IMPORTANCE ESTIMATION WITH RANDOM GRADIENT FOR NEURAL NETWORK PRUNING

Suman Sapkota *
NepAl Applied Mathematics and Informatics Institute for Research (NAAMII), Nepal
suman.sapkota@naamii.org.np

Binod Bhattarai
University of Aberdeen, UK
binod.bhattarai@abdn.ac.uk

ABSTRACT

Global Neuron Importance Estimation is used to prune neural networks for efficiency reasons. To determine the global importance of each neuron or kernel, most of the existing methods either use activation or gradient information or both, which demands abundant labelled examples. In this work, we use heuristics to derive importance estimation similar to Taylor First Order (TaylorFO) approximation based methods. We name our methods TaylorFO-abs and TaylorFO-sq. We propose two additional methods to improve these importance estimation methods. Firstly, we propagate random gradients from the last layer of a network, thus avoiding the need for labelled examples. Secondly, we normalize the gradient magnitude of the last layer output before propagating, which allows all examples to contribute similarly to the importance score. Our methods with additional techniques perform better than previous methods when tested on ResNet and VGG architectures on CIFAR-100 and STL-10 datasets. Furthermore, our method also complements the existing methods and improves their performances when combined with them.

1 INTRODUCTION AND BACKGROUND

Neural Network Pruning LeCun et al. (1989); Han et al. (2015b); Gale et al. (2019); Blalock et al. (2020) is one of the methods to reduce the parameters, compute and memory requirement. This method differs significantly from knowledge distillation Hinton et al. (2015); Gou et al. (2021) where a small model is trained to produce the output of a larger model. Neural Network Pruning is performed at multiple levels; (i) weight pruning Mozer & Smolensky (1989); Han et al. (2015b;a) removes per parameter basis while (ii) neuron/channel Wen et al. (2016); Lebedev & Lempitsky (2016) pruning removes per neuron or channel basis and (iii) block/group Gordon et al. (2018); Leclerc et al. (2018) pruning removes per a block of network such as residual block or sub-network.

Weight pruning generally achieves a very high pruning ratio getting similar performance only with a few percentages of the parameters. This allows a high compression of the network and also accelerates the network on specialized hardware and CPUs. However, weight pruning in a defined format such as N:M block-sparse helps in improving the performance on GPUs Liu & Wang (2023). Pruning network at the level of neurons or channels helps in reducing the parameters with similar performance, however, the pruning ratio is not that high. Furthermore, pruning at the level of blocks helps to reduce the complexity of the network and creates a smaller network for faster inference. All these methods can be applied to the same model as well. Furthermore, pruning and addition of neurons can bring dynamic behaviour of decreasing and increasing network capacity which has found its application in Continual or Incremental Learning settings as well as Neural Architecture Search Gordon et al. (2018); Zhang et al. (2020); Dai et al. (2020); Sapkota & Bhattarai (2022).

In this work, we are particularly interested in neuron level pruning. Apart from the benefit of reduced parameter, memory and compution, neuron/channel level pruning is more similar to a biological for-

*https://tsumansapkota.github.io/

mulation where the neurons are the basic unit of computation. Furthermore, the number of neurons in a neural network is small compared to the number of connections and can easily be pruned by measuring the global importance LeCun et al. (1989); Hassibi et al. (1993); Molchanov et al. (2016); Lee et al. (2018); Yu et al. (2018). We focus on the global importance as it removes the need to inject bias about the number of neurons to prune in each layer. This can simplify our problem to remove less significant neurons globally which allows us to extend it to unorganized networks other than layered formulation. However, in this work, we focus only on layer-wise or block-wise architectures such as ResNet and VGG.

Previous works show that global importance estimation can be computed using one or all of forward/activation Hu et al. (2016), parameter/weight Han et al. (2015b) or backward/gradient Wang et al. (2020); Lubana & Dick (2020); Evci et al. (2022) signals. Some of the previous techniques use Feature Importance propagation Yu et al. (2018) or Gradient propagation Lee et al. (2018) to find the neuron importance. Others use both activation and gradient information for pruning Molchanov et al. (2016; 2019). Although there are methods using such signals for pruning at initialization Wang et al. (2020), we limit our experiment to the pruning of trained models for a given number of neurons.

In this work, we derive a similar importance metric to Taylor First Order (Taylor-FO) approximations Molchanov et al. (2016; 2019) but from heuristics combining both forward and backward signals. The forward signal, namely the activation of the neuron, and the backward signal, the gradient or the random gradient of the output. We also compare the pruning accuracy with a different number of data samples and find that our method performs better than the previous works in most settings. Furthermore, our method of using the random gradient signal on the last layer and gradient normalization is also applicable to previous methods, showing that our approach improves performance on previous methods as well.

