One-Cycle Pruning: Pruning ConvNets Under a Tight Training Budget

Nathan Hubens[1,2]
ISIA Lab (UMONS)

Matei Mancas[1]
ISIA Lab (UMONS)

Bernard Gosselin[1]
ISIA Lab (UMONS)

Marius Preda[2]
Artemis (IP Paris)

Titus Zaharia[2]
Artemis (IP Paris)

[1] {nathan.hubens, matei.mancas, bernard.gosselin}@umons.ac.be
[2] {nathan.hubens, marius.preda, titus.zaharia}@telecom-sudparis.eu

Abstract

Introducing sparsity in a neural network has been an efficient way to reduce its complexity while keeping its performance almost intact. Most of the time, sparsity is introduced using a three-stage pipeline: 1) train the model to convergence, 2) prune the model according to some criterion, 3) fine-tune the pruned model to recover performance. The last two steps are often performed iteratively, leading to reasonable results but also to a time-consuming and complex process. In our work, we propose to get rid of the first step of the pipeline and to combine the two other steps in a single pruning-training cycle, allowing the model to jointly learn for the optimal weights while being pruned. We do this by introducing a novel pruning schedule, named One-Cycle Pruning, which starts pruning from the beginning of the training, and until its very end. Adopting such a schedule not only leads to better performing pruned models but also drastically reduces the training budget required to prune a model. Experiments are conducted on a variety of architectures (VGG-16 and ResNet-18) and datasets (CIFAR-10, CIFAR-100 and Caltech-101), and for relatively high sparsity values (80%, 90%, 95% of weights removed). Our results show that One-Cycle Pruning consistently outperforms commonly used pruning schedules such as One-Shot Pruning, Iterative Pruning and Automated Gradual Pruning, on a fixed training budget.

1 Introduction

Deep neural networks are able to achieve state-of-the-art results in a wide variety of domains, including computer vision, natural language processing and speech recognition. But such achievements imply important increase in budget required both at training and inference time. More specifically, model size, run time memory and a number of computing operations are all constraints that make modern neural networks challenging to deploy on resource-constrained environments such as mobile phones or embedded devices.

For that reason, neural network compression and acceleration have become active fields of research. Several researchers have been interested to create parameter-efficient architectures, by using low-rank approximations, parameters quantization, and neural network pruning.

Recent studies have also exhibited a particular characteristic of neural networks, called the *Lottery Ticket Hypothesis* (1), which suggests that, in regular neural network architectures, there exists a sub-network that can be trained to at least the same level of performance as the original one, in a

Preprint.

comparable training budget, as long as it starts from the same original conditions. Such a sub-network has thus "won" at the initialization lottery and can be preserved, while other parameters of the original network can be removed using pruning methods.

To prune a neural network, the most commonly used technique is the so-called Iterative Pruning (2), requiring several cycles of pruning and fine-tuning which leads to a lengthy process. In this work, we propose to adopt a novel pruning schedule applied directly from the start of the initial training phase. This scheduling function gradually prunes the network during the training phase, thus making training and pruning of a neural network a joint process. Contributions of our work are summarized as following:

- We propose One-Cycle Pruning, a novel pruning schedule with stable, thus generic parameters.
- The proposed pruning schedule is performed using a single training cycle, thus drastically reducing the required training budget.
- Experiments conducted on two architectures and three datasets show that the proposed schedule outperforms commonly used pruning schedules.

2 Related Work

Pruning techniques can differ in many aspects. The main points of differentiation are presented in this section.

Granularity. The granularity used for the pruning is often categorized into two groups: *unstructured*, or when the pruning focuses on removing individual weights in the network and that there is no intent to keep any structure in the filters (3; 4). Such a pruning method leads to sparse weight matrices, requiring dedicated hardware or software to take advantage of the speed and computation gains. To overcome this limitation, *structured* pruning was introduced (5; 6; 7), which takes care of removing complete blocks of weights such as vectors, kernels, or even convolution filters.

