HIGH SCHOOL EDITION
Journal of Student Research
Volume 11 Issue 1 (2022)

LSOP: Layer-Scaled One-shot Pruning A Simple and Effective Deep Pruning Framework for Neural Networks

Oliver Wang[1], Constantine Dovrolis[#], and Jaeho Lee[#]

[1]Leland High School, San Jose, CA, USA
[#]Advisor

ABSTRACT

Neural network pruning is a technique that removes unnecessary weight parameters from a network to decrease its memory and computational requirements. Many different pruning techniques have been proposed to reduce networks with over 90% shrinkage in size while minimizing accuracy loss. This paper aims to establish a framework that can generalize the mechanism among various pruning techniques, which can be used to guide users to design better deep pruning methods in the future. With some basic concepts and findings from data matrix approximation, the framework can explain the success of the state-of-the-art methods as well as a more generalized pruning design (Layer-Scaled One-shot Pruning, LSOP) proposed in this work. After pruning with different algorithms and measuring their accuracies, the researcher also found that those methods or algorithms aligned with the proposed framework were more accurate at sparser networks (density < 10%) than the methods that did not. This suggests that future research into neural network pruning can focus on the proposed framework, which has the potential to accelerate the development of pruning technology and adoption of more efficient neural networks. The LSOP framework's capability to explain strong pruning performances implies that polynomial decay and Low-Rank Matrix Approximation techniques from the field of data science can provide support for neural network pruning.

1 Introduction

1.1 Motivation and Background

Deep neural networks in machine learning are incredibly useful for computer vision applications like image classification, object detection, and image segmentation. Image classification allows an algorithm to determine what object is within an image, while object detection goes one step further and predicts the location and size of that object. Image segmentation partitions a digital image into multiple segments to locate objects and identify shapes. These technologies have important benefits in the fields such as autonomous vehicles, robotics, medical image system, and smart-image processing edge devices. However, these networks demand heavy memory and computational resources. As there is a noticeable limit to such extensive resources onboard cars, robotics and battery-based edge devices, there are also constraints to the accuracy of the object detection systems.

Pruning, an idea to remove extraneous unnecessary weight parameters to create sparse networks (which has density below 20%), can alleviate the demands for huge storage and computation resources. A recent paper proposed "Lottery Ticket Hypothesis" to redefine pruning to be about locating efficient sub-networks that hold all the critical parameters of a neural network (Frankle et al. 2019). However, finding critical parameters for effective pruning methods can be a time-consuming process that often require both empirical and theoretical methods to verify (Golubeva et al. 2021). With essentially infinite ways to process and modify a neural network to obtain a "winning ticket", the process to find an optimized pruning method may be slow, heuristic and inconsistent. While many at-

tempts at finding pruning methods have been suggested, this project proposes a simple and effective framework to generalize pruning methods by infusing the basic concepts of low rank approximation with polynomial decaying matrices from data science (Tropp et al. 2019).

1.2 Contributions to Gap in Current Research

Currently, the state-of-the-art magnitude pruning method LAMP (Lee et al. 2021) is able to prune over 90% of image classification networks with minimal accuracy loss. The explanation provided for its success does not completely explain why certain modifications (which will be explained in the Methods section of this paper) still allow for high performance.
This paper makes the following contributions to provide a generalized framework to effectively prune the deep neural networks.

- This project infuses the merits of a unique layerwise scaling scheme and the global magnitude pruning into a new scheme, named as Layer Scaled One-shot Pruning (LSOP). LSOP creates crucial proxy scores first, then, applies those scores to global magnitude pruning. LSOP scores provide important information among the weight elements to indicate which ones are more important than others across all layers. Global magnitude pruning based on LSOP scores from all prunable layers generates the proper pruning ratio for each layer.

- The LSOP method incorporates the concept of low-rank approximation for matrix with polynomial-decay property to generalize the property of LSOP, LAMP and other pruning algorithms.

To demonstrate the efficacy of LSOP, various experiments based on different setups, including several popular convolutional neural networks (VGG-16, ResNet-18, EfficientNet-B0) with image dataset (CIFAR-10) are conducted. In those tests, LSOP either performs equally with the latest state-of-the-art (LAMP), (Lee et al. 2021), or outperforms the other baseline layer-based global pruning algorithms.

2 Related Work

2.1 Magnitude Pruning (MP) Schemes

Pruning neural networks is a practice that dates to four decades ago, (survey: Reed 1993), and has seen a recent resurgence (survey: Blalock et al., 2020). The study of magnitude-based pruning, as in the works of LeCun et al. (1989) and Janowsky (1989), has been recently revisited (Frankel et al., 2021). An iterative magnitude pruning (IMP) and retraining was introduced (Han et al. 2016) where a layer-wise pruning threshold is determined by a standard deviation - based on heuristic. IMP can obtain higher accuracy, but with relative inefficiency. If 20% of the connections in each round were removed, it would require ten iterations of retraining to reach a final sparsity of 90%. To improve this issue, subsequent research was proposed in Gradual Magnitude Pruning, where connections were gradually removed over the course of one single round (Narang et al. 2017). Their work was further enhanced by uniform MP with a well-tuned slowly pruning schedule (Zhu et al. 2018). A more recent work improved the pruning scheme with an extra heuristic constraint of completely un-pruning the first convolutional layer and maintaining at most 80% prune rate in the last fully connected layer (Gale et al. 2019).

A global MP scheme based on the Erdős–Rényi method to convolutional neural networks was also proposed (Mocanu et al. 2018). It was further improved it with a training scheme for sparsely initialized neural networks, where the layerwise sparsity is determined by the Erdős–Rényi Kernel (ERK) method (Evci et al. 2020).

