PROGRESSIVE WEIGHT PRUNING OF DEEP NEURAL NETWORKS USING ADMM

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep neural networks (DNNs) although achieving human-level performance in many domains, have very large model size that hinders their broader applications on edge computing devices. Extensive research work have been conducted on DNN model compression or pruning. However, most of the previous work took heuristic approaches. This work proposes a progressive weight pruning approach based on ADMM (Alternating Direction Method of Multipliers), a powerful technique to deal with non-convex optimization problems with potentially combinatorial constraints. Motivated by dynamic programming, the proposed method reaches extremely high pruning rate by using partial prunings with moderate pruning rates. Therefore, it resolves the accuracy degradation and long convergence time problems when pursuing extremely high pruning ratios. It achieves up to $34\times$ pruning rate for ImageNet dataset and $167\times$ pruning rate for MNIST dataset, significantly higher than those reached by the literature work. Under the same number of epochs, the proposed method also achieves faster convergence and higher compression rates. The codes and pruned DNN models are released in the anonymous link `bit.ly/2zxdlss`.

1 INTRODUCTION

Deep neural networks (DNNs) have achieved human-level performance in many application domains such as image classification (Krizhevsky et al., 2012), object recognition (LeCun et al., 1998; He et al., 2016), natural language processing (Hinton et al., 2012; Dahl et al., 2012), etc. At the same time, the networks are growing deeper and bigger for higher classification/recognition performance (i.e., accuracy) (Simonyan & Zisserman, 2015). However, the very large DNN model size increases the computation time of the inference phase. To make matters worse, the large model size hinders DNN' deployments on edge computing, which provides the ubiquitous application scenarios of DNNs besides cloud computing applications.

As a result, extensive research efforts have been devoted to the *DNN model compression*, in which DNN *weight pruning* is a representative technique. Han et al. (2015) is the first work to present the DNN weight pruning method, which prunes the weights with small magnitudes and retrains the network model, heuristically and iteratively. After that, more sophisticated heuristics have been proposed for the DNN weight pruning, e.g., incorporating both weight pruning and growing (Guo et al., 2016), L_1 regularization method (Wen et al., 2016), and genetic algorithm (Dai et al., 2017). Other improvement directions of weight pruning include trading-off between accuracy and compression rate, e.g., the *energy-aware pruning* (Yang et al., 2017), and incorporating regularity, e.g., the *channel pruning* (He et al., 2017) and *structured sparsity learning* (Wen et al., 2016).

While the weight pruning technique explores the redundancy in the number of weights of a network model, there are other sources of redundancy in a DNN model. For example, the weight quantization (Leng et al., 2017; Park et al., 2017; Zhou et al., 2017; Lin et al., 2016; Wu et al., 2016; Rastegari et al., 2016; Hubara et al., 2016; Courbariaux et al., 2015) and clustering (Zhu et al., 2017; Han et al., 2016) techniques explore the redundancy in the number of bits for weight representation. The activation pruning technique (Jung et al., 2018; Sharify et al., 2018) leverages the redundancy in the intermediate results. While our work focuses on weight pruning as the major DNN model compression technique, it is orthogonal to the other model compression techniques and might be integrated under a single ADMM-based framework for achieving more compact network models.

The majority of prior work on DNN weight pruning take heuristic approaches to reduce the number of weights as much as possible, while preserving the expressive power of the DNN model. Then how can we push for the utmost of the DNN model sparsity without hurting the accuracy? and what is the maximum compression rate we can achieve by the weight pruning technique? Towards this end, Zhang et al. (2018b) took a tentative step by proposing an optimization-based approach that leverages ADMM (Alternating Direction Method of Multipliers), a powerful technique to deal with non-convex optimization problems with potentially combinatorial constraints. This direct ADMM-based weight pruning technique can be perceived as a smart DNN regularization where the regularization target is dynamically changed in each ADMM iteration. As a result it achieves higher compression (pruning) rate than the heuristic methods.