Criteria. Early work makes use of second-order approximation of the loss surface to remove parameters (3; 4). Other work has also explored the use of l_0 regularization (8) during the training or even the use of variational dropout (9). In addition to being more complex, those criteria often show to be less consistent across datasets and to lead to comparable or worse results than simple magnitude pruning, based on l_1-norm (10).

Scheduling. While early pruning methods cared about removing redundant weights in a single step, called one-shot pruning (5), the most adopted technique nowadays is to perform pruning iteratively (11; 12), starting from a pretrained network and performing several steps of pruning followed by fine-tuning. While having undeniably shown good results, such a pruning schedule is usually time consuming to obtain the final, pruned network (5). For this reason, some research has proposed alternatives to perform pruning during the training or even before it even started (13; 14).

Recently, some work has shown that the most critical phase in the training of a neural network happens during the very first iterations (15; 16) and that applying regularization after that initial transient phase has little effect on the final performance of the network (17). One thus should apply pruning early in the training to take advantage of its regularization effects but must do so very carefully to not irremediably damage the network during this brittle period (18; 19).

For those reasons, we propose a pruning schedule that is applied gradually during the training, and designed to be gentle during the very first iterations. This pruning schedule can be used for any granularity and any selection criteria, but in this work, we are focusing on unstructured pruning, *i.e.* removing individual weights, and we prune the weights having the lowest l_1-norms.

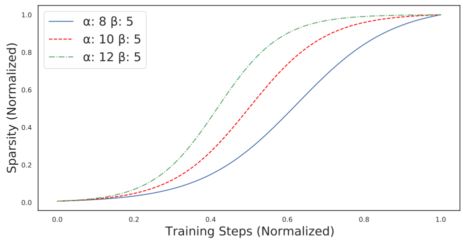
3 Proposed Method

The proposed method for pruning consists of starting from a dense network and inducing sparsity during the whole training phase. The idea is thus to make the network jointly optimize for a given task, taking the pruning constraints into account.

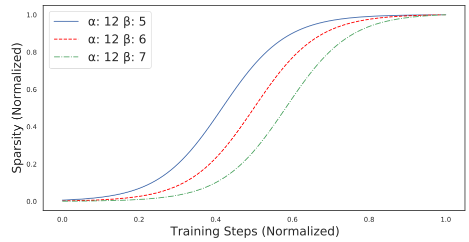
More precisely, we propose to induce sparsity in the network according to the following schedule:

$$s_t = s_i + (s_f - s_i) \cdot \frac{1 + e^{-\alpha + \beta}}{1 + e^{-\alpha t + \beta}} \tag{1}$$

with s_t, the level of sparsity at training step t, s_i and s_f respectively the initial and final level of sparsity, and α, β being two tuning parameters, modifying either the steepness of the scheduling (Figure 1a), or its horizontal offset (Figure 1b), to better suit the problem or architecture that is used.



(a)



(b)

Figure 1: Visualization of the variation of the scheduling for different α and β values.

We empirically found that $\alpha = 14$ and $\beta = 5$ provide good default values for all the tested architectures and datasets and that final performance was very stable when varying the parameters around those default values (Appendix A). Those are the values we use for all of our experiments.

4 Experimental Setup

Pruning Methods. We compare our pruning technique to several state-of-the-art pruning schedules: One-Shot Pruning, Iterative Pruning and Automated Gradual Pruning (AGP), under a fixed training budget. The optimal training iteration at which the pruning process starts for those schedules, *i.e.* the pretraining phase, is determined with a grid search. More precisely, it corresponds to the pruning process starting at 40%, 20% and 20% of the training budget, for the One-Shot Pruning, Iterative Pruning and AGP, respectively, as shown in Figure 2.

