Neuron-based Network Pruning Based on Majority Voting

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

Swansea University Prifysgol Abertawe

ICPR 2020 — 25th International Conference on Pattern Recognition — Milan, Italy 10 | 15 January 2021

1. Overview

- The over-parameterized of deep neural networks present significant challenges for many applications.
- Most of the existing methods tend to compress the networks through multi-step procedures.
- To overcome these issues, we propose a method to prune the network's neurons based on our majority voting method during training.
- This mechanism helps to measure the importance of neurons and to prune them accordingly into the body of the learning phase.
- It also saves time that is needed for the initial training as well as the retraining phases.

2. Neuron Importance 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.
- The activation at each neuron is defined as:

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

- Then the top largest activation neurons are set to 1 and others to 0 by:

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

- After that, we sum over columns (examples) to score the number of times that neuron is one of the top neurons for given examples.

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

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

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

3. Neuron-based Iterative Pruning

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 ← 0
for e ← 1 to E do
    Preform standard training procedure
    Preform weights update
    accuracy ← model accuracy
    if e mod t = 0 and accuracy > best accuracy then
        best accuracy ← accuracy
        for each layer do
            Compute the activation for each neuron
            Vote for largest activations
            Compute the amount of times a neuron has been voted
            Vote for k% of largest voting-score neurons
            Prune the non-important neurons
        end
    end
end
```

4. Experimental Results

- 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.

5. Measuring Neuron Importance via Ablation

- Classification performance was used.
- The ablation refers to the removal of some parts of the model and the study of its performance.

CIFAR-10

	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%**

MNIST

	1st Layer	*2nd Layer*	*3rd Layer*	*Cumulative Ablation*
Random	95.4%	97.96%	98.41%	85.32%
Weights Sum	95.00%	98.39%	98.57%	94.63%
Activation Mean	97.88%	98.52%	98.58%	97.98%
Activation SD	98.58%	98.73%	98.68%	98.44%
Activation $l1$-norms	98.56%	98.72%	98.65%	98.40%
Activation $l2$-norms	98.51%	98.73%	98.67%	98.37%
MV	**98.68%**	**98.75%**	**98.76%**	**98.68%**

6. 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, with only less than 5% of the CNN weights.

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