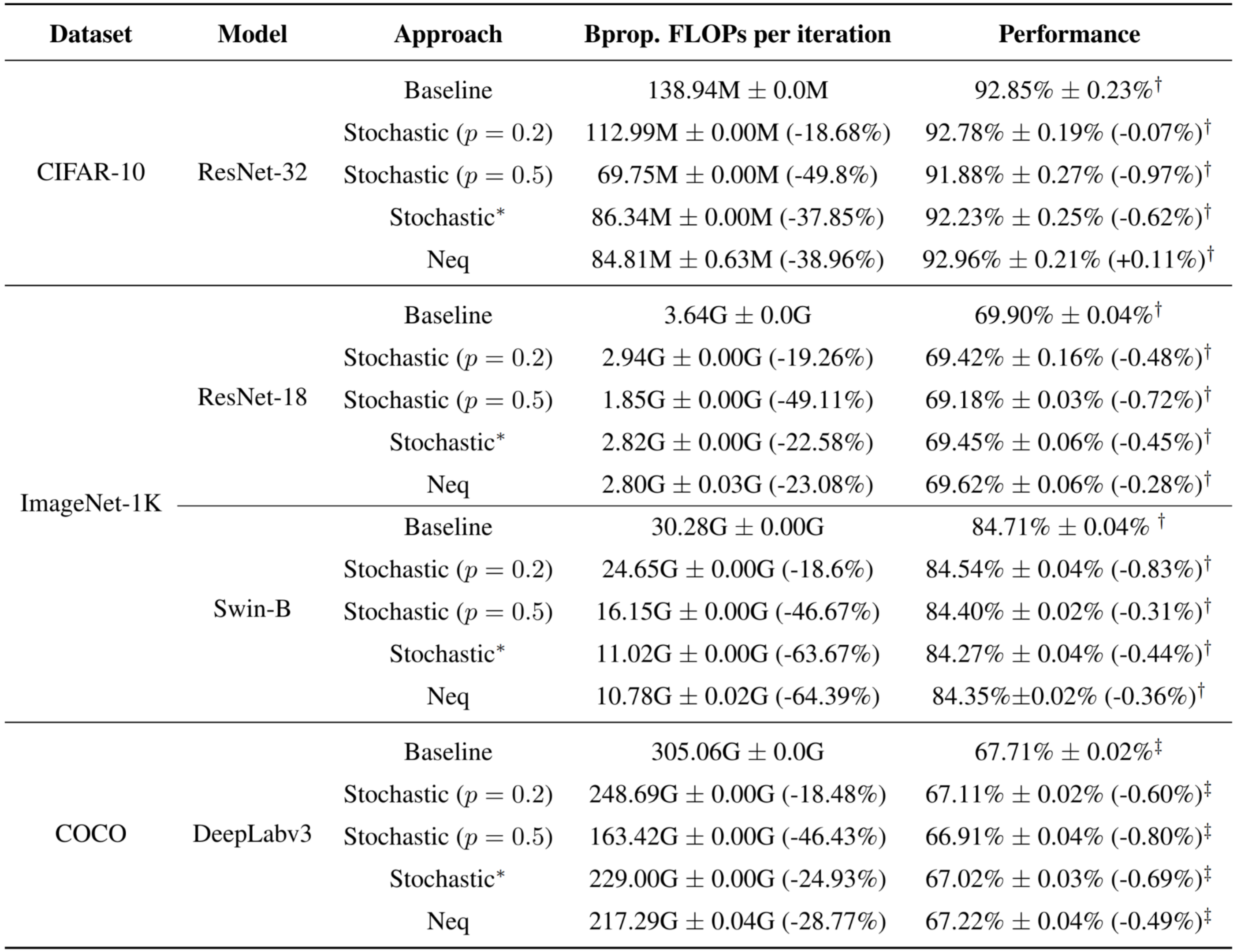
$$\phi_i^t = \sum_{\xi\in\Xi_{val}} \sum_{n=1}^{N_i} \hat{y}^t_{i,n,\xi} \cdot \hat{y}^{t-1}_{i,n,\xi}.$$

$$\Delta\phi_i^t = \phi_i^t - \phi_i^{t-1}.$$

$$v^t_{\Delta\phi_i} = \Delta\phi_i^t - \mu_{eq} v^{t-1}_{\Delta\phi_i},$$

Andrea Bragagnolo, Enzo Tartaglione, Marco Grangetto. To update or not to update? Neurons at equilibrium in deep models. 36th Conference on Neural Information Processing Systems (NeurIPS 2022), Nov 2022, New Orleans, United States.

