Neuron-based Network Pruning Based on Majority Voting

A. Alqahtani, X. Xie, E. Essa, and M. W. Jones

ICPR 2020
25th INTERNATIONAL CONFERENCE ON PATTERN RECOGNITION
Milan, Italy 10 | 15 January 2021

Swansea University
Prifysgol Abertawe

SwanseaVision
Swansea University • United Kingdom
csvision.swansea.ac.uk

- Introduction
- Background
- Motivation
- Method
- Experimental Details
- Experimental Results
- Conclusion

Introduction

- Deep learning algorithms have shown their robust ability in representation learning and driven state-of-the-art performances in various tasks.
- The significant redundancy in the parameterization has become a widely-recognized property of deep learning.
- The over-parametrized and redundant nature of deep neural networks presents significant challenges for many applications (i.e., deploying sizeable deep learning models to a resource-limited device).
- Moreover, training with more parameters than necessary incurs expensive computational costs and high storage requirements.

Neuron Importance Methods

- **Visual assessment of neurons' properties:**
  - Zeiler et al. (2014) studied single-neuron properties to understand deep representations.
  - Bau et al. (2019) explored pixel-level annotations, providing meaningful insight into the characteristics of the internal representations.
- **Quantitative assessment of neurons' properties:**

  In order to evaluate the importance of hidden neurons:
  - Dhamdhere et al. (2019) utilized integrated gradients by summing the gradients of the output prediction.
  - Amjad et al. (2018) applied information-theoretic quantities (i.e., entropy and mutual information) to understand the outputs of individual neurons.
  - Na et al. (2019) used the highest mean activation to measure the importance of individual units.

1. M. D. Zeiler and R. Fergus, "Visualizing and understanding convolu- tional networks," in ECCV. Springer, 2014, pp. 818–833.
2. D. Bau et. al., "Gan dissection: Visualizing and understanding generative adversarial networks," ICLR, 2019.
3. K. Dhamdhere, M. Sundararajan, and Q. Yan, "How important is a neuron?" ICLR, 2019.
4. R. A. Amjad, K. Liu, and B. C. Geiger, "Understanding indi- vidual neuron importance using information theory," arXiv preprint arXiv:1804.06679, 2018.
5. S. Na, Y. J. Choe, D.-H. Lee, and G. Kim, "Discovery of natural language concepts in individual units of cnns," ICLR, 2019.

Pruning Methods

- **Weight-based methods:**

  Weight-based pruning eliminates unnecessary, low- weight connections between layers of a neural network.

  - LeCun et al. (1990) and Hassibi et al. (1993) are seen as some of the pioneering works in network pruning.
  - Han et al. (2015) removed connections whose absolute values are smaller than a predefined threshold.
  - Li et al. (2017) proposed a method based on the absolute weighted sum, pruning the lowest scores.

- **Neuron-based methods:**

  They remove all connections to a specific neuron, including incoming or outgoing connections.

  - He et al. (2014) proposed a method based on summing the output weights of each neuron, pruning the lowest values.
  - Mariet et al. (2016) introduced Divnet, which selects a subset of diverse neurons and subsequently merges similar neurons into one.

1. Y. LeCun, J. S. Denker, and S. A. Solla, "Optimal brain damage," in *NIPS*, 1990, pp. 598–605.
2. B. Hassibi and D. G. Stork, "Second order derivatives for network pruning: Optimal brain surgeon," in *NIPS*, 1993, pp. 164–171.
3. S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *ANIPS,* 2015, pp. 1135–1143.
4. T. He, Y. Fan, Y. Qian, T. Tan, and K. Yu, "Reshaping deep neural network for fast decoding by node-pruning," in IEEE ICASSP, 2014, pp. 245–249.
5. Z. Mariet and S. Sra, "Diversity networks: Neural network compression using determinantal point processes," *in* *ICLR*, 2016.
6. H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," *in* *ICLR*, 2017.