Inspired by Zhang et al. (2018b), in this paper we propose the progressive weight pruning approach that incorporates both ADMM-based regularization and masked retraining, and takes a progressive means targeting at extremely high compression (pruning) rates with negligible accuracy loss. The contributions of this work are summarized as follows:

- We make a key observation that when pursuing the extremely high compression rate (say 150× for LeNet-5 or 30× for AlexNet), the direct ADMM-based weight pruning approach (Zhang et al., 2018b) cannot produce exactly sparse models at convergence, in that many weights to be pruned are close to zeros, but not exactly zeros. Certain accuracy degradation will result from this phenomenon if we simply set the weights to zeros.
- We propose and implement the progressive weight pruning paradigm that reaches an extremely high compression rate through multiple partial prunings with progressive pruning rates. This progressive approach, motivated by dynamic programming, helps to mitigate the long convergence time by direct ADMM pruning.
- Extensive experiments are performed by comparing with many state-of-the-art weight pruning approaches and the highest compression rates in the literature are achieved by our progressive weight pruning framework, while the loss of accuracy is kept negligible. It achieves up to 34× pruning rate for ImageNet dataset and 167× pruning rate for MNIST dataset, with virtually no accuracy loss. Under the same number of epochs, the proposed method achieves notably faster convergence and higher compression rates than the prior iterative pruning and direct ADMM pruning methods.

We provide the codes (both Caffe and TensorFlow versions) and pruned DNN models (both for ImageNet and MNIST data sets) in the anonymous link: `bit.ly/2zxdlss`.

2 The Progressive Weight Pruning Framework of DNNs

This section introduces the proposed progressive weight pruning framework using ADMM. Section 2.1 describes the overall framework. Section 2.2 discusses the ADMM-based regularization for DNN weight pruning (Zhang et al., 2018b), which we will improve and incorporate into the progressive weight pruning framework. Section 2.3 proposes a direct improvement of masked retraining to restore accuracy. Section 2.4 provides the motivations and details of the proposed progressive weight pruning framework.

2.1 The Overall Framework

Figure 1: The overall progressive weight pruning framework including maksed ADMM-based regularization, thresholding mask updating, and masked retraining steps.

The overall framework of the progressive weight pruning is shown in Figure 1. It applies ADMM-based regularization on a pre-trained (uncompressed) network model. Then it defines thresholding masks, with which the weights smaller than thresholds are forced to be zeros. To restore accuracy, the masked retraining step is applied, that only updates non-zero weights specified by the thresholding masks. These ADMM-based regularization, thresholding mask updating, and masked retaining steps are performed for several rounds, and each round is considered as a partial pruning, progressively pushing for the utmost of the DNN model pruning. Note that in our progressive weight pruning framework, we change the ADMM-based regularization into a "masked" version that reuses the partially pruned model by masking the gradients of the pruned weights, thereby preventing them from recovering to non-zero weights and accelerating convergence.

2.2 ADMM-BASED REGULARIZATION STEP

This ADMM-based regularization step takes a pre-trained network as the input and outputs a pruned network model satisfying some sparsity constraints. Consider an N-layer DNN, where the collection of weights in the i-th (convolutional or fully-connected) layer is denoted by $\mathbf{W}_i$ and the collection of biases in the i-th layer is denoted by $\mathbf{b}_i$. The loss function associated with the DNN is denoted by $f\big(\{\mathbf{W}_i\}_{i=1}^N, \{\mathbf{b}_i\}_{i=1}^N\big)$.

The DNN weight pruning problem can be formulated as:

$$\begin{aligned} \underset{\{\mathbf{W}_i\},\{\mathbf{b}_i\}}{\text{minimize}} \quad & f\big(\{\mathbf{W}_i\},\{\mathbf{b}_i\}\big), \\ \text{subject to} \quad & \mathbf{W}_i \in \mathbf{S}_i,\ i = 1,\ldots,N, \end{aligned} \tag{1}$$

where $\mathbf{S}_i = \{\mathbf{W}_i \mid \text{card}(\mathbf{W}_i) \le l_i\}, i = 1,\ldots,N$ and l_i is the desired number of weights in the i-th layer of the DNN. It is clear that $\mathbf{S}_1,\ldots,\mathbf{S}_N$ are nonconvex sets, and it is in general difficult to solve optimization problems with nonconvex constraints.