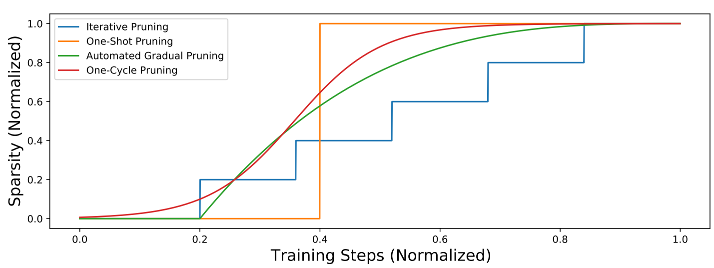


Figure 2: The Studied Pruning Schedules.

Datasets and Architectures. For our experiments, the datasets have been chosen to be various in terms of image resolution and number of classes. In particular, we evaluate our methods on the three following datasets: CIFAR-10 (20), CIFAR-100 (20) and Caltech-101 (21).

Moreover, those datasets are tested on two types of popular convolutional network architectures: VGG-16 (22) and ResNet-18 (23).

Training Procedure. The networks we use for our experiments are trained from a random initialization. The images are first resized to 224×224 and are augmented by using horizontal flips, rotations, image warping and random cropping. We train each model using the 1cycle learning rate method (24), where the training starts with a learning rate warmup until a nominal value of 0.001, then gradually decay until the end of the training.

5 Results

5.1 ResNet-18

We compare the results on the three studied datasets and for high sparsity levels. The results are reported in Table 1.

		One-Shot	Iterative	AGP	One-Cycle
CIFAR-10					
Sparsity	80%	93.10 ± 0.03	93.13 ± 0.03	93.22 ± 0.22	**93.49 ± 0.14**
	90%	92.42 ± 0.21	91.72 ± 0.08	92.85 ± 0.09	**93.31 ± 0.20**
	95%	91.58 ± 0.04	87.54 ± 0.39	92.04 ± 0.07	**92.76 ± 0.16**
CIFAR-100					
Sparsity	80%	74.21 ± 0.09	74.18 ± 0.29	74.78 ± 0.09	**74.81 ± 0.16**
	90%	73.34 ± 0.23	71.80 ± 0.05	73.83 ± 0.41	**74.50 ± 0.24**
	95%	71.68 ± 0.16	62.88 ± 0.27	71.92 ± 0.30	**73.34 ± 0.21**
Caltech-101					
Sparsity	80%	80.31 ± 0.89	79.78 ± 0.56	81.93 ± 0.85	**82.31 ± 0.88**
	90%	79.87 ± 0.54	77.84 ± 0.31	80.89 ± 0.90	**81.84 ± 0.16**
	95%	78.57 ± 1.02	73.83 ± 1.28	78.76 ± 1.27	**79.81 ± 0.92**

Table 1: Comparison of results of 3 runs on ResNet-18.

We observe that One-Cycle Pruning consistently outperforms other pruning schedules. Interestingly, One-Shot pruning outperforms iterative pruning for most datasets and sparsity levels, which can be explained by the restricted training budget. Longer retraining are required for iterative pruning, which was also reported by (5). Results for VGG16 are reported in Appendix B.

5.2 Speed of Convergence

Previous experiments where conducted with a fixed training budget. To better emphasize the impact of the pruning schedule on the training dynamics, we now propose to let the training budget change according to the needs of the pruning method until reaching a desired performance. The pretraining phase is kept unchanged, with only the fine-tuning budget being extended accordingly.

We report in Table 2 the training budget required to reach 90%, 70% and 80% accuracy on CIFAR-10, CIFAR-100, Caltech-101 dataset respectively, using ResNet-18 pruned to a sparsity of 95%.