2 Motivation and Our Method

Previous works on global importance based post-training pruning of neurons focus on using forward and backward signals. Since most of these methods are based on Taylor approximation of the change in loss after the removal of a neuron or group of parameters, these methods require input and target value for computing the importance.

We tackle the problem of pruning from the perspective of overall function output without considering the loss.

Forward Signal: The forward signal is generally given by the pre-activation (x_i). If a pre-activation is zero, then it has no impact on the output of the function, i.e. the output deviation with respect to the removal of the neuron is zero. If the incoming connection of a neuron is zero-weights, then the neuron can be simply removed, i.e. it has no significance. If the incoming connection is non-zero then the neuron has significance. Forward signal takes into consideration how data affects a particular neuron.

Backward Signal: The backward signal is generally given by back-propagating the loss. If the outgoing connection of the neuron is zeros, then the neuron has no significance to the function even if it has positive activation. The gradient(δx_i) provides us with information on how the function or loss will change if the neuron is removed.

Importance Metric: Combining the forward and backward signal we can get the influence of the neuron on the loss or the function for given data. Hence, the importance metric (I_i) of each neuron (n_i) for dataset of size M is given by $I_i = \frac{1}{M}\sum_{n=1}^{M} x_i.\delta x_i$, where x_i is the pre-activation and δx_i is its gradient. It fulfils the criterion that importance should be low if incoming or outgoing connections are zeros and higher otherwise.

Problem 1: This importance metric is similar to Taylor-FO Molchanov et al. (2016). However, analysing this, we find that this method produces low importance if the gradient is negative, which to our application is a problem as the function will be changed significantly, even if it lowers the loss. Hence, we make the importance positive either by using the absolute value or the squared value. This gives rise to two of our methods namely:

Taylor-FO-abs : $I_i = \frac{1}{M}\sum_{n=1}^{M} |x_i.\delta x_i|$ **Taylor-FO-sq :** $I_i = \frac{1}{M}\sum_{n=1}^{M} (x_i.\delta x_i)^2$