Experiments



Dataset	Model	Approach	Bprop. FLOPs per iteration	Performance
CIFAR-10	ResNet-32	Baseline	138.94M ± 0.0M	92.85% ± 0.23%†
		Stochastic ($p = 0.2$)	112.99M ± 0.00M (-18.68%)	92.78% ± 0.19% (-0.07%)†
		Stochastic ($p = 0.5$)	69.75M ± 0.00M (-49.8%)	91.88% ± 0.27% (-0.97%)†
		Stochastic*	86.34M ± 0.00M (-37.85%)	92.23% ± 0.25% (-0.62%)†
		Neq	84.81M ± 0.63M (-38.96%)	92.96% ± 0.21% (+0.11%)†
ImageNet-1K	ResNet-18	Baseline	3.64G ± 0.0G	69.90% ± 0.04%†
		Stochastic ($p = 0.2$)	2.94G ± 0.00G (-19.26%)	69.42% ± 0.16% (-0.48%)†
		Stochastic ($p = 0.5$)	1.85G ± 0.00G (-49.11%)	69.18% ± 0.03% (-0.72%)†
		Stochastic*	2.82G ± 0.00G (-22.58%)	69.45% ± 0.06% (-0.45%)†
		Neq	2.80G ± 0.03G (-23.08%)	69.62% ± 0.06% (-0.28%)†
	Swin-B	Baseline	30.28G ± 0.00G	84.71% ± 0.04% †
		Stochastic ($p = 0.2$)	24.65G ± 0.00G (-18.6%)	84.54% ± 0.04% (-0.83%)†
		Stochastic ($p = 0.5$)	16.15G ± 0.00G (-46.67%)	84.40% ± 0.02% (-0.31%)†
		Stochastic*	11.02G ± 0.00G (-63.67%)	84.27% ± 0.04% (-0.44%)†
		Neq	10.78G ± 0.02G (-64.39%)	84.35%±0.02% (-0.36%)†
COCO	DeepLabv3	Baseline	305.06G ± 0.0G	67.71% ± 0.02%‡
		Stochastic ($p = 0.2$)	248.69G ± 0.00G (-18.48%)	67.11% ± 0.02% (-0.60%)‡
		Stochastic ($p = 0.5$)	163.42G ± 0.00G (-46.43%)	66.91% ± 0.04% (-0.80%)‡
		Stochastic*	229.00G ± 0.00G (-24.93%)	67.02% ± 0.03% (-0.69%)‡
		Neq	217.29G ± 0.04G (-28.77%)	67.22% ± 0.04% (-0.49%)‡

Evaluated on **different architecture, tasks** and **training policies**.

Compares:

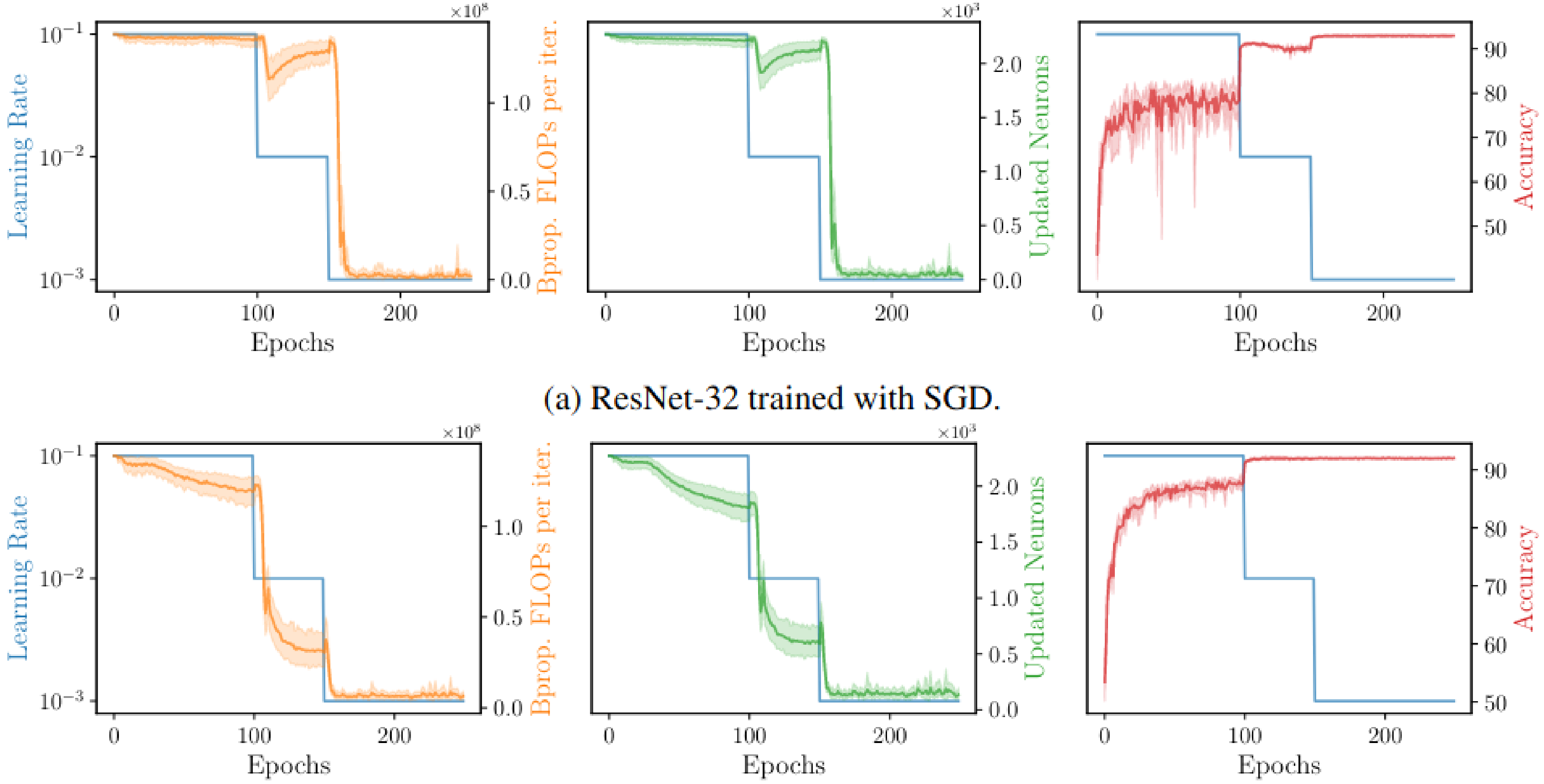
1. **Baseline:** vanilla training.
2. **Stochastic:** for every epoch a random (changing) amount of neurons is not updated.
3. **NEq:** our proposed approach.

NEq is able to **reduce the FLOPs requirement** for backpropagation without particular impact on the network's performance.

Andrea Bragagnolo, Enzo Tartaglione, Marco Grangetto. To update or not to update? Neurons at equilibrium in deep models. 36th Conference on Neural Information Processing Systems (NeurIPS 2022), Nov 2022, New Orleans, United States.

NEURAL INFORMATION PROCESSING SYSTEMS

Effectiveness dependent on the o ptimizer!