Motivation

- Most of the existing methods tend to compress the networks through multi-step procedures.
- They mainly benefit from the substantial retraining step, especially when adopting less efficient measurement standards.
- To overcome these issues, our proposed method introduces competent neuron measurement into the pruning process.
- We introduce a comprehensive approach to prune the network’s neurons based on our majority voting method during training, without involving any pre-training or fine-tuning procedures.
- This mechanism helps to measure the importance of neurons and to prune them accordingly into the body of the learning phase.
- This saves time that is needed for the initial training as well as the retraining phases; saving twice the amount of time that is usually necessary to train a model from scratch.

Neuron-based Iterative Pruning

- **Importance of Individual Neuron via Majority voting (MV)**
  - We aim to detect influential neurons in neural networks by evaluating their activation.
  - A majority voting approach is introduced to determine the importance of neurons in each layer.
  - Our method was named majority voting (MV) as it utilizes a majority voting strategy to measure the importance of neurons.
  - It votes for a neuron when all the instances agree.
  - The activation at each neuron is defined as:

$$t_j^{(i)}(x_n) = \sigma(b_j^{(i)} + \sum_p w_{p,j}^{(i-1)} t_p^{(i-1)}(x_n)), \tag{1}$$

Neuron-based Iterative Pruning

- After this, the activation matrix is obtained, the top largest activation neurons are set to 1 and others to 0 by:

$$v_j^{(i)}(x_n) = \begin{cases} 1 & \text{If argsort } (t_j^{(i)}(x_n))[1:l] \\ 0 & \text{Otherwise} \end{cases} \quad (2)$$

- As a result, a binary matrix is obtained, representing the number of neurons and the number of input examples.
- Then, we sum over columns (examples) to score the number of times that neuron is one of the top neurons for given examples, voting for the crucial neurons.

$$y_j^{(i)} = \sum_{n=1}^{N} v_j^{(i)}(x_n) \quad (3)$$

$$\psi_j^{(i)} = y_j^{(i)} = \begin{cases} 1 & \text{If argsort } (y_j^{(i)})[1:k*J] \\ 0 & \text{Otherwise} \end{cases} \quad (4)$$

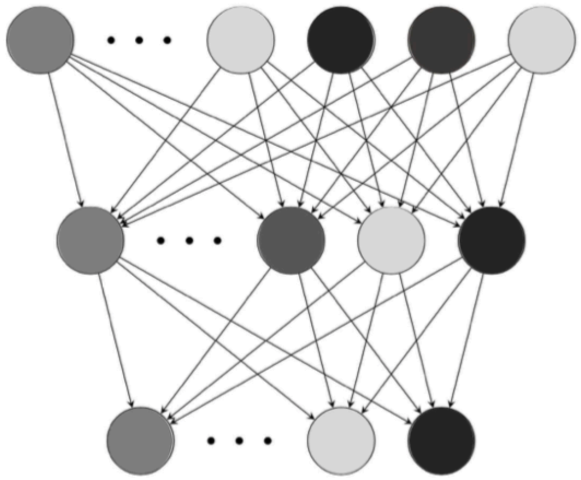
- A set of neurons, which have the largest voting scores, is set to 1 and the remaining to 0.
- We will come up with a binary vector that indicates whether such neurons are important or not.

Neuron-based Iterative Pruning

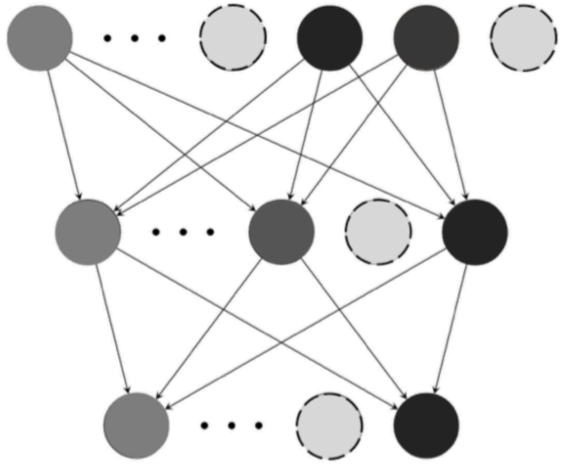
- **Pruning algorithm**

Algorithm 1 Pruning algorithm using Majority Voting (MV)