	One-Shot	Iterative	AGP	One-Cycle
CIFAR-10	$2.5\times$	$4\times$	$1\times$	$1\times$
CIFAR-100	$2.5\times$	$5\times$	$1.25\times$	$1\times$
Caltech-101	$2\times$	$3.2\times$	$1.4\times$	$1\times$

Table 2: Training budget required to prune ResNet-18 to 95% to a fixed performance. The budget is expressed relatively to One-Cycle Pruning

Overall, the technique that requires the most important training budget is iterative pruning, followed by one-shot pruning. This trend has also been reported a retraining time required by (5; 11). The training budget of AGP is equal or higher to our method for all the tested cases.

6 Conclusion & Perspectives

In this work, we have proposed One-Cycle Pruning, a novel pruning schedule that allows a network to be pruned during the training phase, removing the needs of an initial pretraining phase but also of a complex and time-consuming fine-tuning phase. When compared to common pruning schedules, One-Cycle Pruning provides comparable or better results with significantly less computation required. Further work could include the study of the quality of lottery ticket found with such a pruning schedule when compared to other schedules. Supplementary materials such as the code to reproduce our results and a blog post describing the experiments are available in Appendix C.

References

[1] Jonathan Frankle and Michael Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks.," in *International Conference on Learning Representations, ICLR*, 2019.

[2] Davis W. Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John V. Guttag, "What is the state of neural network pruning?," in *Proceedings of Machine Learning and Systems, MLSys*, 2020.

[3] Yann LeCun, John S. Denker, and Sara A. Solla, "Optimal brain damage," in *Advances in Neural Information Processing Systems, NeurIPS*, 1989.

[4] Babak Hassibi, David G. Stork, Gregory Wolff, and Takahiro Watanabe, "Optimal brain surgeon: Extensions and performance comparisons," in *Proceedings of the 6th International Conference on Neural Information Processing Systems, NeurIPS*, 1993.

[5] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf, "Pruning filters for efficient convnets," *International Conference on Learning Representations, ICLR*, 2017.

[6] Yihui He, Xiangyu Zhang, and Jian Sun, "Channel pruning for accelerating very deep neural networks," in *International Conference on Computer Vision, ICCV*, 2017.

[7] Nathan Hubens et al., "An experimental study of the impact of pre-training on the pruning of a convolutional neural network," in *The International Conference on Applications of Intelligent Systems, APPIS*, 2020.

[8] Christos Louizos, Max Welling, and Diederik P. Kingma, "Learning sparse neural networks through l_0 regularization," in *International Conference on Learning Representations, ICLR*, 2018.

[9] Dmitry Molchanov, Arsenii Ashukha, and Dmitry Vetrov, "Variational dropout sparsifies deep neural networks," in *Proceedings of the 34th International Conference on Machine Learning, ICML*, 2017.

[10] Trevor Gale, Erich Elsen, and Sara Hooker, "The state of sparsity in deep neural networks," *The International Conference on Machine Learning, ICML*, 2019.

[11] Song Han, Jeff Pool, John Tran, and William Dally, "Learning both weights and connections for efficient neural network," in *Advances in Neural Information Processing Systems, NeurIPS*, 2015.

[12] Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz, "Pruning convolutional neural networks for resource efficient inference," in *International Conference on Learning Representations, ICLR*, 2017.

[13] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip Torr, "SNIP: Single-shot network pruning based on connection sensitivity," in *International Conference on Learning Representations, ICLR*, 2019.

[14] Michael Zhu and Suyog Gupta, "To prune, or not to prune: Exploring the efficacy of pruning for model compression," in *International Conference on Learning Representations, ICLR*, 2018.

[15] Jonathan Frankle, David J. Schwab, and Ari S. Morcos, "The early phase of neural network training," in *International Conference on Learning Representations, ICLR*, 2020.

[16] Alessandro Achille, Matteo Rovere, and Stefano Soatto, "Critical learning periods in deep networks," in *International Conference on Learning Representations, ICLR*, 2019.

[17] Aditya Golatkar, Alessandro Achille, and Stefano Soatto, "Time matters in regularizing deep networks: Weight decay and data augmentation affect early learning dynamics, matter little near convergence," *CoRR*, vol. abs/1905.13277, 2019.

