How I Learned to Stop Worrying and Love Retraining

Max Zimmer, Christoph Spiegel, Sebastian Pokutta

Cooperation: TU Berlin, ZIB
Funding: Math+

Neural Network Pruning

Neural Network Pruning approaches remove weights from the network, inducing sparsity in the corresponding tensors. The resulting *sparse* models require less storage and FLOPS at inference, while maintaining similar performance to the over-parameterized *dense* model. Two main paradigms have emerged for training dense models from scratch towards sparsity:

- **Prune-during-training** approaches: Find (almost) sparse model at the end of training
  - Are **pruning-stable**: The final 'hard' pruning step results in negligible accuracy loss, eliminating the need for retraining.
  - Require strong implicit bias towards sparsity (e.g., gradual pruning, regularization techniques).
- **Prune-after-training** approaches: Prune following standard training for T epochs, retrain for T_{rt} to compensate for losses.
  - Are **pruning-instable**: Most of the performance is lost at pruning and needs to be recovered during retraining.
  - Example: *Iterative Magnitude Pruning* (IMP) (Han et al., 2015): Iteratively remove low-magnitude weights and retrain.
- **Focus of our work**: The retraining phase of IMP and optimal learning rate schedules.

Our contributions

IMP is simple and easy to implement, however it is often claimed to be inferior to pruning-stable approaches:

- *It is computationally inefficient since it requires many expensive prune-retrain cycles.* Pruning-stable approaches find a sparse solution throughout regular training.
- *It achieves sub-optimal states since it employs 'hard' pruning instead of 'learning' the sparsity pattern throughout training.*

We challenge these commonly held beliefs by rethinking the retraining phase in the context of *Budgeted Training* (Li et al., 2020), i.e., the setting of training networks under a fixed iteration budget.

Our major contributions:

1. We demonstrate that the findings of Li et al. (2020) on Budgeted Training apply to IMP's retraining phase, providing further insights into the results presented by Renda et al. (2020) and Le and Hua (2021). We show that a simple linear learning rate schedule can significantly reduce IMP's runtime without sacrificing model performance.
2. We propose Adaptive Linear Learning Rate Restarting (ALLR), a novel method for selecting the initial value of the linear schedule without additional hyperparameter tuning. This approach considers both pruning effects and retraining time, outperforming previous retraining schedules across various learning tasks.
3. By integrating the initial dense training phase into the same budgeted training scheme, we develop Budgeted IMP (BIMP), a simple yet effective method that outperforms many pruning-stable approaches under the same number of training iterations.

Retraining schedules

Consider the learning rate schedule $(\eta_t)_{t\leq T}$ of the original training and let T_{rt} be the number of retraining epochs. Existing schedules:

- **FT** (Han et al., 2015): Use last learning rate η_T for all epochs.
- **LRW** (Renda et al., 2020): Rewind learning rate to epoch $T - T_{rt}$.
- **SLR** (Le and Hua, 2021): Proportional to the original schedule.
- **CLR** (Le and Hua, 2021): 1-cycle cosine decay schedule.

The empirical insights concerning Budgeted Training (Li et al., 2020) closely resemble the development and improvement of retraining schedules in the pruning context. We claim that retraining should first and foremost be considered under the aspect of Budgeted Training and that lessons derived in the latter, such as the budget-optimality of a linearly decaying schedule, are generally applicable in this context.

We propose the following schedules for each prune-retrain-cycle:

- **Linear Learning Rate Restarting (LLR)**: Linear decay from η_1 to zero after a short warm-up phase.
- **Adaptive Linear Learning Rate Restarting (ALLR)**: LLR, but discount the initial value η_1 by $d \in [0,1]$ to account for the available retraining time and the performance drop induced by pruning.

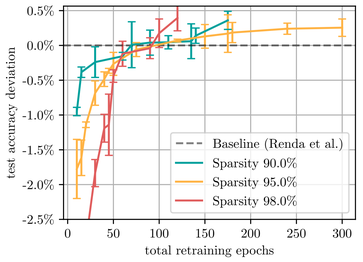
ALLR calculates the relative L_2-norm change between the weight vector θ and its pruned version θ^p after pruning a fraction $s \in (0,1]$ of θ:

$$d_1 = \frac{\|\theta - \theta^p\|_2}{\|\theta\|_2 \cdot \sqrt{s}} \in [0,1], \tag{1}$$

where normalization by $\sqrt{s}$ allows d_1 to attain the full range of values in $[0,1]$. We then compute $d_2 = T_{rt}/T$ to consider the retrain phase length and choose $d \cdot \eta_1$ as the initial learning rate for ALLR, with $d = \max(d_1, d_2)$.

Budgeting the retraining phase I

Figure 1: Envelope of IMP with ALLR for ResNet-56 on CIFAR-10 when treating the number of prune-retrain cycles and their individual length as hyperparameters. IMP is capable of achieving what has previously been considered its full potential with significantly less than the total number of retraining epochs budgeted for its iterative form: the baseline of Renda et al. (2020) requiring 2000, 2800, and 3600 epochs for sparsities 90%, 95% and 98%, respectively, is met after around 100 epochs of retraining.