The problem can be equivalently rewritten in a format without constraint, which is

$$\underset{\{\mathbf{W}_i\},\{\mathbf{b}_i\}}{\text{minimize}} \quad f\big(\{\mathbf{W}_i\},\{\mathbf{b}_i\}\big) + \sum_{i=1}^N g_i(\mathbf{W}_i), \tag{2}$$

where $g_i(\cdot)$ is the indicator function of $\mathbf{S}_i$, i.e.,

$$g_i(\mathbf{W}_i) = \begin{cases} 0 & \text{if } \text{card}(\mathbf{W}_i) \le l_i, \\ +\infty & \text{otherwise.} \end{cases} \tag{3}$$

The ADMM technique (Boyd et al., 2011) can be applied to solve the weight pruning by formulating the problem as:

$$\begin{aligned} \underset{\{\mathbf{W}_i\},\{\mathbf{b}_i\}}{\text{minimize}} \quad & f\big(\{\mathbf{W}_i\},\{\mathbf{b}_i\}\big) + \sum_{i=1}^N g_i(\mathbf{Z}_i), \\ \text{subject to} \quad & \mathbf{W}_i = \mathbf{Z}_i,\ i = 1,\ldots,N. \end{aligned}$$

Through augmented Lagrangian, the ADMM technique decomposes the weight pruning problem into two subproblems, solving both problems iteratively until convergence. The first subproblem is:

$$\underset{\{\mathbf{W}_i\},\{\mathbf{b}_i\}}{\text{minimize}} \quad f\big(\{\mathbf{W}_i\},\{\mathbf{b}_i\}\big) + \sum_{i=1}^N \frac{\rho_i}{2}\|\mathbf{W}_i - \mathbf{Z}_i^k + \mathbf{U}_i^k\|_F^2. \tag{4}$$

This subproblem is equivalent to the original DNN training plus an L_2 regularization term, and can be effectively solved using stochastic gradient descent with the same complexity as the original DNN training. Note that we cannot prove global optimality of the solution to subproblem (4), just as we cannot prove optimality of the solution to the original DNN training problem.

On the other hand, the second subproblem is:

$$\underset{\{\mathbf{Z}_i\}}{\text{minimize}} \quad \sum_{i=1}^N g_i(\mathbf{Z}_i) + \sum_{i=1}^N \frac{\rho_i}{2}\|\mathbf{W}_i^{k+1} - \mathbf{Z}_i + \mathbf{U}_i^k\|_F^2.$$

Since $g_i(\cdot)$ is the indicator function of the set $\mathbf{S}_i$, the globally optimal solution to this subproblem can be explicitly derived as Boyd et al. (2011):

$$\mathbf{Z}_i^{k+1} = \mathbf{\Pi}_{\mathbf{S}_i}(\mathbf{W}_i^{k+1} + \mathbf{U}_i^k), \tag{5}$$

where $\mathbf{\Pi}_{\mathbf{S}_i}(\cdot)$ denotes the Euclidean projection onto the set $\mathbf{S}_i$. Note that $\mathbf{S}_i$ is a nonconvex set, and computing the projection onto a nonconvex set is a difficult problem in general. However, the special structure of $\mathbf{S}_i = \{\mathbf{W}_i \mid \mathrm{card}(\mathbf{W}_i) \leq l_i\}$ allows us to express this Euclidean projection analytically. Namely, the optimal solution (5) is to keep the l_i largest elements of $\mathbf{W}_i^{k+1} + \mathbf{U}_i^k$ and set the rest to zeros (Boyd et al., 2011).

Finally, we update the dual variable $\mathbf{U}_i$ as $\mathbf{U}_i^{k+1} = \mathbf{U}_i^k + \mathbf{W}_i^{k+1} - \mathbf{Z}_i^{k+1}$. This concludes one iteration of the ADMM algorithm.

In the context of deep learning, the ADMM-based regularization for DNN weight pruning can be understood as a smart DNN regularization technique (see Eqn. (4)), in which the regularization target (in the L_2 regularization term) is dynamically updated in each ADMM iteration. This is one reason that ADMM-based regularization for weight pruning achieves higher performance than heuristic methods and other regularization techniques (Wen et al., 2016), and the Projected Gradient Descent technique (Zhang et al., 2018a).