On top of the previous works, a more recent study (Lee et al. (2021) proposed Layer-Adaptive Magnitude-based Pruning (LAMP) scores for iterative MP based on the observation that "layerwise MP is the solution of the layerwise minimization of *Frobenius distortion* incurred by pruning, which can be viewed as a *relaxation* of the output L_2 distortion minimization with respect to the worst-case input." This observation leads to their speculation that "reducing the pruning-incurred L_2 distortion of the model output with respect to the worst-case output may be beneficial to the performance of the retrained model." In highly sparse networks (network density < 10%), their results indeed presented the state-of-the-art performance in the iterative mode. Their work also shows good results in a one-shot LAMP scheme (VGG-16 with CIFAR-10) with only 1.09% accuracy degradation at 1.15% of overall network density comparing with iterative LAMP.

According to the authors of Single-Shot Network Pruning SNIP (Lee et al. 2019), one-shot pruning methods avoid "expensive prune-retrain cycles and heuristic design choices with additional hyperparameters". The potential benefits of designing better one-shot pruning methods lead to that improving one-shot magnitude pruning is a valuable area of research that can consolidate the accuracy gains of current MP schemes and the efficiency of One-Shot methods discussed above.

2.2 Low-Rank Matrix Approximation (LRMA)

For data science analysis and scientific simulation, a matrix-data processing scheme was introduced to utilize Low-Rank Matrix Approximation that could reach a near-optimal approximation and significantly reduce the vast amount of streaming data storage and processing (Tropp et al. 2019). If the input data matrix has the property of a "decaying spectrum," the truncated rank-k approximation of the input matrix can achieve a very small relative error compared to the un-truncated approximation. The authors proposed several synthetic decay examples to show the effectiveness of the degree of the "decaying" property. Polynomial decay is one of them and performs well in matrix approximation. Polynomial decay is defined as follows:

For a decay parameter $p > 0$, consider matrices

$$A = diag(1, 2^{-p}, 3^{-p}, \dots, n^{-p}) \qquad \in \ \mathbb{C}^{n \times n}$$

As $p = 1$ or 2, the truncated rank-k (with $k < n$) scheme shows good results in the test results. With the similar methodology on real data sets, it can approximate matrix accurately and with minimal resource usage for scientific simulation and data analysis, such as efficiently processing sea-surface-temperature dataset with the data size of 75 gigabyte (GB) and other large data sets.

3 Preliminaries

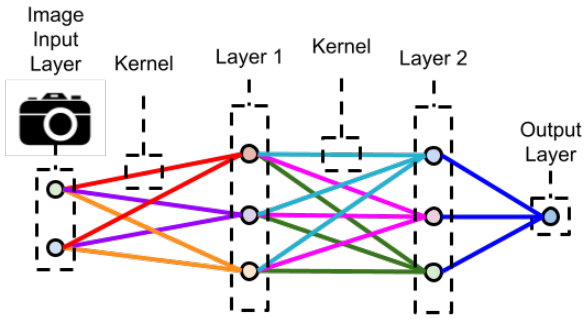

Figure 1. Generic Neural Network Diagram

Information in a neural network begins at an input layer (such as from an image input). Here, two *channels* of information are present at the input, which can be thought of as the first layer of a network.

Each pathway represented by a solid line, known as a kernel, contains many operations of multiplication that use stored numerical "weights". These weights are produced by the neural network and are multiplied with the input of the previous layer at a node for processing, represented by a circle. The initial weight values are determined using "pretraining", which is essentially using weight data from a similar task. As the CNN trains it will tweak the values of these weights to increase its accuracy for the current task. In the final layer, the initial data has been processed to the point where the network can determine what the image is most likely to be. The number of nodes and pathways here has been greatly reduced for simplicity; most common CNNs use many more.

The following is a representation of the element-wise multiplication that occurs within each node. The red squares are kernels (whose elements are weights), corresponding to the red lines of Fig 1.

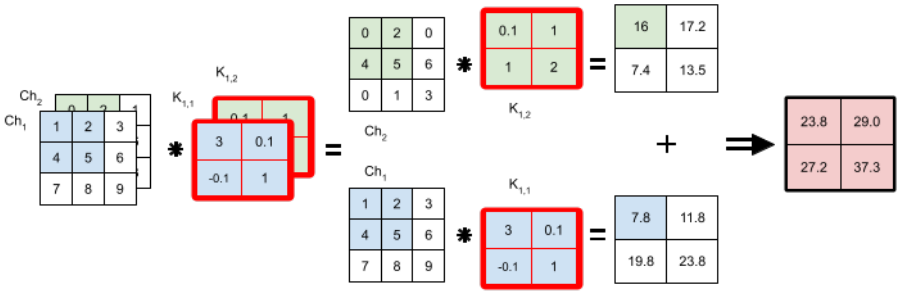

Figure 2. Weights in a CNN

Two channels of information (Ch_1 and Ch_2) separately undergo *element-wise multiplication* by kernels $K_{1,1}$ and $K_{1,2}$. As in Figure 1, the kernels are represented with a red border, and each contain four weights. Element-wise multiplication involves the following combinations of multiplication and addition:

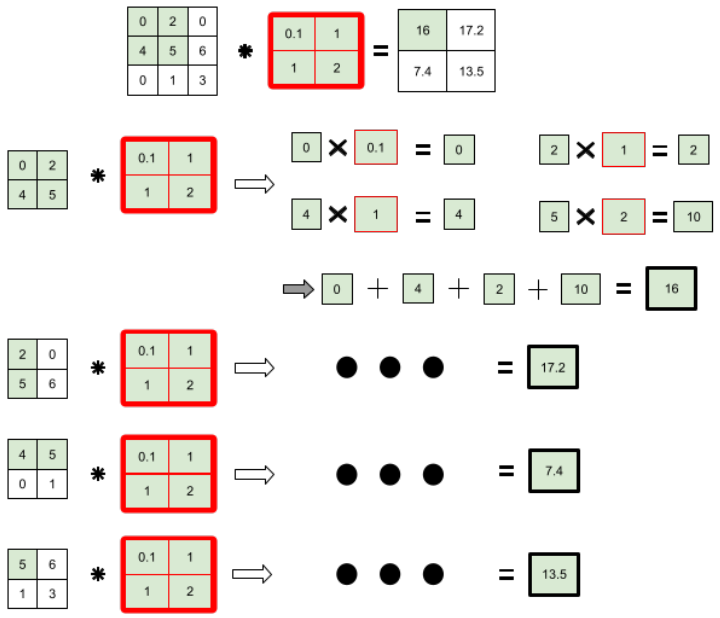

Figure 3. Detailed Element-Wise Multiplication

Here, the * symbol represents element-wise multiplication of two matrices, while the × is the multiplication of two numbers.

