Les Cahiers du GERAD ISSN: 0711–2440

Neural network sparsification using Gibbs measures

A. Labach,
S. Valaee

G–2020–23–EIW13

April 2020

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2020
– Bibliothèque et Archives Canada, 2020

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2020
– Library and Archives Canada, 2020

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Neural network sparsification using Gibbs measures

Alex Labach
Shahrokh Valaee

*Department of Electrical & Computer Engineering,
University of Toronto, Toronto (Ontario) Canada,
M5S 1A1*

alex.labach@mail.utoronto.ca
valaee@ece.utoronto.ca

April 2020
Les Cahiers du GERAD
G–2020–23–EIW13

Abstract: *Pruning methods for deep neural networks based on weight magnitude have shown promise in recent research. We propose a new, highly flexible approach to neural network pruning based on Gibbs measures. We apply it with a Hamiltonian that is a function of weight magnitude, using the annealing capabilities of Gibbs measures to smoothly move from regularization to adaptive pruning during an ordinary neural network training schedule. Comparing to several established methods, we find that our network outperforms those that do not use extra training steps and achieves a high accuracy much faster than those that do. We achieve a ¡3% reduction in accuracy on CIFAR-10 with ResNet-32 when pruning 90% of weights.*

1 Introduction

Neural network sparsification via connection pruning has recently been a topic of renewed academic interest. Recent work has shown both the practical utility of pruning in neural network compression systems [6] and theoretical insights gained by experiments with pruning [2]. With modern deep neural network topologies, it is common to be able to prune over 80% of weights with little degradation in performance.

While other criteria have been explored, recent research has focused on pruning weights with low magnitude, since magnitude-based pruning schemes have proven more effective than other established ones [4, 5]. However, existing schemes often require loops of pruning and retraining to achieve good results [6, 2], substantially increasing training time. Our work seeks to approach the performance of such methods without requiring additional training.

Taking inspiration from statistical physics, we propose inducing a Gibbs measure over the weights of a neural network, and sampling from it during training to determine pruning masks. This procedure induces a learned network structure that is resilient to high degrees of pruning. Gibbs measures are highly flexible in terms of network properties that they can express, and quadratic energy functions, such as Ising models, can capture parameter interactions and induce desired structure in pruning masks. They also naturally allow a temperature parameter to be used for annealing, gradually converging to a final pruning mask during training and improving network resilience to pruning. We propose a simple energy function based on weight magnitude to define the Gibbs measure used in our experiments.

2 Proposed method

A Gibbs measure is a probability measure over a vector $\mathbf{x}$ of the form:

$$p(\mathbf{x}) = \frac{1}{Z(\beta)} e^{-\beta H(\mathbf{x})}, \tag{1}$$

where $H(\mathbf{x})$ is the Hamiltonian function, or energy, β is an inverse temperature parameter that can be used for annealing, and $Z(\beta)$ is a partition function that normalizes the measure. In our proposed method, $\mathbf{x}$ represents a pruning mask for a single neural network layer, with $x_i \in \{0, 1\}$. Given the weights of the layer as a flattened vector $\mathbf{w}$, a weight w_i is masked (treated as zero) during training if the corresponding x_i is 0. $\mathbf{x}$ is sampled from $p(\mathbf{x})$ at every training step and once again after training to determine the final mask.

To sample in such a way that weights with lower magnitude are more likely to be masked, we use a Hamiltonian of the form:

$$H(\mathbf{x}) = \sum_i (Q(p, \mathbf{w} \circ \mathbf{w}) - w_i^2) x_i, \tag{2}$$

where $\circ$ represents elementwise multiplication, p is the target pruning fraction, and $Q(p, \mathbf{w} \circ \mathbf{w})$ represents the pth quantile of the values in the vector $\mathbf{w} \circ \mathbf{w}$.

We perform annealing by increasing β from a low value to a high value while training. At high temperatures (low β), differences in the Hamiltonian do not affect sampling much, and so roughly 50% of weights are pruned randomly. This is equivalent to the regularization method dropconnect [10] and starts conditioning the network to be robust under weight pruning. Once annealed to a low temperature (high β), the Gibbs measure converges to pruning the fraction p of weights with lowest magnitude.

This is a highly flexible pruning approach, and although we only consider one Hamiltonian formulation in this paper, others could be designed that take into account characteristics such as network activations or interactions between weights. A Hamiltonian with quadratic terms, such as an Ising model, could induce structure in the pruning masks so as to make them more practically applicable to network compression.