2.3 Masked Retraining Step

Applying the ADMM-based regularization alone has limitation for high compression rates. At convergence, the pruned DNN model will not be exactly sparse, in that many weights to be pruned will be close to zeros instead of exactly zeros. This is because of the non-convexity property in Subproblem 1 in ADMM-based regularization. Certain accuracy degradation will result from this phenomenon if we simply set those weights to zeros. This accuracy degradation will be non-negligible for high compression rates.

Instead of waiting for the full convergence of ADMM-based regularization, a masked retraining step is proposed, that (i) early terminates the ADMM regularization, (ii) keeps the l_i largest (in terms of magnitude) weights and sets the other weights to zeros, and (iii) performs retraining on the non-zero weights (with zero weights masked) using the training data set. More specifically, masks are applied to gradients of zero weights, preventing them from updating. Essentially, the ADMM-based regularization step sets a good starting point, and then the masked retraining step encourages the remaining non-zero weights to learn to recover classification accuracies.

Integrating masked retraining after ADMM-based regularization, a good compression rate can be achieved with reasonable training time. For example, we can achieve 21× model pruning rate without accuracy loss for AlexNet using a total of 417 epochs, much faster than the iterative weight pruning method (Han et al., 2016), which achieves 9× pruning rate in a total of 960 epochs. When translating into training time, our training time is 72 hours using single NVIDIA 1080Ti GPU, whereas the reported training time in (Han et al., 2016) is 173 hours.

2.4 Progressive Weight Pruning

Although the ADMM-based regularization step in Section 2.2 and the masked retraining step in Section 2.3 together can achieve the state-of-the-art model compression (pruning) rates for many network models, we find a limitation of such approach at extremely high pruning rates, for example, at 150× pruning rate for LeNet-5 or 30× pruning rate for AlexNet.

Specifically, with a very high weight pruning rate, it takes relatively long time for ADMM-based regularization to choose the weights to prune and to make those weights converge to 0. For example, it is difficult for ADMM regularization to converge for 30× pruning rate on AlexNet but easy for 21× pruning rate.

To overcome this difficulty, we propose the progressive weight pruning method. This technique is motivated by dynamic programming, achieving high weight pruning rate by using partial pruning models with moderate pruning rates. We use Figure 2 as an example to show the process to achieve 30× weight pruning rate in AlexNet without accuracy loss. In Figure 2 (a), we start from three partial pruning models, with 15×, 18×, and 21× pruning rates, which can be directly derived from

the uncompressed DNN model via ADMM regularization with masked retraining. To achieve 24× weight pruning rate, we start from these three models and check which gives the highest accuracy (suppose it is the 15× one). Because we start from partial pruning models, the convergence is fast. We then replace 15× partial pruning model by 24× model to derive the 27× model, see Figure 2 (b). In this way we always maintain three partial results and limit the total searching time. Suppose this time the 18× pruning model results in the highest accuracy and then we replace it with the 27× one. Finally, in Figure 2 (c), we find 24× model gives highest accuracy to reach 30× pruning rate.

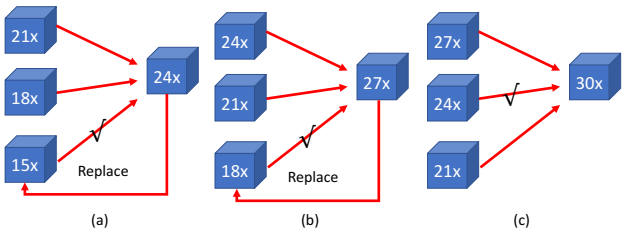

Figure 2: Illustration of the progressive weight pruning idea.

Please note that during the progressive weight pruning, to leverage the partial pruning models, we use a "masked" ADMM regularization to reuse the partial pruning models into the ADMM-based regularization. Specifically, it masks the gradients of the already pruned weights to prevent them from recovering to non-zeros. In this way, ADMM regularization is encouraged to focus on pruning non-zero weights.

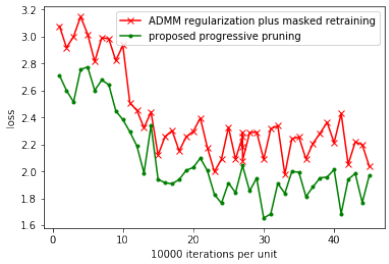

Figure 3: The convergence of the retraining loss of AlexNet by (a) ADMM regularization plus masked retraining and (b) proposed progressive pruning.