The number of weights in a neural network (sometimes on the order of tens of millions) places a considerable strain on memory requirements. Pruning aims to simplify the multiplication along the pathways. Magnitude pruning, which this paper focuses on, approximates weights as zeros if they have a small enough magnitude, significantly reducing the required memory.

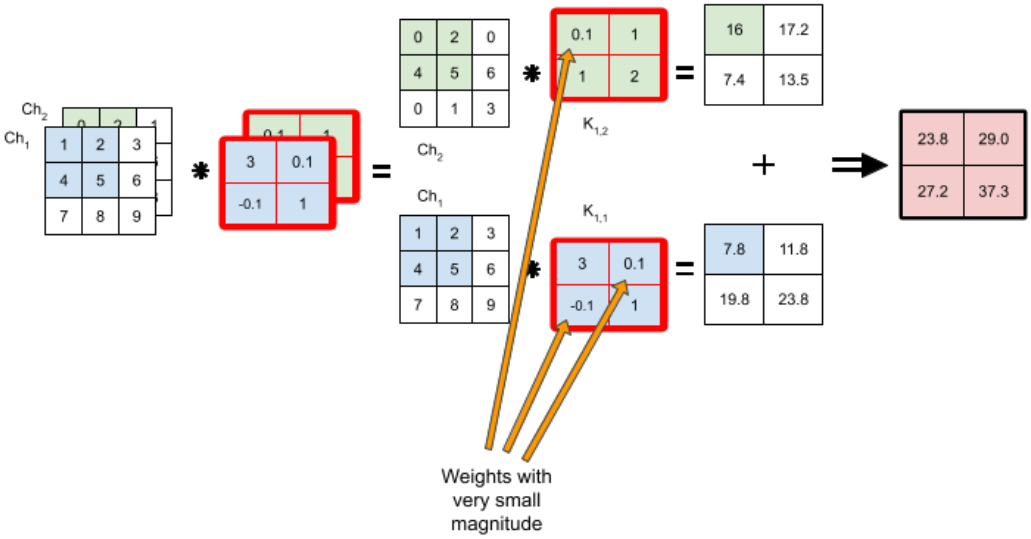

Figure 4. Identification of Weights

The weights in each kernel with sufficiently small magnitudes (as determined by the pruning method) have been identified. If the weights marked in Fig. (4) were to be approximated as zeros, as in Fig. (5), three out of the eight weights pictured would not need to be stored as memory.

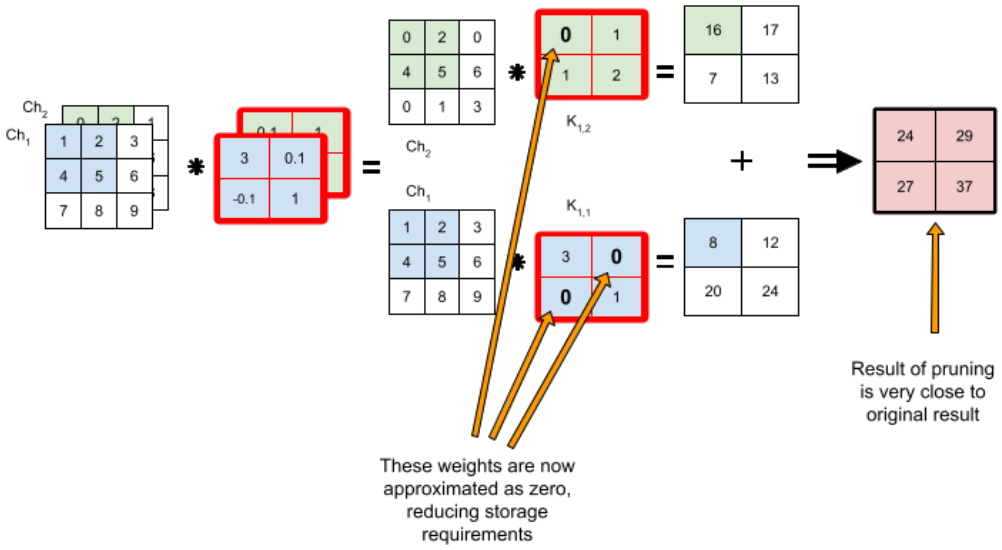

Figure 5. Result After Pruning

Upon inspection, the result of both the unpruned and pruned diagrams are very similar. The effectiveness of pruning relies on those results being sufficiently similar in conjunction with lower system memory requirements. Each different pruning method assigns "zeros" in a different way. "Magnitude pruning" is the practice of taking into account the absolute value of each element. "Global magnitude pruning" analyzes all elements in a neural network, and prunes out the lowest x percent of weights by magnitude, where x is a user-specified amount. This strategy is among the simplest types of pruning, and so it frequently used as a baseline (Blalock et al., 2020).

4 Proposed Method: Layer-Scaled One-shot Pruning (LSOP)

This paper aims to introduce a new framework to deeply prune neural networks that infuses the basic concept of polynomial decay property in LRMA to generate the LSOP scores for the commonly used global magnitude pruning. The procedures are shown as follows:

4.1 Design of Layer-Scaled One-shot Pruning (LSOP)

Consider a feedforward neural network with a depth of L layers and weight tensors of ${}^{(1)}W, {}^{(2)}W, \ldots, {}^{(L)}W$ corresponding to each convolutional or fully connected layer. Each 2D convolutional layer has a four-dimensional tensor, and each fully connected layer has a two-dimensional tensor. The tensor in each layer is flattened into a one-dimensional vector, and its weight elements are sorted in descending order such that $|w[i]| \leq |w[j]|$ holds whenever $i > j$, where $w[i]$ is the weight of the i-th element in that flattened vector.