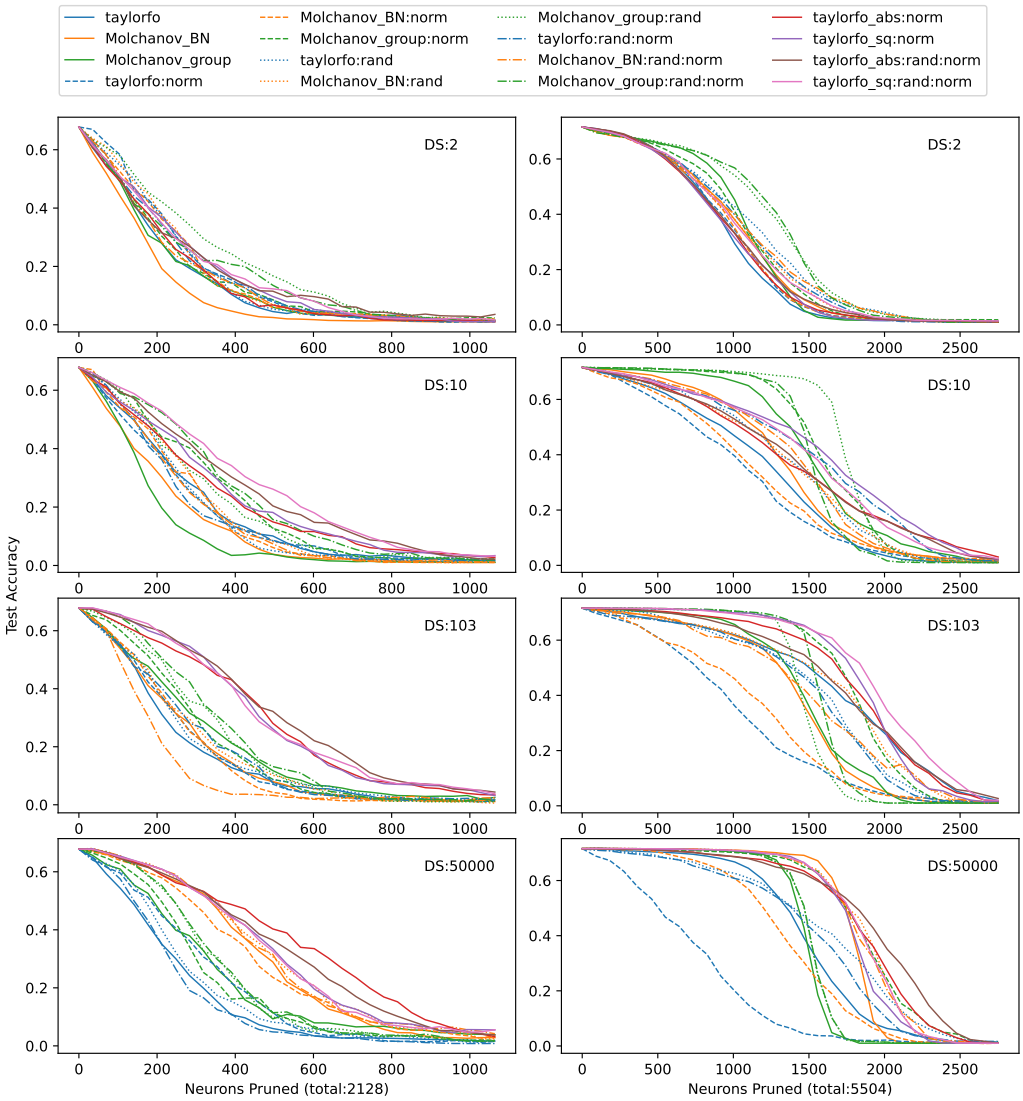

Figure 1: The number of neurons pruned vs Accuracy plot for (left) ResNet-54 and (right) VGG19 for various data sizes (DS). Methods: `taylorfo` Molchanov et al. (2016), `Molchanov_BN` and `Molchanov_group` Molchanov et al. (2019) are the baselines while `taylorfo_sq` and `taylorfo_abs` are *our methods*. `rand` represents use of random gradient and `norm` represents use of gradient magnitude normalization.

Problem 2: The goal of neuron removal is to make the least change in the loss value, which requires the label to calculate the loss. However, according to our intuition, the goal should be to make the least change in the output of the function. Having the least change in the function implicitly fulfils the objective of previous methods to have the least change in the loss. This removes the need to have labels for pruning. Furthermore, viewing from the backward signal, the slope/gradient of the output towards the correct label should not be the requirement, as data points getting output the same as the target will not produce any gradient and hence deemed insignificant by previous methods.

Alternatively, we test the hypothesis that any *random gradient* should work fine, and the gradient should be *normalized* to the same magnitude (of 1). Doing so makes the contribution of each data point equal for computing the importance. We find that the use of random gradient also produces similar pruning performance as shown in Experiment Section Figure 1and 2 supporting our hypothesis.

Our method of using random gradient can be applied to all the previous methods using gradient signal from output to compute the importance. The implication can useful in the setting of unsupervised pruning, where a network can be pruned to target datasets without labels. The application of this method can be significant for settings such as Reinforcement Learning, where the input data is abundant but the label is scarce.

Similarity to Molchanov-BN pruning: Our method is similar to Molchanov - Batch Norm pruning Molchanov et al. (2019) as both of our methods perform neuron/channel level pruning using forward and backward signals.

Molchanov-BN has Importance given by $I_i = \sum(\gamma_i.\delta\gamma_i + \beta_i.\delta\beta_i)^2$, where γ, β_i represents the scaler and bias terms of BN and $\delta\gamma$ and $\delta\beta$ represents their gradient respectively. If we consider $\beta = 0$ then $I_i = \sum(\gamma_i.\delta\gamma_i)^2$.. If we consider the input of batch-norm to be x_i and output after scaling to be $y_i = x_i.\gamma_i$ then the gradient $\delta\gamma_i = \delta y_i.x_i$ and the overall importance is $I_i = \sum(\gamma_i.\delta y_i.x_i)^2$.

In our case, we take the importance $I_i = \sum(x_i.\delta x_i)^2$. Considering the value with respect to δy_i, we get our importance to be $\sum x_i.\gamma.\delta y_i$. Which differs only by the constant shift term β. It turns out that our method produces better accuracy for ResNet style architecture. However, in VGG architecture, our pruning method performs better than Molanchonov-BN and is competitive or better than Molanchonov-WeightGroup when tested in various settings.