In general, Gibbs measures are computationally expensive to sample from, since the partition function contains many terms. However, since our proposed Hamiltonian is a sum of terms that are each a function of just one element in $\mathbf{x}$, the measure factors into functions of each x_i, and each x_i can therefore be sampled independently. More complex Hamiltonians would likely require similar shortcuts based on independence, or acceleration on GPU or hardware using an efficient sampler such as the Swendsen-Wang algorithm [1] to be realistically usable at each training step.

The goal of our proposed method is to optimize a network's parameters through training at the same time as the final pruning mask over the network is being determined. This means that early on in training, the network learns to be robust under random pruning and receives the regularization benefits of dropconnect, and then gradually converges towards a particular pruning mask, spending a significant amount of later training time adapting to the particular structure of the pruning mask. The balance between time spent on adapting to random masks and time spent adapting to the final mask can be controlled by the particular annealing schedule for β used in training.

3 Experiments

We compare the performance of our proposed method to various baselines and established methods using ResNet-20 and ResNet-32 [7] on the CIFAR-10 [9] dataset. The networks are trained for 200 epochs using the Adam optimizer [8], with a learning rate initially set to 1×10^{-3} and reduced by a factor of 10 every 60 epochs. This achieves baseline top-1 accuracies of 90.7% for ResNet-20 and 91.6% for ResNet-32. We prune all convolutional layers except for the first one, following the recommendation in [4]. When using our proposed method, we anneal β according to a logarithmic schedule from 0.7 to 10000 over the first 128 epochs. These values were chosen empirically to cover a range from an effective pruning rate of 50% to p.

We compare our methods to two kinds of established methods: those that do not add extra training steps to the optimization procedure and those that do. Since our method does not add extra training steps, when comparing to other methods that do not either, we simply compare final accuracies at different p values. This comparison is shown in Tables 1 and 2. One-off pruning simply removes the fraction p of weights in each layer with the lowest magnitude, with no changes to the training procedure. We also test applying a random pruning mask at the beginning of training and maintaining it throughout, effectively training a smaller network from the start. We test using $l1$ regularization with a tuned penalty of 0.001 to induce sparsity during training before masking the fraction p weights in each layer with the lowest magnitude. Finally, we test targeted dropout, which is a recently proposed method described in [5]. We use the most successful hyperparameter settings described in the paper: $\alpha = 0.75, \gamma = 0.9$. The results show our proposed method consistently outperforming the other tested pruning methods at all p values.

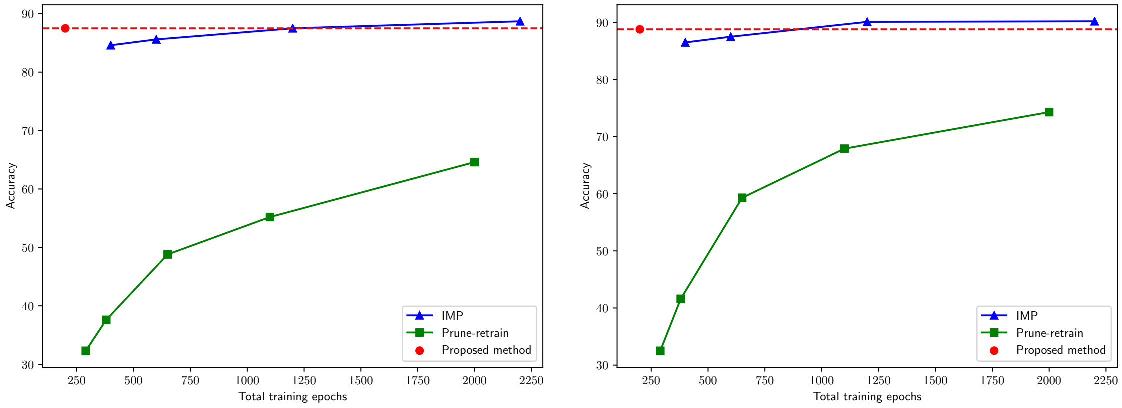
We also compare our proposed method to established pruning methods that require additional training steps beyond the initial network training in Figure 1. In these comparisons, we look at how many training epochs are required for such methods to match the performance of our proposed method.