[18] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M. Roy, and Michael Carbin, "The lottery ticket hypothesis at scale," *CoRR*, vol. abs/1903.01611, 2019.

[19] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin, "Linear mode connectivity and the lottery ticket hypothesis," in *International Conference on Machine Learning, ICML*, 2020.

[20] Alex Krizhevsky, "Learning multiple layers of features from tiny images," 2009.

[21] Fei-Fei Li, Rob Fergus, and Pietro Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," in *Conference on Computer Vision and Pattern Recognition Workshop, CVPR*, 2004.

[22] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR*, 2015.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Conference on Computer Vision and Pattern Recognition, CVPR*, 2016.

[24] Leslie Smith and Nicholay Topin, "Super-convergence: Very fast training of residual networks using large learning rates," *CoRR*, vol. abs/1708.07120, 2017.

[25] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell, "Rethinking the value of network pruning," in *International Conference on Learning Representations, ICLR*, 2019.

A Experiments on α and β

We performed a grid-search to find the best pair of α and β values and found it to be $\alpha = 14$ and $\beta = 5$. In Table 1, we report pairs close to the optimal one, to illustrate the stability of One-Cycle Pruning.

		β				
		3	4	5	6	7
α	13	93.10 ± 0.18	93.23 ± 0.12	93.30 ± 0.07	93.31 ± 0.05	92.97 ± 0.07
	14	93.13 ± 0.06	93.16 ± 0.07	**93.46 ± 0.13**	93.09 ± 0.18	93.25 ± 0.12
	15	93.10 ± 0.03	93.21 ± 0.14	93.19 ± 0.19	93.17 ± 0.08	93.28 ± 0.03

Table 3: Results of the pruning to 90% on Resnet-18 trained on CIFAR-10.

B VGG16

For our experiments, we use the VGG-16 network (22), as modified by (25). It consists of 13 convolutional layers and 2 fully-connected layers, with each convolutional layer being followed by a batch normalization layer. We compare the results on the three studied datasets and for high sparsity levels. The results are reported in Table 4.

		One-Shot	Iterative	AGP	One-Cycle
CIFAR-10					
Sparsity	80%	90.25 ± 0.14	90.64 ± 0.19	**90.87 ± 0.15**	90.84 ± 0.09
	90%	89.82 ± 0.19	89.76 ± 0.18	90.67 ± 0.25	**90.72 ± 0.40**
	95%	89.73 ± 0.37	81.46 ± 2.87	90.56 ± 0.31	**90.67 ± 0.11**
CIFAR-100					
Sparsity	80%	67.83 ± 0.19	67.80 ± 0.15	67.93 ± 0.06	**68.34 ± 0.38**
	90%	67.33 ± 0.16	2.66 ± 1.31	67.88 ± 0.39	**68.24 ± 0.45**
	95%	66.16 ± 0.49	1.95 ± 0.70	**67.51 ± 0.19**	**67.51 ± 0.16**
Caltech-101					
Sparsity	80%	77.81 ± 0.96	78.23 ± 0.35	78.45 ± 0.85	**78.90 ± 0.88**
	90%	78.77 ± 1.06	74.42 ± 2.79	**78.57 ± 0.21**	78.56 ± 0.31
	95%	76.99 ± 0.78	42.61 ± 2.60	78.68 ± 0.53	**78.99 ± 0.50**

Table 4: Comparison of results of VGG-16. Accuracies are averaged over 3 iterations.

As for ResNet-18, we observe that One-Cycle Pruning outperforms other pruning schedules in most cases.

C Supplementary Material

The code to reproduce our results has been made available at `https://github.com/nathanhubens/One-Cycle-Pruning`.

Also, a blog post describing some of our experiments is available at: `https://nathanhubens.github.io/posts/deep%20learning/2021/06/15/OneCycle.html`