3 EXPERIMENTS

Procedure: First, we compute the importance score for each neuron/channel on a given dataset. Secondly, we prune the P least important neurons of total N by importance metric (I) given by different methods. We measure the resulting accuracy and plot the Number of neurons pruned. We test for Taylor method Molchanov et al. (2016; 2019), and on two of our methods. The pruning is performed on the training dataset and accuracy is measured on the test dataset.

Furthermore, to test the performance of different methods on different dataset sizes, we test for D data points of total M data points on all the datasets and architectures. The total dataset size for CIFAR-100 is 50K and for STL-10 is 5K.

CIFAR-100: We test the pruning performance of different methods on CIFAR-100 Krizhevsky et al. (2009) dataset for dataset size $D \in [2, 10, 103, 50K]$ as shown in the Figure 1 2. We test the model for Resnet-20, ResNet-56 and VGG-19.

STL-10: First, we compute the importance score for each neuron/channel on a given dataset. Secondly, we prune the P least important neurons of total N by importance metric (I) given by different methods. We measure the resulting accuracy and plot the Number of neurons pruned. We test for Taylor method Molchanov et al. (2016; 2019), and on two of our methods. The pruning is performed on the training dataset and accuracy is measured on the test dataset.

Furthermore, to test the performance of different methods on different dataset sizes, we test for D data points of total M data points on all the datasets and architectures. The total dataset size for CIFAR-100 is 50K and for STL-10 is 5K.

4 CONCLUSION

In this paper, we improve on previously proposed Taylor First Order based pruning methods Molchanov et al. (2019) by using random gradients and by normalizing the gradients. We show that our techniques improve on the previous methods as well as our own variation. The knowledge that these methods allow for pruning with random gradient backpropagation is the main contribution of our work.

