Gradient and Magnitude Based Pruning for Sparse Deep Neural Networks

Kaleab Belay
Addis Ababa Institute of Technology
belete.kaleab@gmail.com

Abstract

Deep Neural Networks have memory and computational demands that often render them difficult to use in low-resource environments. Also, highly dense networks are over-parameterized and thus prone to overfitting. To address these problems, we introduce a novel algorithm that prunes (sparsifies) weights from the network by taking into account their magnitudes and gradients taken against a validation dataset. Unlike existing pruning methods, our method does not require the network model to be retrained once initial training is completed. On the CIFAR-10 dataset, our method reduced the number of para[1]meters of MobileNet by a factor of 9X, from 14 million to 1.5 million, with just a 3.8% drop in accuracy.

Introduction

Although algorithms utilizing Multi-Layer Perceptrons have existed since the 1990s, it is only over the last decade that Neural Networks have started seeing wide usage. This is largely due to advances in computing technologies that allow for faster and more parallel execution of operations. State-of-the-art Deep Neural Network (DNN) architectures are dozens of layers deep and have hundreds of millions of parameters, making them impossible to use in low-resource computing environments.

We present an algorithm for reducing the number of parameters of a DNN with minimal loss in accuracy. Our method avoids having to retrain the model by learning the importance of a specific weight in the network through its gradient against a validation dataset. Weights which have gradients below a specified threshold are assumed to have settled close to their resting values and can be pruned based on their magnitudes. Pruning is carried out iteratively according to a specified schedule.

Related Work

Various pruning methods have been proposed to reduce model size while incurring minimal loss in accuracy. In a closely related work, Han et al. (2015) propose a magnitude-based pruning approach in which the algorithm removes all weights below a specific threshold and re-adjusts the remaining weights in the next round of training. The drawback of this method is that it requires the model to be trained first so that the connections that matter can be distinguished from those that do not, resulting in more than 100% performance overhead. The method suggested by Zhu and Gupta (2017) also suffers from similar performance issues.

Method

We present the pseudocode for our proposed pruning scheme, which sparsifies a network given a set of model hyperparameters. For every weight in the model's weight tensors and its corresponding elements in the gradient and bitmask tensors, if the absolute value of the weight is below the weight threshold β and its gradient is less than the gradient threshold γ, we set its weight and its corresponding bitmask to zero. Using NumPy's array vectorization operations, this algorithm takes less than a second to implement on a network with 13.8 million parameters.

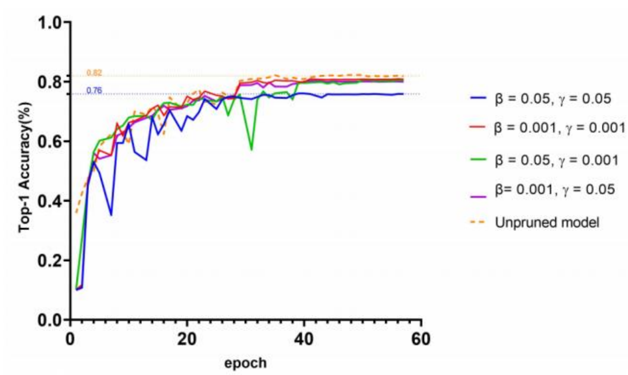
Algorithm 1: Gradient and Magnitude Based Pruning

Parameters: ∇_{wC}, tensor of weight gradients
$\boldsymbol{W}$, weight tensor of the network
$\boldsymbol{B}$, bitmask tensor of the network
β, weight threshold
γ, gradient threshold
δ, pruning schedule
epoch, current epoch

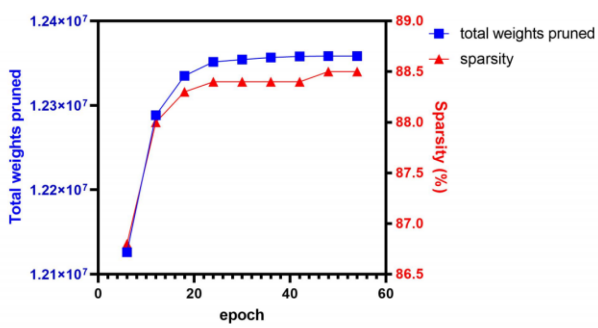
Output: Sparse Neural Network

for each: $w \in \boldsymbol{W}$, $g \in \nabla_{wC}$, $b \in \boldsymbol{B}$ **do**
  if $|w| < \beta$ **and** $|g| < \gamma$ **and** $b \neq 0$ **then**
    $b \leftarrow 0$
    $w \leftarrow 0$

Ensure: $epoch \ \% \ \delta = 0$



(a)



(b)

Figure 1: (a) Top-1 accuracy for different values of β and γ on MobileNet (b) Total weights pruned and sparsity for β = 0.05, γ = 0.05

The bitmask tensors were introduced so as to prevent weights that have already been set to zero from being updated in future backpropagation operations. This is done by performing the Hadamard product of the training gradients with the bitmasks.

Results

We define our own MobileNet in Keras based on the instructions provided by Chen & Su (2017). We compare the accuracy and compression rate achieved by pruning networks trained with the scaling factor α of 0.5 and 1, and width multipliers 1, 2 and 4 on the CIFAR-10 dataset. We vary the pruning parameters β and γ as well as the pruning schedule δ to determine how their interactions affect model size and accuracy. We see that moderately aggressive values assigned to the pruning parameters yield significant reduction in model size while incurring minimal drop in accuracy.

In the experimental results presented in the paper, pruning is executed according to a specified schedule δ. So long as the model received sufficient number of training steps to recover from the changes introduced during pruning, varying the δ didn't yield any sufficient changes in accuracy or compression rate. Figure 1a shows the accuracy of a MobileNet model pruned with different values of β and γ. Most pruning parameters yielded similar results, with the model pruned with β = 0.05 and γ = 0.05 performing worse than the others. The model pruned with β = 0.05 and γ = 0.001 had the best accuracy-compression tradeoff, achieving 89.1% sparsity and a Top-1 accuracy of 78.2%. The model had a higher accuracy when compared to the approach by Chen et al., who were able to achieve an accuracy of 73.8% for similar levels of sparsity.

Conclusion

This work introduces an iterative pruning scheme that reduces model size during training. We demonstrate the efficacy of this scheme and its compression-accuracy tradeoff by using different pruning parameters. We believe these results will encourage the use of model pruning in low resource environments.

References

Chen, H. and Su, C. 2019. An Enhanced Hybrid MobileNet. arXiv preprint. arXiv:1712.04698 [cs.CV].

Han, S.; Pool, J.; Tran, J. and Dally W. J. 2015. Learning both weights and connections for efficient neural networks. In *NeurIPS*.

Rhu, M.; Gimelshein, N.; Clemons, J.; Zulfiqar, A. and Keckler, S. 2015. vdnn: Virtualized deep neural networks for scalable, memory-efficient neural network design. arXiv preprint. arXiv:1602.08124 [cs.DC].

Zhu, M. and Gupta, S. 2015. To prune, or not to prune: exploring the efficacy of pruning for model compression. arXiv preprint. arXiv:1710.01878v2 [stat.ML].