The LSOP score for the i-th element, $w[i]$, is defined as
Equation 1:

$$LSOP(w[i]) = Score(w[i]) = \frac{LsA(i)}{LsB(i)} \quad (1)$$

LsA(*i*) is a scaling factor with a polynomial relationship to the magnitude of *w*[*i*], such as |*w*[*i*]| or |*w*[*i*]|^2, or |*w*[*i*]|^3. *LsB*(*i*) is related to the summation of cumulative elements with respect to the index. Together, LSOP can exhibit "polynomial decaying value" behavior.

A candidate for *LsA*(*i*) can be |*w*[*i*]| sorted in the descending order. Therefore,
Equation 2:

$$LsA(i) = |w[i]| \leq |w[1]| = LsA(1) \qquad when\ i \geq 1 \quad (2)$$

A choice for *LsB*(*i*) is the partial sum *pSum*(*i*) of the weight magnitudes from the 1st element up to the *i*-th element, which is defined and proved as follows:
Equation 3:

$$LsB(i) = pSum_1(i) = \sum_{i \geq k \geq 1} |w[k]| \quad (3)$$

Equation 4:

$$\text{defining} \quad a(k) = \frac{|w[k]|}{|w[i]|}, \qquad \text{then } a(k) \geq 1, \qquad \text{so} \sum_{i \geq k \geq 1} a(k) \geq i. \quad (4)$$

Combining (1), (2), (3), and (4) to define a case of LSOP as "LSOP$_1$",
Equation 5:

$$LSOP_1(w[i]) = \frac{LsA_1(i)}{LsB_1(i)} = \frac{|w[i]|}{\sum_{i \geq k \geq 1} |w[k]|} = \frac{1}{\sum_{i \geq k \geq 1} a(k)} \leq \frac{1}{i} = i^{-1} \quad (5)$$

The above derivation shows that if $LsA_1(i)$ is chosen as the magnitude of the weight, and $LsB_1(i)$ is defined as the partial sum of values from |*w*[1]| to |*w*[*i*]|, then, LSOP$_1$(*w*[*i*]) has the "polynomial decaying value" property similar to $k^{(-1)}$, as the upper bound decaying properly. LSOP$_1$ is used as the LSOP scores for the subsequent experimental tests.

Likewise, *LsA*(*i*) and *LsB(i)* can also be chosen as follows, when *i* ≥ 1:
Equation 6:

$$LsA_2(i) = |w[i]|^2 \leq |w[1]|^2 = LsA_2(1) \quad (6)$$

Equation 7:

$$LsB_2(i) = pSum_2(i) = \sum_{i \geq k \geq 1} |w[k]|^2 \geq pSum_2(1) = LsB_2(1) \quad (7)$$

Equation 8:

$$since \qquad a(k) = \frac{|w[k]|}{|w[i]|} \geq 1, \qquad so, \sum_{i \geq k \geq 1} a^2(k) \geq i * a^2(k) \geq i \quad (8)$$

Combining (1), (6) to (8),
Equation 9:

$$LSOP_2(w[i]) = \frac{LsA_2(i)}{LsB_2(i)} = \frac{|w[i]|^2}{\sum_{i \geq k \geq 1} |w[k]|^2} = \frac{1}{\sum_{i \geq k \geq 1} a^2(k)} \leq \frac{1}{i} = i^{-1} \quad (9)$$

The above derivation shows that if $LsA_2(i)$ is chosen as $|w[i]|^2$, the square of the magnitude of the weight, and $LsB_2(i)$ is defined as the partial sum of values from $|w[i]|^2$ to $|w[1]|^2$, then, $LSOP_2(w[i])$ also has the "polynomial decaying value" property. Meanwhile, it can be shown that $LSOP_2$ is also equal to LAMP score (Lee et at., 2021) as follows:

Equation 10:

$$LSOP_2(w[i]) = \frac{LsA_2(i)}{LsB_2(i)} = \frac{|w(i)|^2}{\sum_{i \geq k \geq 1} |w[k]|^2} = \frac{\left(w(i)\right)^2}{\sum_{i \geq k \geq 1} \left(w(k)\right)^2} = LAMP\ score(i\,; w) \quad (10)$$

While LAMP uses a completely different approach based on the assumption of iterative pruning scheme to generate the same scores as $LSOP_2$, the derivation of LSOP does not require information of whether the pruning is iterative or one-shot.

$LSOP(w[i])$ score can be viewed as a normalized score for an element $w[i]$ in a given layer which is scaled by $LsA(i)$ and LsB(i). There could be many different ways to choose $LsA(i)$ and $LsB(i)$ besides the two examples discussed above. Although it appears simple and straightforward, LSOP scores along with the subsequent global magnitude pruning provide a distinctive way to differentiate the importance of each connection or weight according to its in-layer scaling effect and the overall global ranking.

Borrowing the basic concept of truncated rank-k approximation with polynomial decaying property of the elements from LRMA, LSOP scores readily provide the top-k ranking information among all the weight elements within one layer which can be used for optimal pruning later. The element with the largest magnitude ($|w[1]|$) in each layer is used as a base reference and its LSOP score is always equal to 1 (LSOP($w[1]$) = 1). After each element acquires its LSOP score, global magnitude pruning is then applied to remove the weight elements with LSOP scores lower than a threshold based on the target pruning ratio or network density. This methodology has two important implications:

1) Since max{LSOP()} = 1 for the largest weight magnitude element in each layer, each layer at least keeps one element unpruned, which prevents layer collapse from significantly degrading the accuracy performance (Tanaka et al, 2020).

