Cross-Layer Group Regularization for Deep Neural Network Pruning

Shuang Gao
NVIDIA Corp.
Santa Clara, CA, USA
susang@nvidia.com

Xin Liu
NVIDIA Corp.
Santa Clara, CA, USA
xinl@nvidia.com

ABSTRACT

Improving weights sparsity is a common strategy for deep neural network pruning. Most existing methods use regularizations that only consider structural sparsity within an individual layer. In this paper, we propose a cross-layer group regularization taking into account the statistics from multiple layers. For residual networks, we use this approach to align kernel sparsity across layers that are tied to each other through element-wise operations: the i^{th} kernel of these layers are put into one regularization group, they either stay or be removed simultaneously during pruning. In this way, the computational and parameter storage cost could be significantly reduced. Experimental results show that this method does not only improve weights sparsity but also align kernel weights sparsity across related layers. Our method is able to prune ResNet up to 90.4% of parameters and improve runtime by 1.5x speedup, without loss of accuracy.

1. INTRODUCTION

While deep neural networks are successfully applied in varies application areas, researchers are making tremendous efforts on reducing its computational cost. Main strategies include developing small network architecture, using low precision computation, model pruning, distilling, and etc.

Given the observation that many neural network weights do not have observable impact on quality, varies pruning methods have been proposed to eliminate useless weights and reduce the computation workload. These methods include optimal brain surgeon, training sparse network and then pruning [4], adding computational loss to the cost during training, and etc.

Among these efforts, training sparse networks is a popular one. The most commonly used regularization is L1, which pushes unimportant weights close to zero. After training, the weight matrix is sparse and it needs sparse matrix multiplication. Group lasso regularization enforces structured sparsity in one layer. By defining all weights in a kernel as a group, it makes these weights to be small or large simultaneously. Thus the kernels with mostly small weights can be removed. Based on L1 and group lasso, Scardapane [5] introduced sparse group lasso, which can further boost the pruning efficiency. However, all these methods improve the sparsity of each layer independently. This limits pruning ability from the model point of view. Figure 1.a illustrates the problem by using two layers in ResNet [1,2]. These layers are followed by a "+" operation. Since each layer is regularized independently, the effective kernel indices for the two layers could be very different. As the amount of element-wise operation is the union of effective weights from two sides, the final number of remaining weights in each layer is much larger than the number of its effective weights. For any layer in figure 1.a, although only 2 kernels are effective, 4 kernels cannot be pruned.

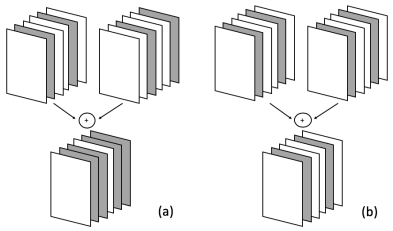

Figure 1: The proposed method improves sparsity alignment among layers.

In this paper, we propose a ***cross-layer group regularization*** (CLG) to solve the problem. It extends the "group" definition to include weights from multiple layers, and thus introduces aligned sparsity pattern among these layers (figure 1.b). Our contributions are: (1) we propose a novel regularization method which introduces aligned kernel sparsity among multiple layers; (2) the method could be used with other regularization such as L1 to achieve state-of-the-art compact model; (3) the experiments show that this method reduces ResNet sizes by 74%~90.4% pruned ratio, with quality comparable to the original work and state-of-the-art pruning results.

2. CROSS-LAYER REGULARIZATION

Given $\boldsymbol{X}$ the mini-batch of $\boldsymbol{N}$ samples (x_i, y_i), with x_i the network input and y_i the ground-truth, the cost function is:

$$Loss(X, W) = \sum_{i=1}^{N} E\big(y_i, f(x_i, W)\big) + \lambda R(W) \qquad (1)$$

with $\boldsymbol{E}$ the prediction loss, $\boldsymbol{R}$ the regularization loss, $\boldsymbol{W}$ the weights, and λ the regularization weight. Most commonly used regularizations are L1 and L2. Group lasso is a variation which can improve structured sparsity within a layer. Given w_g denoting weights of one kernel in layer l, group lasso is defined as:

$$R(w) = \sum_{l \in L} \sum_{g \in G_l} \sqrt{p_g} \times \left\| w_g \right\|_2 \qquad (2)$$

$\boldsymbol{G_l}$ is the set of weight groups, $\boldsymbol{L}$ is the set of layers in the model, p_g is the weight number in w_g. It firstly calculates the 2-norm for weights in each group, and then gets the weighted summary overall all groups cross all layers.

All these regularizations improve weight sparsity of each layer independently, which limits the prune efficiency. We propose CLG regularization which extends the weight group definition: instead of dividing weights of a single layer into groups, it re-organizes weights from multiple convolutional layers into a set of weight groups, and each weight group includes a portion of weights from all these layers. The purpose of this design is to align sparsity of the i[th] kernel among these layers. As figure 2 shows, for convolutional layers that are directly connected by element-wise operations, their i[th] kernels are included in the weight group *i*.

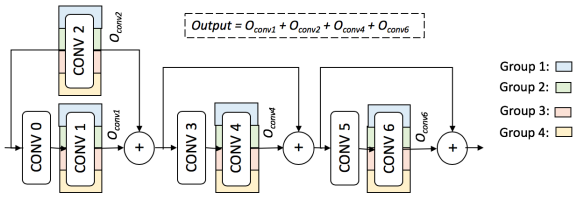

Figure 2: An example of layer group definition.