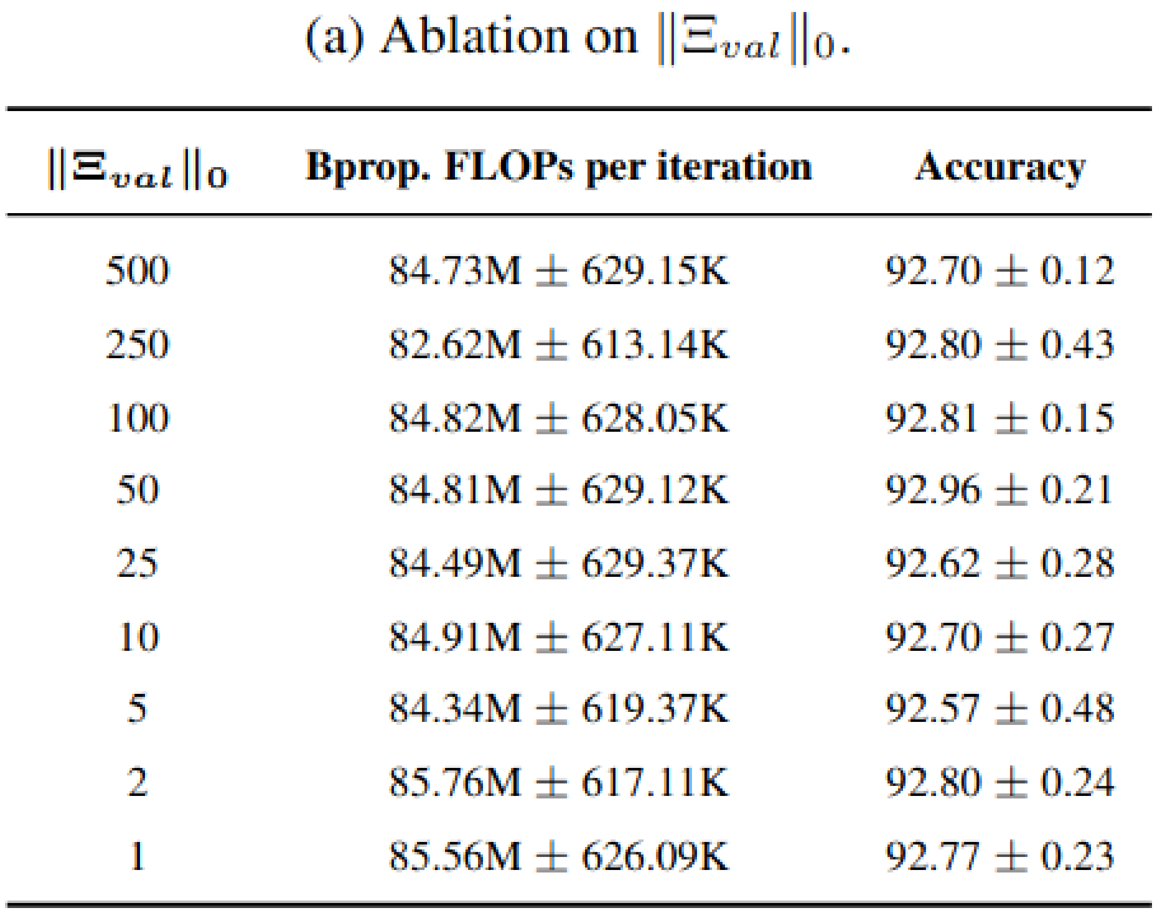


(a) ResNet-32 trained with SGD.

(b) ResNet-32 trained with Adam.

Andrea Bragagnolo, Enzo Tartaglione, Marco Grangetto. To update or not to update? Neurons at equilibrium in deep models. 36th Conference on Neural Information Processing Systems (NeurIPS 2022), Nov 2022, New Orleans, United States.

What about the size of the validation set?



(a) Ablation on $||\Xi_{val}||_0$.

$\lVert\Xi_{val}\rVert_0$	**Bprop. FLOPs per iteration**	**Accuracy**
500	84.73M ± 629.15K	92.70 ± 0.12
250	82.62M ± 613.14K	92.80 ± 0.43
100	84.82M ± 628.05K	92.81 ± 0.15
50	84.81M ± 629.12K	92.96 ± 0.21
25	84.49M ± 629.37K	92.62 ± 0.28
10	84.91M ± 627.11K	92.70 ± 0.27
5	84.34M ± 619.37K	92.57 ± 0.48
2	85.76M ± 617.11K	92.80 ± 0.24
1	85.56M ± 626.09K	92.77 ± 0.23

Andrea Bragagnolo, Enzo Tartaglione, Marco Grangetto. To update or not to update? Neurons at equilibrium in deep models. 36th Conference on Neural Information Processing Systems (NeurIPS 2022), Nov 2022, New Orleans, United States.

What are the current challenges?

- Pruning in order to push the **REAL ADVANTAGE** on the final device (hence, structured sparsity).

- Alternatively, Knowledge Distillation/NAS approaches can achieve very tiny models… but at which training cost?

- Currently, iterative algorithms achieve better results in terms of accuracy/compression than few-shot pruning… but at higher training cost.

- A lot of research is being done on **pruning at initialization** right now (towards saving of computation at training time)… some suggested to do so in transfer learning scenarios!

- The new learning approach is more oriented towards fine-tuning + pruning, starting from pre-trained models. New paradigms involve pruning from pre-trained architectures.

Q&A