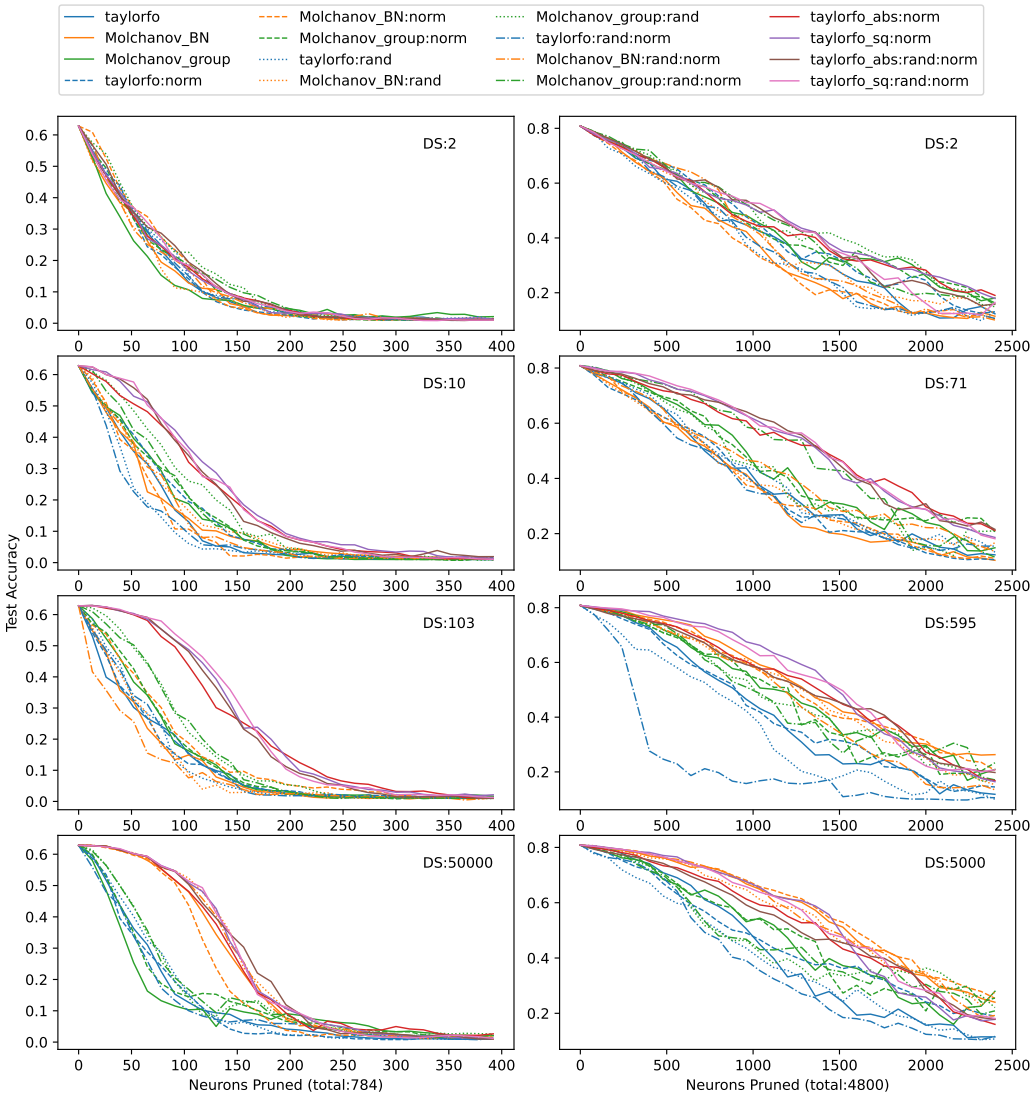

Figure 2: The number of neurons pruned vs Accuracy plot for (left) ResNet-20 on CIFAR-100 and (right) ReNet-18 on STL-10 dataset for various data sizes (DS). Methods: `taylorfo` Molchanov et al. (2016), `Molchanov_BN` and `Molchanov_group` Molchanov et al. (2019) are the baselines while `taylorfo_sq` and `taylorfo_abs` are *our methods*. `rand` represents use of random gradient and `norm` represents use of gradient magnitude normalization.

REFERENCES

Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag. What is the state of neural network pruning? *Proceedings of machine learning and systems*, 2:129–146, 2020.

Xiaoliang Dai, Hongxu Yin, and Niraj K Jha. Incremental learning using a grow-and-prune paradigm with efficient neural networks. *IEEE Transactions on Emerging Topics in Computing*, 10(2):752–762, 2020.

Utku Evci, Yani Ioannou, Cem Keskin, and Yann Dauphin. Gradient flow in sparse neural networks and how lottery tickets win. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 6577–6586, 2022.

Trevor Gale, Erich Elsen, and Sara Hooker. The state of sparsity in deep neural networks. *arXiv preprint arXiv:1902.09574*, 2019.

Ariel Gordon, Elad Eban, Ofir Nachum, Bo Chen, Hao Wu, Tien-Ju Yang, and Edward Choi. Morphnet: Fast & simple resource-constrained structure learning of deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1586–1595, 2018.

Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129:1789–1819, 2021.

Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015a.

Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015b.

Babak Hassibi, David Stork, and Gregory Wolff. Optimal brain surgeon: Extensions and performance comparisons. *Advances in neural information processing systems*, 6, 1993.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Hengyuan Hu, Rui Peng, Yu-Wing Tai, and Chi-Keung Tang. Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. *arXiv preprint arXiv:1607.03250*, 2016.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Vadim Lebedev and Victor Lempitsky. Fast convnets using group-wise brain damage. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2554–2564, 2016.

Guillaume Leclerc, Manasi Vartak, Raul Castro Fernandez, Tim Kraska, and Samuel Madden. Smallify: Learning network size while training. *arXiv preprint arXiv:1806.03723*, 2018.

Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip HS Torr. Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*, 2018.

Shiwei Liu and Zhangyang Wang. Ten lessons we have learned in the new" sparseland": A short handbook for sparse neural network researchers. *arXiv preprint arXiv:2302.02596*, 2023.

Ekdeep Singh Lubana and Robert P Dick. A gradient flow framework for analyzing network pruning. *arXiv preprint arXiv:2009.11839*, 2020.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. *arXiv preprint arXiv:1611.06440*, 2016.

Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. Importance estimation for neural network pruning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11264–11272, 2019.

Michael C Mozer and Paul Smolensky. Using relevance to reduce network size automatically. *Connection Science*, 1(1):3–16, 1989.

Suman Sapkota and Binod Bhattarai. Noisy heuristics nas: A network morphism based neural architecture search using heuristics. *arXiv preprint arXiv:2207.04467*, 2022.

Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. *arXiv preprint arXiv:2002.07376*, 2020.

Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. *Advances in neural information processing systems*, 29, 2016.

Ruichi Yu, Ang Li, Chun-Fu Chen, Jui-Hsin Lai, Vlad I Morariu, Xintong Han, Mingfei Gao, Ching-Yung Lin, and Larry S Davis. Nisp: Pruning networks using neuron importance score propagation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9194–9203, 2018.

Jie Zhang, Junting Zhang, Shalini Ghosh, Dawei Li, Jingwen Zhu, Heming Zhang, and Yalin Wang. Regularize, expand and compress: Nonexpansive continual learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 854–862, 2020.