2) The scaling factors, $LsA(w(i))$ and $LsB(w(i))$ achieve a polynomial-decay effect that matches with polynomial decay in LRMA. The factors are first used to re-scale each element within a layer to generate its LSOP score. The first few $LsB(w(i))$s are the dominant factors for the in-layer scaling. Such as, the larger the $LsB(w(1))$ (=$|w[1]|$) is, the more the rest of the weights in that layer are de-emphasized. This property also affects the following global-pruning results and could prevent those essential weight elements from being pruned away. For example (without loss of generality), if three weight elements need to be chosen to be preserved from the four weights in arbitrary two layers, L_1 and L_2

$$L_1 = \begin{bmatrix} 4 \\ 2.5 \end{bmatrix}, \quad L_2 = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

Using a traditional global pruning (solely based on the absolute magnitude ranking), {4, 3, 2.5} would be selected. However, if $LSOP_1$ or $LSOP_2$ score is used with the global magnitude pruning, then, {4, 3, 2} would be chosen because the following LSOP scores are calculated:

$$LSOP_1(L_1) = \begin{bmatrix} 4/4 \\ 2.5/(4+2.5) \end{bmatrix} = \begin{bmatrix} 1 \\ 0.385 \end{bmatrix}, \quad LSOP_1(L_2) = \begin{bmatrix} 3/3 \\ 2/(3+2) \end{bmatrix} = \begin{bmatrix} 1 \\ 0.4 \end{bmatrix}$$

$$LSOP_2(L_1) = \begin{bmatrix} 4^2/4^2 \\ 2.5^2/(4^2+2.5^2) \end{bmatrix} = \begin{bmatrix} 1 \\ 0.28 \end{bmatrix}, LSOP_2(L_2) = \begin{bmatrix} 3^2/3^2 \\ 2^2/(3^2+2^2) \end{bmatrix} = \begin{bmatrix} 1 \\ 0.31 \end{bmatrix}$$

The above example can be generalized to multiple elements across multi-layers. This result may seem counter-intuitive at first. However, in neural network multi-layer formation, compared with traditional global pruning, LSOP scores plus global pruning techniques provide extra crucial scaling information about which elements are more important than others and should be preserved. Especially, in a network of deep pruning rate > 90% (network density < 10%), this property becomes more prominent and can be seen from the comparison among different previous Global Magnitude Pruning (GMP) methods and the proposed LSOP in the following experimental results.

4.2 LSOP with Global Magnitude Pruning

After LSOP scores are generated for all trainable weight elements in the whole network, a typical global pruning is performed as follows:

The LSOP scores in all layers are concatenated and sorted to form a ranked list. The top-*K* elements of the list are preserved based on the following equation:

K = (Pruning ratio) * (total number of elements in the whole neural network).

The total top-*K* elements are the summation of the top-*K*(n) elements for individual n-th layer. Such top-*K*(n) values are used to preserve those crucial weight elements in the n-th layer, while the rest of elements are pruned away to form a target sparse network.

5 Experiments and Analyses

To validate the effectiveness of the proposed scheme, **LSOP** is compared with the following state-of-the-art baseline layerwise magnitude-based pruning algorithms.

- **Global.** A target global magnitude pruning (GMP) ratio is defined to specify the overall network density. Then, each layer density is automatically generated based on this ratio through the algorithm described in the research (Morcos et al., 2019).

- ***Erdős–Rényi-Kernel* (ERK).** Google AI group proposed this modified version of *Erdős–Rényi* algorithm (Evci et al. 2020), originally published by Mocanu et al. 2018) for pruning convolutional layers in neural networks. The numbers of pruned elements on sparse convolutional layers are scaled proportional to

$$\frac{n^{L-1} + n^L + w^L + h^L}{n^{L-1} \cdot n^L \cdot w^L \cdot h^L} \quad (10)$$

  where n^{L-1} is the number of the output neurons at layer L-1, while n^L is the number of the output neurons at layer L, and w^L, h^L are the width and the height of the convolutional kernel at layer L.

- **LAMP.** This algorithm minimizes the L_2 distortion caused by layerwise magnitude pruning (Lee et al. 2021). Although, LAMP was originally derived based on the assumption of Iterative Magnitude Pruning (IMP) mechanism, it also performed well under one-shot pruning scheme.

5.1 Experiment Procedures

Figure 1 illustrate the main flow chart of experiment. The training flow starts with randomized Initialization (using different seeds) followed by PreTraining to generate the pretrained weights matrix, which is fed into different One-shot Pruning algorithms to create the pruned weight mask. Then the weight matrix and the pruned weight mask are fed into Training process to produce the final weight matrix. Such final weight matrix is used for inference and tested for the accuracy.

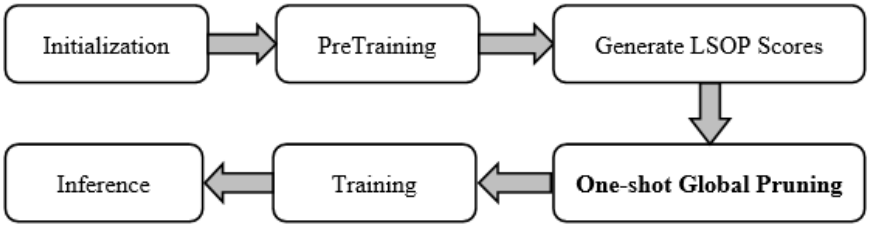

Figure 6. Flow Chart of LSOP Pruning Experiment

The method by which these algorithms are compared is as follows:

First, LSOP and the three baseline pruning schemes take five independent trials. Each trial uses a different seed. Each trial is one-shot pruning followed by training once.

Then, the results are plotted on graphs comparing test accuracy versus predetermined network density (= 1 - prune percentage) from 0.5% to 16%. Dash lines are the individual trials and solid lines are the averages of the five individual trials for the corresponding cases.

5.2 Experiment Setup

The following main setups are used for the experiments based on the work from Lee et al. (2021):

- Five independent trials with seeds of 7, 11, 59, 83 and 111
- Three neural network models: VGG16, ResNet18, and EfficientNet-B0
- Image dataset of CIFAR-10
- Network densities of 1%, 2%, 4%, 8%, and 16% are used for all networks, while more finer points are added for VGG16 and ResNet18.
- Optimizer: vanilla AdamW (Loshchilov & Hutter, 2019) with learning rate 0.0003
- Hyperparameters: PyTorch default setup: $\beta = (0.9, 0.999)$, wd = 0.01, $\varepsilon = 10$
- Pre-processing: CIFAR-10 dataset is augmented with random crops with a padding of 4 and random horizontal flips, and then normalization with constants (0.4914, 0.4822, 0.4465), (0.237, 0.243, 0.261).