Table 1: Comparison of our proposed method to other pruning methods that do not use retraining on ResNet-20. Values are top-1 accuracy in percent on CIFAR-10 averaged over two runs

Method	Pruning rate			
	50%	75%	90%	95%
One-off	79.6	17.6	10.0	9.9
Random mask	89.3	87.2	83.2	79.8
l1 loss	81.6	81.7	75.6	43.1
Targeted dropout	82.1	82.1	82.1	21.2
Proposed method	89.9	89.0	87.5	85.2

Table 2: Comparison of our proposed method to other pruning methods that do not use retraining on ResNet-32. Values are top-1 accuracy in percent on CIFAR-10 averaged over two runs

Method	Pruning rate			
	50%	75%	90%	95%
One-off	86.3	29.9	10.0	10.0
Random mask	90.3	88.3	84.8	82.1
l1 loss	81.6	81.1	82.4	55.6
Targeted dropout	87.1	87.1	87.1	26.5
Proposed method	90.5	90.4	88.8	87.1



(a) ResNet-20

(b) ResNet-32

Figure 1: Comparison of our proposed method to other pruning methods that use retraining on ResNet variants. Values are top-1 accuracy in percent on CIFAR-10. All methods prune 90% of weights

The first method we compare to is that proposed in [6], which we call prune-retrain sparsification. In this method, once training is complete, a certain percentage of the weights with lowest magnitude are pruned and the network is retrained for a certain number of epochs. This procedure is then repeated several times, with the pruning percentage increasing each time. We use pruning percentages of 10%, 20%, 30%, etc. up to 90%, retraining for n epochs after each step, leading to a total of $200 + 9n$ epochs. n is varied to test different overall training times. The other method we compare to is iterative magnitude pruning (IMP) [2] with rewinding [3]. This method trains the network several times, pruning gradually more each time and then rewinding the network weights to the values they had after the first 500 training steps. To test different training times, we vary the number of number of times that the network is trained with some pruning rate between the initial rate of 0% and the final rate of 90%.

These results show that established iterative methods require many more training steps to achieve the same results as our proposed method: six times as many on ResNet-20 and over three times as long on ResNet-32 for IMP. Prune-retrain sparsification had far worse performance than our method, even with ten times the number of training epochs. For many applications and datasets, trading a slight reduction in accuracy for being able to avoid expensive retraining schemes could be preferable. It also might be possible to combine our proposed method with an iterative retraining scheme to produce the same accuracies as IMP with less training time.

In summary, our experiments show our proposed method outperforming several other methods that do not use network retraining, and achieving high performance much more quickly than methods that use network retraining.

4 Conclusion

We introduce a novel neural network sparsification method based on Gibbs measures that achieves high pruning ratios with little reduction in accuracy and without requiring additional training steps. It shows the efficacy of simultaneously training and pruning a network rather than training and pruning as distinct steps at different times. Future work could use this method to accelerate iterative magnitude pruning by converging to an effective pruning mask more quickly. The generality of Gibbs measures also means that similar pruning methods could be developed that induce desired structures in the final pruning mask.

References

[1] A. Barbu and Song-Chun Zhu. Generalizing Swendsen-Wang to sampling arbitrary posterior probabilities. IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(8):1239–1253, 2005.

[2] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In Proceedings of the International Conference on Learning Representations (ICLR), 2019.

[3] Jonathan Frankle, Karolina Dziugaite, Daniel M. Roy, and Michael Carbin. Stabilizing the lottery ticket hypothesis. arXiv preprint arXiv:1903.01611, 2019.

[4] Trevor Gale, Erich Elsen, and Sara Hooker. The state of sparsity in deep neural networks. arXiv preprint arXiv:1902.09574, 2019.

[5] Aidan N Gomez, Ivan Zhang, Kevin Swersky, Yarin Gal, and Geoffrey E Hinton. Learning sparse networks using targeted dropout. arXiv preprint arXiv:1905.13678, 2019.

[6] Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. In Proceedings of the International Conference on Learning Representations (ICLR), 2016.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016.

[8] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[9] Alex Krizhevsky. Learning multiple layers of features from tiny images. University of Toronto, 05 2012.

[10] Li Wan, Matthew Zeiler, Sixin Zhang, Yann Le Cun, and Rob Fergus. Regularization of neural networks using dropconnect. In International Conference on Machine Learning, pages 1058–1066, 2013.