Figure 3 demonstrates the convergence of the retraining loss of AlexNet model by (a) ADMM regularization with masked retraining and (b) the proposed progressive pruning. Both methods target at 30× pruning rate. The ADMM regularization with masked retraining performs one-round pruning to 30×, while the proposed progressive pruning performs multiple partial prunings (15× to 24× to 30×). We apply the same total number of iterations of both methods for fair comparison. The total number of epochs will be 730 for both cases, which is still lower than 960 epochs in (Han et al., 2016). We can observe in Figure 3 that by using multiple partial prunings we can achieve faster convergence with lower loss.

3 Experimental Results and Discussions

3.1 Experimental Setups

We evaluate the proposed ADMM-based progressive weight pruning framework on the ImageNet ILSVRC-2012 dataset (Deng et al., 2009) and MNIST dataset (LeCun et al., 1998). We also use DNN weight pruning results from many previous work for comparison. For ImageNet data set, we test on a variety of DNN models including AlexNet (both BAIR/BVLC model and CaffeNet model), VGG-16, ResNet-18, and ResNet-50 models. We test on LeNet-5 model for MNIST data set. The accuracies of the uncompressed DNN models are reported in the tables for reference.

Table 1: Comparisons of weight pruning results on AlexNet for ImageNet data set.

Method	Top-5 Acc.	No. Para.	Rate
Uncompressed	80.27%	61.0M	1×
Network Pruning (Han et al., 2015)	80.3%	6.7M	9×
Optimal Brain Surgeon (Dong et al., 2017)	80.0%	6.7M	9.1×
Low Rank and Sparse Decomposition (Yu et al., 2017)	80.3%	6.1M	10×
Fine-Grained Pruning (Mao et al., 2017)	80.4%	5.1M	11.9×
NeST (Dai et al., 2017)	80.2%	3.9M	15.7×
Dynamic Surgery (Guo et al., 2016)	80.0%	3.4M	17.7×
ADMM Pruning (Zhang et al., 2018b)	80.2%	2.9M	21×
Progressive Weight Pruning (BVLC Model)	80.1%	2.0M	30×
Progressive Weight Pruning (CaffeNet Model)	80.2%	2.0M	30×

Table 2: Top-5 accuracy of direct ADMM pruning (Zhang et al., 2018b) and progressive pruning at different pruning rates on AlexNet for ImageNet data set.

Pruning Rate	Direct ADMM Pruning	Progressive Weight Pruning
18×	80.3%	80.9%
21×	80.2%	80.8%
30×	76.7%	80.1%

We implement our codes in Caffe. Experiments are tested on 12 Nvidia GTX 1080Ti GPUs and 12 Tesla P100 GPUs. As the key parameters in ADMM-based weight pruning, we set the penalty parameter ρ as 1.5×10^{-3} for the masked ADMM-based regularization. When targeting at a high weight pruning rate, we change it to 3.0×10^{-3} for higher performance. To eliminate the already pruned weights in partial pruning results from the masked ADMM-based regularization step, ρ_i is forced to be zero if no more prunings are performed for a specific layer i. We use learning rate of 1.0×10^{-3} for masked ADMM-based regularization and 1.0×10^{-2} for masked retraining.

We provide the codes (both Caffe and TensorFlow versions) and all pruned DNN models (both for ImageNet and MNIST data sets) in the anonymous link: `bit.ly/2zxdlss`.

3.2 Comparison Results and Discussions

Table 1 presents the weight pruning comparison results on the AlexNet model, between our proposed method with prior works. Our weight pruning results clearly outperform the prior work, in that we can achieve 30× weight reduction rate without loss of accuracy. Our progressive weight pruning also outperforms the direct ADMM weight pruning in Zhang et al. (2018b) that achieves 21× compression rate. Also the CaffeNet model results in slightly higher accuracy compared with the BVLC AlexNet model. Table 2 presents more comparison results with the direct ADMM pruning. It can be observed that (i) with the same compression rate, our progressive weight pruning outperforms the direct pruning in accuracy; (ii) the direct ADMM weight pruning suffers from significant accuracy drop with high compression rate (say 30× for AlexNet); and (iii) for a good compression rate (18× and 21×), our progressive weight pruning technique can even achieve higher accuracy compared with the original, uncompressed DNN model.