5.3 Experiment Results and Observations

Five trials of pruning were run for each pruning scheme (e.g., LSOP, LAMP, etc.) on each network (e.g., VGG16, ResNet18, etc.). Fig. 7-10 shows accuracy comparison between LSOP and other prior art GMP methods when a

given network is deeply pruned with less than 20% density left. Each of the five independent trials is shown as a dotted line on the graphs, while a solid line shows the mean of the trials.

From the experimental results in Fig. (2), LSOP ($LSOP_1$) and LAMP (same as $LSOP_2$) are statistically comparable with only $\pm$ 0.2% deviation of accuracy from each other. From Fig. (3)-(5), LSOP and LAMP outperform ERK and Global. LSOP shows significant better test accuracy (>8%) than that of Global pruning when network density < 1%. This result highlights the implication (2) in section 3.2 that the relative ratio/weighting among elements in the same layer is more important than the absolute difference among elements across different layers.

In EfficientNet-B0, a neural network structure developed by the Google AI team and published in 2019, where the traditional convolutional layers are replaced by mobile inverted bottleneck convolutional layers, Global pruning is substantially worse than the other pruning methods when the network density is < 5%.

Overall, the results demonstrate that LSOP has compatible or even better achievement in accuracy than other leading methods for the sparse networks.

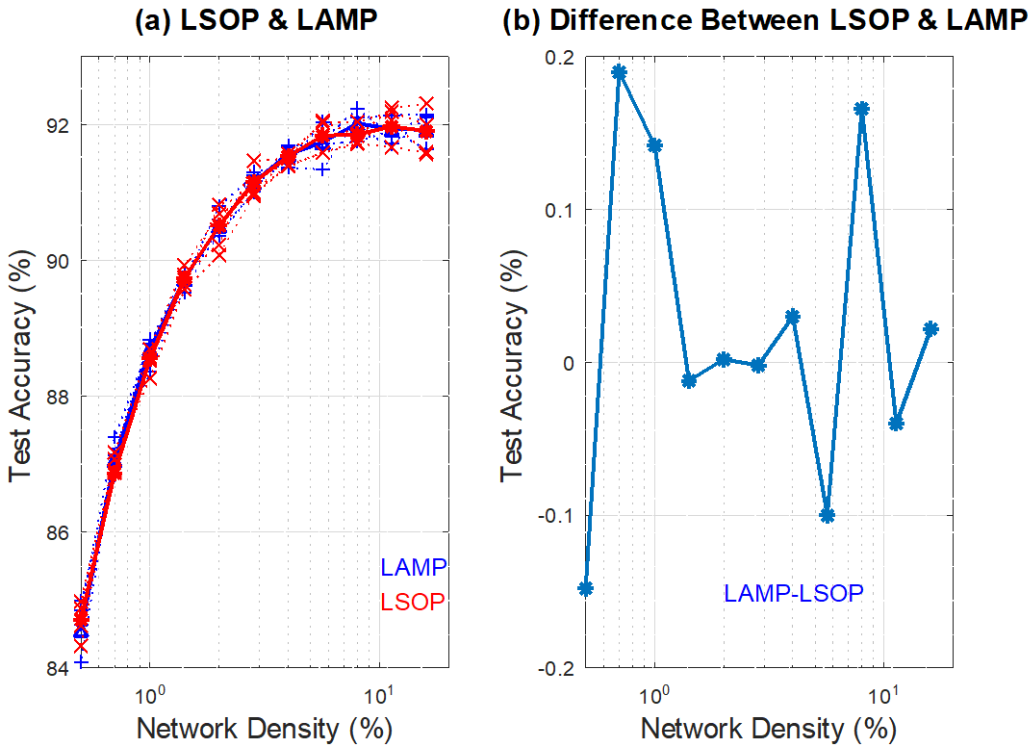

Figure 7. Accuracy Comparison between LSOP and LAMP on VGG-16 Neural Network

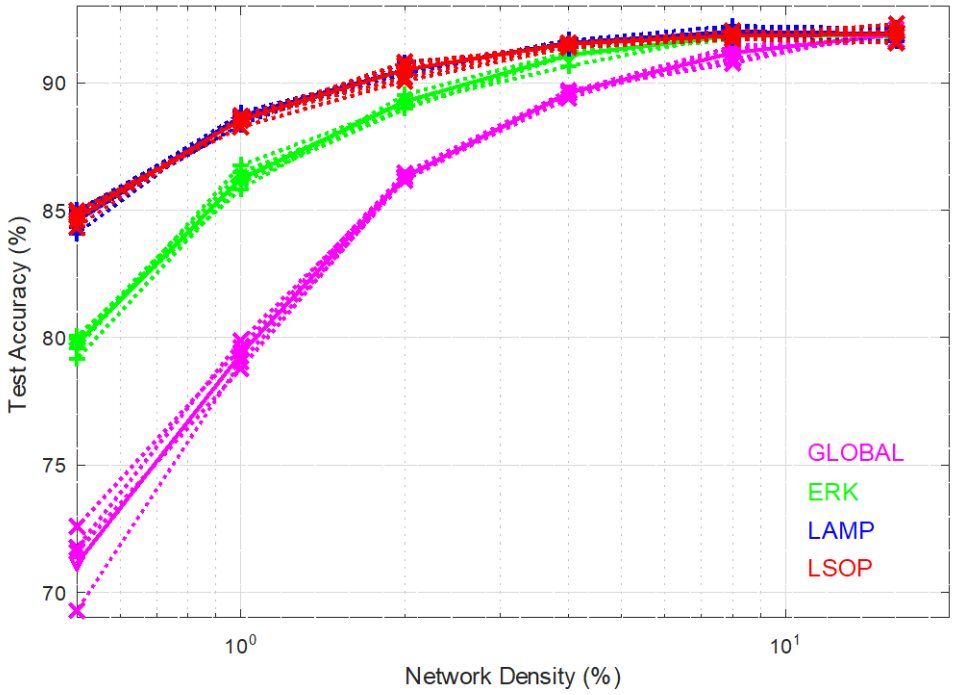

Figure 8. Accuracy Comparison among Four Pruning Methods on VGG-10 Neural Network.