Eq. 3 describes how to calculate regularization loss $\boldsymbol{R_S}$ for a layer subset (such as colored layers in figure 2). It firstly calculates 2-norm [3] of $\boldsymbol{W_i}$ as the sparsity, with $\boldsymbol{W_i}$ the weight group *i*, and then sums results of $\boldsymbol{K}$ weight groups. p_i is the weight number of group $\boldsymbol{i}$. In case all groups have same amount of weights, they share the same value of p_i. When the layer subset contains only one layer, eq.3 is equivalent to group lasso regularization.

$$R_s = \sum_{i=0}^{K} \sqrt{p_i} \|W_i\|_2 \quad (3)$$

Finally, given $\boldsymbol{G_M}$ the set of all layer subsets make up the model, the total regularization loss of the model is:

$$R = \lambda \sum_{S \in G_M} R_s \quad (4)$$

Intuitively, the effect of eq. 3 is to push the i[th] kernel of all related layers to either as close as to zero, or far away from zero. As a result, every layer has aligned sparsity pattern across kernels, and the number of effective element-wise operations decreases.

EXPERIMENT

We use CIFAR10 dataset to evaluate CLG regularization on ResNet v2 [2] models with bottleneck. Our training and pruning pipeline is similar to Han's method [4]. We use three training-pruning phases: (1) train with CLG, and prune the model; (2) train with L1, and prune the model; (3) fine-tune the model with no regularization. We use Keras with TensorFlow as backend. The runtime is collected on a platform with NVIDIA TitanX Pascal GPU.

Firstly, we exam the effect of CLG regularization by comparing weight sparsity. 5 sample layers are used to produce 5 plots for each method. Every plot has 16x16 grid, representing 256 channels of one layer. White means the sparsity is under pruning threshold, and vice versa. These layers are connected by element-wise operations, and we aim to align their sparsity. As it shows, CLG regularization produces obviously aligned weight sparsity, but L1 does not.

Table 1 shows the pruning and accuracy comparison summary. Models trained by the proposed method (denote as CLG-L1) achieve comparable accuracy as original ResNet [1,2], but they use 74%~90.4% less parameters. Compared to L1 regularization and Li's pruning method [3], CLG-L1 models have 37%~65%, and 40%~72% less parameters respectively. The speedup compared to original work and L1 regularization are 1.5x and 1.32x (figure 4).

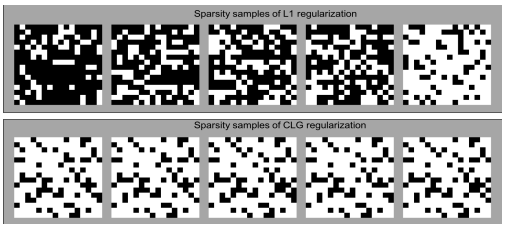

Figure 3: Sparsity comparison

TABLE I. COMPARISON OF ACCURACY AND COMPRESSION

Layer #	*Model*	*Param Num*	*Acc*	*Pruned Ratio*
56	ResNet v1 [1]	0.85M	93.04	---
	Li's v1 [3]	0.73M	93.1	13.7%
	ResNet v2[a]	1.67M	92.9	---
	L1 v2	0.69M	93.6	58.5%
	CLG-L1_A v2[a]	0.44M	**93.4**	74.0%
	CLG-L1_B v2[a]	0.24M	93.0	**85.7%**
110	ResNet v1[1]	1.72M	93.5	---
	Li's v1 [3]	1.16M	93.3	32.4%
	ResNet v2[a]	3.32M	93.7	---
	L1 v2	0.84M	93.5	74.7%
	CLG-L1_A v2[a]	0.46M	**93.7**	86.1%
	CLG-L1_B v2[sa]	0.32M	93.4	**90.4%**

a. 'v2' means v2 with bottleneck.

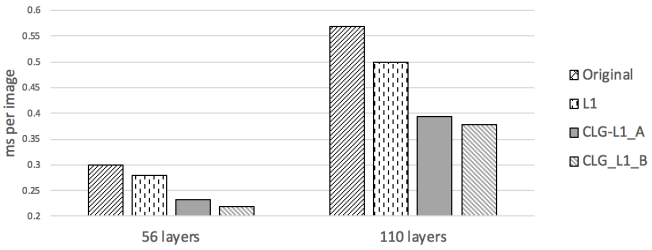

Figure 4: Runtime comparison

CONCLUSION

This paper introduces CLG regularization for deep neural networks pruning. It aligns kernel sparsity across multiple layers; and it can be applied with other regularization method simultaneously. Experimental results demonstrate the efficiency of our algorithm which achieves the state-of-the-art compact model without loss in quality in CIFAR10 classification with ResNet 56/110.

REFERENCES

[1] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016.
[2] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In ECCV, 2016
[3] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf. Pruning filters for efficient convnets. In International Conference on Learning Representations (ICLR), 2017
[4] Han, Song, Pool, Jeff, Tran, John, and Dally, William J. Learning both weights and connections for efficient neural networks. In Advances in Neural Information Processing Systems, 2015.
[5] S. Scardapane, D. Comminiello, A. Hussain, A. Uncini, Group sparse regularization for deep neural networks, Neurocomputing, v.241 n.C, p.81-89, June 2017