Table 3, Table 4, and Table 5 present the comparison results on the VGG-16, ResNet-18, and LeNet-5 (for MNIST) models, respectively. These weight pruning results we achieved clearly outperform the prior work, consistently achieving the highest sparsities in the benchmark DNN models. On the VGG-16 model, we achieve 30× weight pruning with comparable accuracy with prior works, while the highest pruning rate in prior work is 19.5×. We also achieve 34× weight pruning with minor accuracy loss. For ResNet-18 model, we have tested 7× weight pruning rate and confirmed no accuracy loss, and 13× pruning and confirmed minor accuracy loss. The experiments on ResNet-50 model has not finished, and we have confirmed 7× weight pruning with no accuracy loss. In fact, there is limited prior work on ResNet weight pruning for ImageNet data set, due to (i) the difficulty in weight pruning since ResNet mainly consists of convolutional layers, and (ii) the slow training

Table 3: Comparisons of weight pruning results on VGG-16 for ImageNet data set.

Method	Top-5 Acc.	No. Para.	Rate
Uncompressed	88.7%	138M	1×
Network Pruning (Han et al., 2015)	89.1%	10.6M	13×
Optimal Brain Surgeon (Dong et al., 2017)	89.0%	10.3M	13.3×
Low Rank and Sparse Decomposition (Yu et al., 2017)	89.1%	9.2M	15×
ADMM Pruning (Zhang et al., 2018b)	88.7%	7.26M	19.5×
Progressive Weight Pruning	88.7%	4.6M	30×
Progressive Weight Pruning	88.2%	4.1M	34×

Table 4: Comparisons of weight pruning results on ResNet-18 (ResNet-50) for ImageNet data set.

Method	Top-5 Acc.	No. Para.	Rate
Uncompressed	89.0%	11.69M	1×
Fine-grained Pruning (Mao et al., 2017)*	92.3%	9.6M	3.4×
Progressive Weight Pruning	89.0%	1.67M	7×
Progressive Weight Pruning	88.0%	0.899M	13×

*Network pruning uses ResNet-50. ResNet-50 has a higher accuracy of 92.4%. It has more parameters and is usually easier to compress with high pruning rate than ResNet-18.

speed of ResNet. Our method, on the other hand, achieves a relatively high training speed, thereby allowing for the weight pruning testing on different large-scale DNN models.

For LeNet-5 model compression, we achieve 167× weight reduction with almost no accuracy loss, which is much higher than prior work under the same accuracy. The prior work Optimal Brain Surgeon (Dong et al., 2017) also achieves a high pruning rate of 111×, but suffers from accuracy drop of around 1% (already non-negligible for MNIST data set).

For other types of DNN models, we have tested the proposed method on the facial recognition application on two representative DNN models (Krafka et al., 2016; Ho, 2016). We demonstrate over 10× weight pruning rate with 0.2% and 0.4% accuracy loss, respectively, compared with the original DNN models.

In summary, the experimental results demonstrate that our framework applies to a broad set of representative DNN models and consistently outperforms the prior work. It also applies to the DNN models that consist of mainly convolutional layers, which are different with weight pruning using prior methods. These promising results will significantly contribute to the energy-efficient implementation of DNNs in mobile and embedded systems, and on various hardware platforms.

Finally, some recent work have focused on the simultaneous weight pruning and weight quantization, as both will contribute to the model storage compression of DNNs. Weight pruning and quantization can be unified under the ADMM framework, and we demonstrate the comparison results in Table 6 using the LeNet-5 model as illustrative example. As can be observed in the table, we can simultaneously achieve 167× weight reduction and use 2-bit for fully-connected layer weight quantization and 3-bit for convolutional layer weight quantization. The overall accuracy is 99.0%. When we focus on the weight data storage, the compression rate is unprecendented 1,910× compared with the original DNN model with floating point representation. When indices (required in weight pruning) are accounted for, the overall compression rate is 623×, which is still much higher than the prior work. It is interesting to observe that the amount of storage for indices is even higher than that for actual weight data.