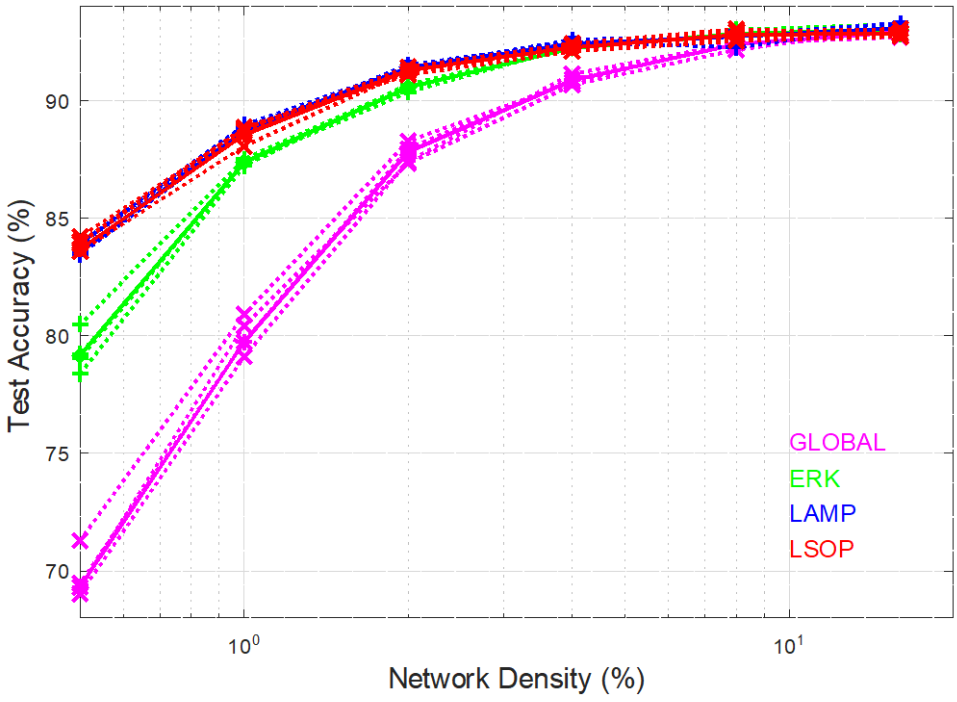

Figure 9. Accuracy Comparison among Four Pruning Methods on ResNet18 Neural Network.

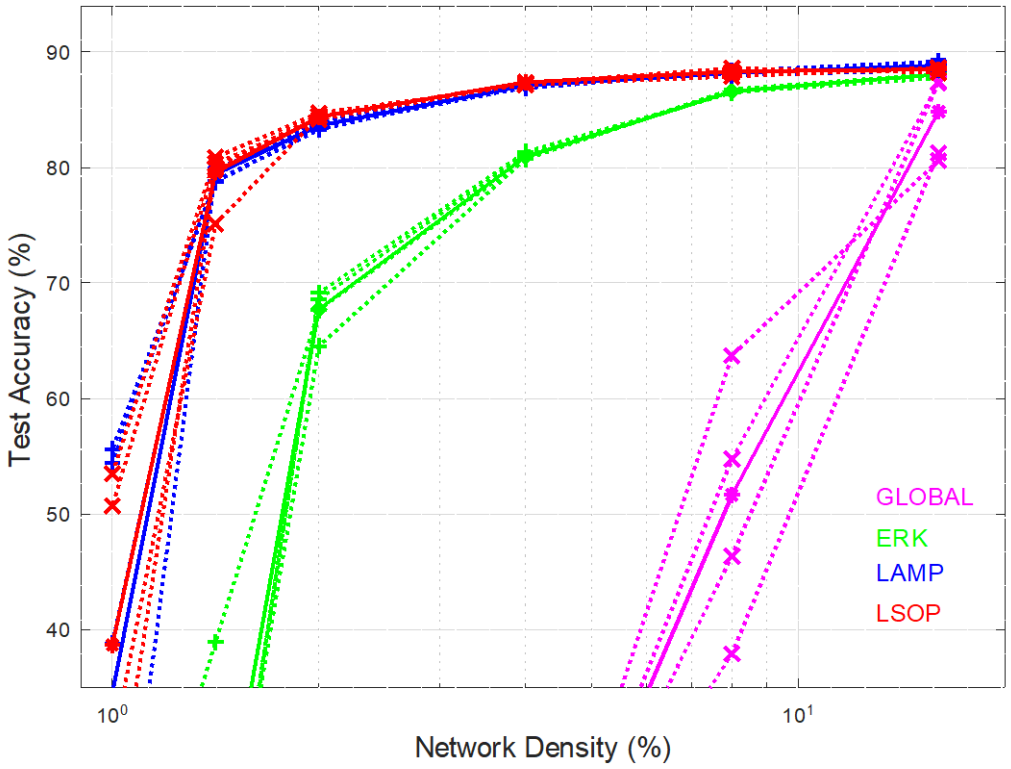

Figure 10. Accuracy Comparison among Four Pruning Methods on EfficientNet-B0 Neural Network.

Table 1 shows that LSOP has a smaller standard deviation in accuracy, which indicates that LSOP is a robust scheme that could be implemented in applications where costly and time-consuming iterative pruning is not desired.

Neural Network	Pruning Scheme	Accuracy Standard Deviation (%)
VGG-16	GLOBAL	5.22
	ERK	2.28
	LAMP	1.54
	LSOP	1.39
ResNet-18	GLOBAL	5.44
	ERK	2.34
	LAMP	1.77
	LSOP	2.06
EfficientNet-B0	GLOBAL	35.3
	ERK	20.3
	LAMP	3.88
	LSOP	3.59

Table 1. Standard Deviation of Accuracy in Multiple Trials with Different Networks and Pruning Schemes

6 Conclusion

In this project, a unique framework, LSOP (Layer-Scaled One-shot Pruning), is developed and evaluated on different networks with a CIFAR-10 image dataset.