Input: Training set (x, y), Validation set $(\grave{x}, \grave{y})$, t, and k
Output: A pruned model
initialization
best accuracy $\leftarrow 0$
for $e \leftarrow 1$ *to* E **do**
    Preform standard training procedure
    Preform weights update
    accuracy $\leftarrow$ model accuracy
    if $e \bmod t = 0$ *and* *accuracy* $>$ *best accuracy* **then**
        best accuracy $\leftarrow$ accuracy
        for *each layer* **do**
            Compute the activation for each neuron Eq.(1)
            Vote for largest activations Eq.(2)
            Compute the amount of times a neuron has been voted Eq.(3)
            Vote for $k\%$ of largest voting-score neurons Eq.(4)
            Prune the non-important neurons
        end
    end
end



Before



After

Experimental Details

- The proposed method was trained using Keras and Tensorflow in Python.
- Our proposed method is evaluated using two computer vision benchmark datasets: MNIST and CIFAR-10.
- For fully-connected models, the network architecture consists of three fully-connected layers:
    - (784-1000-1000-1000-10) for MNIST.
    - (3072-4000-1000-4000-10) for CIFAR-10.
- The model was trained end-to-end. No fine-tuning procedures.
- A stochastic gradient descent optimizer was used.
- Each batch contained 100 random shuffled images.
- An initial learning rate of 0.006 with a momentum of 0.9 and weight decay of 0.0002 were used.

Measuring Neuron Importance via Ablation

- Classification performance was used to evaluate the impact of our majority voting method.
- An ablation study allows for evaluating the effectiveness of measuring neuron importance quantitatively.
- The ablation refers to the removal of some parts of the model and the study of its performance.
- We ablate unimportant neurons by forcing the activation to be zero and compute the classification accuracy on the test-set.

	1st Layer	*2nd Layer*	*3rd Layer*	*Cumulative Ablation*
Random	45.46%	61.84%	65.07%	21.39%
Weights Sum	63.75%	67.47%	67.04%	48.62%
Activation Mean	68.97%	68.47%	68.48%	64.75%
Activation SD	69.49%	68.90%	69.22%	66.33%
Activation $l1$-norms	69.39%	68.71%	69.32%	65.94%
Activation $l2$-norms	69.45%	68.73%	69.31%	65.81%
MV	**69.77%**	**69.39%**	**69.66%**	**68.28%**

CIFAR-10

1st Layer	*2nd Layer*	*3rd Layer*	*Cumulative Ablation*
95.4%	97.96%	98.41%	85.32%
95.00%	98.39%	98.57%	94.63%
97.88%	98.52%	98.58%	97.98%
98.58%	98.73%	98.68%	98.44%
98.56%	98.72%	98.65%	98.40%
98.51%	98.73%	98.67%	98.37%
98.68%	**98.75%**	**98.76%**	**98.68%**

MNIST

-

Pruning redundant Neurons during Training

- Our Pruning Method with fully-connected Network.

	FC		*MV Pruning*	
Dataset	***Accuracy***	n^W	***Accuracy***	n^W
MNIST	98.78%	2,794K	98.88%	232K
CIFAR10	71.90%	20,328K	74.21%	4,245K

- Integrating our Pruning Method to Existing Sparse Neural Network

	SC		*MV Pruning*	
Dataset	***Accuracy***	n^W	***Accuracy***	n^W
MNIST	98.74%	89K	98.84%	34K
CIFAR10	74.84%	278K	75.05%	214K

- Extension to Convolutional Neural Networks
  - Our pruned model has reached a maximum of 90.12% accuracy compared to 89.30% accuracy, which was achieved by standard CNN.
  - Our pruned model has removed more than 95% of the CNN's parameters.

Conclusion

- We propose a pruning framework that simultaneously identifies the most critical neurons and removes redundant nodes accordingly.
- The experimental results have demonstrated the effectiveness of our pruning method in maintaining or even improving accuracy after removing unimportant neurons.
- The results also demonstrate that our proposed method is applicable to weight-based pruning methods and adds extra compression.
- Our potential future work is to extend this framework to filters in convolutional neural networks and experiment with more difficult datasets.

Thank You

Any Questions?