4 Related Work on DNN Weight Pruning/Model Compression

The pioneering work by Han et al. (2015) shows that DNN weights could be effectively pruned while maintaining the same accuracy after iterative retraining, which gives 9× pruning in AlexNet and 13× pruning in VGG-16. However, higher compression rates could hardly be obtained as the method remains highly heuristic and time-consuming. Extensions of this initial work apply algorithm-level

Table 5: Comparisons of weight pruning results on LeNet-5 for MNIST data set.

Method	Accuracy	No. Para.	Rate
Uncompressed	99.2%	431K	1×
Network Pruning (Han et al., 2015)	99.2%	36K	12.5×
ADMM Pruning (Zhang et al., 2018b)	99.2%	6.05K	71.2×
Optimal Brain Surgeon (Dong et al., 2017)	98.3%	3.88K	111×
Progressive Weight Pruning	99.0%	2.58K	167×

Table 6: Comparisons of weight pruning with quantization results on LeNet-5 for MNIST data set.

Method	Acc. Loss	No. Para.	Conv No. bits	FC No. bits	Total data size /Compress rate	Total size w. index /Compress rate
Uncompressed	0.0%	430.5K	32	32	1.7MB	1.7MB
Iterative pruning (Han et al., 2016)	0.1%	35.8K	8	5	24.2KB / 70.2×	52.1KB / 33×
Learning to share (Ullrich et al., 2017)	0.2%	–	–	–	–	10.4KB / 162×
Our Method	0.2%	2.57K	3	2 (3 for output layer)	0.89KB / **1,910**×	2.73KB / 623×

improvements. For example, Guo et al. (2016) adopts a method that performs both pruning and growing of DNN weights, achieving 17.7× pruning rate in AlexNet. Dai et al. (2017) applies the evolutionary algorithm that prunes and grows weights in a random manner, achieving 15.7× pruning rate in AlexNet. The Optimal Brain Surgeon technique has been proposed Dong et al. (2017), achieving minor improvement in AlexNet/VGGNet but a good pruning ratio of 111× with less than 1% accuracy degradation in MNIST. The L_1 regularization method (Wen et al., 2016) achieves 6× weight pruning in the convolutional layers of CaffeNet. Mao et al. (2017) uses different versions of DNN weight pruning methods, from the fine-grained pruning to channel-wise regular pruning methods. Recently, the direct ADMM weight pruning algorithm has been developed (Zhang et al., 2018b), which is a systematic weight pruning framework and achieves state-of-the-art performance in multiple DNN models.

The above weight pruning methods result in irregularity in weight storage, in that indices are need to locate the next weight in sparse matrix representations. To mitigate the associated overheads, many recent work have proposed to incorporate regularity and structure in the weight pruning framework. Representative work include the channel pruning methods (He et al., 2017; Mao et al., 2017), and row/column weight pruning method (Wen et al., 2016). The latter has been extended in a systematic way in Zhang et al. (2018c). These work can partially mitigate the overheads in GPU, embedded systems, and hardware implementations and result in higher acceleration in these platforms, but typically cannot result in higher pruning ratio than unrestricted pruning. We will investigate the application of progressive weight pruning to the regular/structured pruning as future work.

5 Conclusion

This work proposes a progressive weight pruning approach based on ADMM, a powerful technique to deal with non-convex optimization problems with potentially combinatorial constraints. Motivated by dynamic programming, the proposed method reaches extremely high pruning rate by using partial prunings with moderate pruning rates. Therefore, it resolves the accuracy degradation and long convergence time problems when pursuing extremely high pruning ratios. It achieves up to 34× pruning rate for ImageNet dataset and 167× pruning rate for MNIST dataset, significantly higher than those reached by the literature work. Under the same number of epochs, the proposed method also achieves faster convergence and higher compression rates.

REFERENCES

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. In *Advances in neural information processing systems*, pp. 3123–3131, 2015.

George E Dahl, Dong Yu, Li Deng, and Alex Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on audio, speech, and language processing*, 20(1):30–42, 2012.

Xiaoliang Dai, Hongxu Yin, and Niraj K Jha. Nest: a neural network synthesis tool based on a grow-and-prune paradigm. *arXiv preprint arXiv:1711.02017*, 2017.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

Xin Dong, Shangyu Chen, and Sinno Pan. Learning to prune deep neural networks via layer-wise optimal brain surgeon. In *Advances in Neural Information Processing Systems*, pp. 4857–4867, 2017.