Budgeting the retraining phase II

Table 1: ResNet-50 on ImageNet: Performance of the different learning rate translation schemes for One Shot IMP for target sparsities of 70%, 80% and 90% and retrain times of 2.22% (2 epochs), 5.55% (5 epochs) and 11.11% (10 epochs) of the initial training budget. The **first**, **second**, and **third** best values are highlighted.

ImageNet

	Model sparsity 70%			Model sparsity 80%			Model sparsity 90%		
Budget:	2.22%	5.55%	11.11%	2.22%	5.55%	11.11%	2.22%	5.55%	11.11%
FT	**73.51** ±0.04	73.98 ±0.04	74.44 ±0.11	70.45 ±0.20	71.81 ±0.11	72.68 ±0.07	56.75 ±0.01	61.60 ±0.30	64.61 ±0.21
LRW	73.50 ±0.04	**73.99** ±0.04	**74.45** ±0.11	70.45 ±0.20	71.82 ±0.12	72.67 ±0.07	56.75 ±0.01	61.61 ±0.30	64.60 ±0.23
SLR	70.93 ±0.01	72.58 ±0.03	73.69 ±0.11	70.48 ±0.04	72.37 ±0.02	73.44 ±0.18	67.19 ±0.23	69.45 ±0.01	70.80 ±0.09
CLR	72.22 ±0.09	73.58 ±0.08	**74.49** ±0.04	**71.96** ±0.09	**73.30** ±0.08	**74.24** ±0.08	**68.72** ±0.06	**70.60** ±0.15	**71.51** ±0.13
LLR (ours)	**72.39** ±0.13	**73.65** ±0.05	74.34 ±0.02	**72.07** ±0.09	**73.41** ±0.05	**74.23** ±0.10	**68.90** ±0.05	**70.48** ±0.01	**71.53** ±0.09
ALLR (ours)	**73.69** ±0.03	**74.37** ±0.05	**74.89** ±0.04	**72.96** ±0.15	**74.02** ±0.08	**74.71** ±0.04	**69.56** ±0.07	**71.19** ±0.01	**71.99** ±0.07

BIMP and comparison to pruning-stable methods

Given a budget of T epochs, Budgeted IMP (BIMP) simply trains a network for some $T_0 < T$ epochs using a linear schedule and then applies IMP with ALLR on the output for the remaining $T - T_0$ epochs. BIMP obtains a pruned model from scratch within the same budget as pruning-stable methods, while still maintaining the key characteristics of IMP, i.e.,

- we 'hard' prune and do not allow weights to recover in subsequent steps and
- we do not impose any particular additional implicit bias during either training or retraining.

Table 2: ResNet-50 on ImageNet: Comparison between BIMP and pruning-stable methods when training for goal sparsity levels as denoted in the main columns. We denote the images-per-second throughput during training. The **first**, **second**, and **third** best values are highlighted.

ImageNet

		Model sparsity 70%			Model sparsity 80%			Model sparsity 90%		
Method	# img/s	Accuracy	Speedup	Sparsity	Accuracy	Speedup	Sparsity	Accuracy	Speedup	Sparsity
BIMP (ours)	1454	**75.62** ±0.02	2 ±0.0	70.00 ±0.00	**75.08** ±0.16	3 ±0.0	80.00 ±0.00	**73.53** ±0.05	6 ±0.0	90.00 ±0.00
GMP	1425	74.62 ±0.08	2 ±0.0	70.00 ±0.00	74.19 ±0.17	4 ±0.0	80.00 ±0.00	72.80 ±0.03	7 ±0.1	90.00 ±0.00
GSM	1349	73.69 ±0.70	2 ±0.1	70.00 ±0.00	72.75 ±0.62	4 ±0.3	80.00 ±0.00	70.08 ±0.94	9 ±0.8	90.00 ±0.00
DPF	1456	**75.59** ±0.07	2 ±0.0	70.00 ±0.00	**75.30** ±0.02	3 ±0.0	80.00 ±0.00	**74.05** ±0.05	6 ±0.0	90.00 ±0.00
DNW	530	**75.60** ±0.01	2 ±0.0	70.00 ±0.00	**75.27** ±0.01	3 ±0.0	80.00 ±0.00	**74.29** ±0.03	5 ±0.1	90.00 ±0.00
LC	1436	75.03 ±0.20	2 ±0.0	70.00 ±0.00	73.87 ±0.62	3 ±0.0	80.00 ±0.00	67.57 ±2.71	5 ±0.0	90.00 ±0.00
STR	1396	70.66 ±0.13	3 ±0.0	75.34 ±0.01	70.70 ±0.13	4 ±0.0	80.93 ±0.00	70.13 ±0.01	8 ±0.0	90.00 ±0.00
DST	1219	74.63 ±0.22	4 ±0.1	70.00 ±0.00	73.16 ±0.11	6 ±0.1	80.00 ±0.00	71.35 ±0.09	13 ±0.4	90.00 ±0.00