Borrowed from data science and mathematics, basic concepts of truncated low-rank approximation with polynomial decay are incorporated to develop the theoretical framework for more efficient pruning schemes in deep neural networks. This crucial aspect implies that modifying neural networks to exhibit polynomial decay (i.e. assigning LSOP scores to each element) as a form of low rank matrix approximation is a promising way to develop pruning methods.

Additionally, LSOP provides the option to easily choose different subsets of pruning schemes, like $LSOP_1$ and $LSOP_2$. Though $LSOP_2$ is equivalent to LAMP, the latter is developed from a completely difference approach. which outperform other state-of-the-art magnitude pruning algorithms, like ERK and Global magnitude pruning. LSOP also sheds light on an obscure nature of neural networks: the relative weight ratios among weight elements within the same layer also play an important role besides the absolute weight differences among elements across different layers.

7 Future Works

More rigorous research can be conducted on many aspects. Further exploration of the role of polynomial decay in pruning and the relationship between data science and neural networks can be done. Perhaps this particular strategy holds promise for the development of neural network pruning.

Using more image datasets, like Tiny-ImageNet, CIFAR-100, etc., and more different neural networks, like ResNet-50, DenseNet-121/201 to analyze the pruning schemes for image classification.

If higher accuracy is preferrable, multiple pruning-training process can be included to improve test accuracy based on LSOP framework, such as iterative magnitude pruning (IMP), or gradual magnitude pruning.

Since the LSOP framework is not limited by the types of the neural networks and applications, more complicated structures, like YOLO, can also be tested for image classification and object detection in the future.

8 Acknowledgments

I would like to thank Dr. Constantine Dovrolis for his guidance, support, review, and valuable feedback. I also wish to thank Dr. Jaeho Lee for his sharing of open-source code published in GitHub, discussion, review, and helpful feedback to this project.

References

X. Dong, S. Chen, and S. J. Pan. Learning to prune deep neural networks via layer-wise optimal brain surgeon. In *Advances in Neural Information Processing Systems*, 2017.

E. Elsen, M. Dukhan, T. Gale, and K. Simonyan. Fast sparse ConvNets. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

U. Evci, T. Gale, J. Menick, P. S. Castro, and E. Elsen. Rigging the lottery: Making all tickets winners. In *Proceedings of International Conference on Machine Learning*, 2020.

J. Frankle and M. Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019.

T. Gale, E. Elsen, and S. Hooker. The state of sparsity in deep neural networks. In *ICML ODML- CDNNR Workshop*, 2019.

T. Gale, M. Zaharia, C. Young, and E. Elsen. Sparse GPU kernels for deep learning. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis*, 2020.

S. Han, J. Pool, J. Tran, and W. Dally. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems*, 2015.

Y. LeCun, J. S. Denker, and S. A. Solla. Optimal brain damage. In *Advances in Neural Information Processing Systems*, 1989.

J. Lee, S. Park, S. Mo, S. Ahn and J. Shin. Snip: Layer-Adaptive Sparsity For the Magnitude-Based Pruning. In *International Conference on Learning Representations*, 2019.

N. Lee, T. Ajanthan, and P. H. S. Torr. Snip: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations*, 2019.

Z. Liu, M. Sun, T. Zhou, G. Huang, and T. Darrell. Rethinking the value of network pruning. In *International Conference on Learning Representations*, 2019.

D. C. Mocanu, E. Mocanu, P. Stone, P. H. Nguyen, M. Gibescu, and A. Liotta. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. *Nature Communications*, 2018.

A. S. Morcos, H. Yi, M. Paganini, and Y. Tian. One ticket to win them all: Generalizing lottery ticket initializations across datasets and optimizers. In *Advances in Neural Information Processing Systems*, 2019.

M. C. Mozer and P. Smolensky. Skeletonization: A technique for trimming the fat from a network via relevance assessment. In *Advances in Neural Information Processing Systems*, 1988.

B. Mussay, M. Osadchy, V. Braverman, S. Zhou, and D. Feldman. Data-indenpendent neural pruning via coresets. In *International Conference on Learning Representations*, 2020.

B. Neyshabur, R. Tomioka, and N. Srebro. Norm-based capacity control in neural networks. In *Conference on Learning Theory*, 2015.

S. Park, J. Lee, S. Mo, and J. Shin. Lookahead: A far-sighted alternative of magnitude-based pruning. In *International Conference on Learning Representations*, 2020.

A. Renda, J. Frankle, and M. Carbin. Comparing fine-tuning and rewinding in neural network pruning. In *International Conference on Learning Representations*, 2020.
V. Sanh, T. Wolf, and A. M. Rush. Movement pruning: Adaptive sparsity by fine-tuning. arXiv preprint 2005.07683, 2020.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

M. Tan and Q. V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *Proceedings of International Conference on Machine Learning*, 2019.

J. Tropp, A. Yurtsever, M. Udell, and V. Cevher. Streaming Low-Rank Matrix Approximation with an Application to Scientific. SIAM J. Scientific Computing, 2019.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

X. Xiao, Z. Wang, and S. Rajasekaran. AutoPrune: Automatic network pruning by regularizing auxiliary parameters. In *Advances in Neural Information Processing Systems*, 2019.

C. Yun, S. Sra, and A. Jadbabaie. Small ReLU networks are powerful memorizers: a tight analysis of memorization capacity. In *Advances in Neural Information Processing Systems*, 2019.

H. Zhou, J. Lan, R. Liu, and J. Yosinki. Deconstructing lottery tickets: Zeros, signs, and supermasks. In *Advances in Neural Information Processing Systems*, 2019.

M. Zhu and S. Gupta. To prune, or not to prune: exploring the efficacy of pruning for model compression. In *ICLR Workshop track*, 2018.