Yiwen Guo, Anbang Yao, and Yurong Chen. Dynamic network surgery for efficient dnns. In *Advances In Neural Information Processing Systems*, pp. 1379–1387, 2016.

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*, pp. 1135–1143, 2015.

Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *International Conference on Learning Representations (ICLR)*, 2016.

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*, pp. 770–778, 2016.

Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pp. 1398–1406. IEEE, 2017.

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, and Brian Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

Jostine Ho. mememoji. `https://github.com/JostineHo/mememoji`, 2016.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In *Advances in neural information processing systems*, pp. 4107–4115, 2016.

Sangil Jung, Changyong Son, Seohyung Lee, Jinwoo Son, Youngjun Kwak, Jae-Joon Han, and Changkyu Choi. Joint training of low-precision neural network with quantization interval parameters. *arXiv preprint arXiv:1808.05779*, 2018.

Kyle Krafka, Aditya Khosla, Petr Kellnhofer, Harini Kannan, Suchendra Bhandarkar, Wojciech Matusik, and Antonio Torralba. Eye tracking for everyone. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Cong Leng, Hao Li, Shenghuo Zhu, and Rong Jin. Extremely low bit neural network: Squeeze the last bit out with admm. *arXiv preprint arXiv:1707.09870*, 2017.

Darryl Lin, Sachin Talathi, and Sreekanth Annapureddy. Fixed point quantization of deep convolutional networks. In *International Conference on Machine Learning*, pp. 2849–2858, 2016.

Huizi Mao, Song Han, Jeff Pool, Wenshuo Li, Xingyu Liu, Yu Wang, and William J Dally. Exploring the regularity of sparse structure in convolutional neural networks. *arXiv preprint arXiv:1705.08922*, 2017.

Eunhyeok Park, Junwhan Ahn, and Sungjoo Yoo. Weighted-entropy-based quantization for deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7197–7205, 2017.

Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European Conference on Computer Vision*, pp. 525–542. Springer, 2016.

Sayeh Sharify, Alberto Delmas Lascorz, Kevin Siu, Patrick Judd, and Andreas Moshovos. Loom: Exploiting weight and activation precisions to accelerate convolutional neural networks. In *Proceedings of the 55th Annual Design Automation Conference*, pp. 20. ACM, 2018.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015.

Karen Ullrich, Edward Meeds, and Max Welling. Soft weight-sharing for neural network compression. *arXiv preprint arXiv:1702.04008*, 2017.

Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *Advances in Neural Information Processing Systems*, pp. 2074–2082, 2016.

Jiaxiang Wu, Cong Leng, Yuhang Wang, Qinghao Hu, and Jian Cheng. Quantized convolutional neural networks for mobile devices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4820–4828, 2016.

Tien-Ju Yang, Yu-Hsin Chen, and Vivienne Sze. Designing energy-efficient convolutional neural networks using energy-aware pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6071–6079, 2017.

Xiyu Yu, Tongliang Liu, Xinchao Wang, and Dacheng Tao. On compressing deep models by low rank and sparse decomposition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7370–7379, 2017.

Dejiao Zhang, Haozhu Wang, Mario Figueiredo, and Laura Balzano. Learning to share: Simultaneous parameter tying and sparsification in deep learning. 2018a.

Tianyun Zhang, Shaokai Ye, Kaiqi Zhang, Jian Tang, Wujie Wen, Makan Fardad, and Yanzhi Wang. A systematic dnn weight pruning framework using alternating direction method of multipliers. *arXiv preprint arXiv:1804.03294*, 2018b.

Tianyun Zhang, Kaiqi Zhang, Shaokai Ye, Jiayu Li, Jian Tang, Wujie Wen, Xue Lin, Makan Fardad, and Yanzhi Wang. Adam-admm: A unified, systematic framework of structured weight pruning for dnns. *arXiv preprint arXiv:1807.11091*, 2018c.

Aojun Zhou, Anbang Yao, Yiwen Guo, Lin Xu, and Yurong Chen. Incremental network quantization: Towards lossless cnns with low-precision weights. In *International Conference on Learning Representations (ICLR)*, 2017.

Chenzhuo Zhu, Song Han, Huizi Mao, and William J Dally. Trained ternary quantization. In *International Conference on Learning Representations (ICLR)*, 2017.