Report: Advancing Neural Network Pruning

Dr. Ahmad Mousavi, Dwanith Venkat Girish

American University
DATA - 793 - 001 (Data Science Practicum)

Abstract

This capstone project explores structured pruning techniques specifically L1 and L2 norm-based filter pruning applied to widely used convolutional neural network (CNN) architectures: ResNet and VGG. The primary objective is to investigate the trade-off between model sparsity and classification performance on the CIFAR-100 dataset. The study begins with baseline training of ResNet (18, 34, 50, 101) and VGG (11, 13, 16, 19) models, followed by iterative pruning at multiple compression levels (10%, 20%, 30%, and 40%). Later, fine-tuning is applied to recover accuracy loss post-pruning. All experiments are implemented using PyTorch and only python code, with accuracy and parameter changes evaluated systematically. Results reveal that moderate pruning (up to 10%) retains competitive performance, while aggressive pruning may degrade model accuracy significantly. This project provides insights into the pruning capabilities of different CNN architectures and serves as a foundation for further research in model compression.

1. Introduction

As CNNs have demonstrated remarkable performance in the recent years in the field of vision, however the increasing complexity of the new architectures make the training and the deployment process slower than the smaller architecture counterparts. In order to counter the issue of slow training, the process of structural pruning has been implemented on the filter level in the convolutional layers across multiple pruning percentage (%) thresholds (10%, 20%, 30%, 40%).

2. Literature Review

Several pruning techniques have been explored in recent literature to reduce the computational cost of CNNs without significantly compromising performance. Han et al. (2015)[1] introduced weight pruning to eliminate redundant connections. Li et al.[2] (2017) proposed L1-norm-based filter pruning, where filters with smaller L1-norms are removed, which served as a foundation for our L1 pruning approach. He et al. (2019)[3] extended structured pruning by incorporating channel importance using L2-norm. More recently, Yang et al. (2022)

[1] **Han, S., Pool, J., Tran, J., & Dally, W.** (2015). *Learning both weights and connections for efficient neural network*. Advances in Neural Information Processing Systems, 28. https://papers.nips.cc/paper_files/paper/2015/file/ae0eb3eed39d2bcef4622b2499a05fe6-Paper.pdf

[2] **Li, H., Kadav, A., Durdanovic, I., Samet, H., & Graf, H. P.** (2017). *Pruning filters for efficient convnets*. International Conference on Learning Representations (ICLR). https://arxiv.org/abs/1608.08710

[3] **He, Y., Liu, P., Wang, Z., Hu, Z., & Yang, Y**. (2019). *Filter pruning via geometric median for deep convolutional neural networks acceleration*. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 4340–4349. https://doi.org/10.1109/CVPR.2019.00448

introduced RL-Pruner, a reinforcement learning-based structured pruning method that learns a pruning policy

3. Methodology

3.1 Dataset

In this study, the dataset which was used is CIFAR-100[4], a well known benchmark for image classification tasks. The dataset consists of 60,000 32*32 coloured images divided into 100 classes or categories. The data is pre-divided into 50,000 training images and 10,000 test images following the standard training/testing split for neural network models.

3.2 Models

This project focuses on two popular convolutional neural network architectures: ResNet and VGG, both of which are widely used in image classification tasks. For ResNet models, ResNet18, ResNet34, ResNet50, and ResNet101 was used, while for VGG models, VGG11, VGG13, VGG16, and VGG19 was employed.

3.3 Approach

Initially, the experiment was started by implementing RL-Pruner[5] method, where the structural pruning is conducted using Reinforcement Learning on the convolutional layers based on the "SAR" method, which essentially means that the agent in the Reinforcement Learning method works on the 'S' - situation, 'A' - action (pruning) and 'R' - Reward system, due to lower performance accuracy with this approach, it was decided to change pruning methods such as L1 and L2 norm based pruning methods, the details of the two methods are described further.

3.4 Training Baseline Models

For both ResNet and VGG, the authors in the RL-Pruner[6] paper had used the following, the training process of all baseline CNN models was conducted using the SGD optimizer with a momentum of 0.9 and a weight decay of 5e-4. A cosine annealing learning rate scheduler was used to adjust the learning rate throughout the training process. The loss function employed was cross-entropy loss, which is standard for multi-class classification tasks.

3.5 L1 pruning

The first technique of pruning which was explored in this experiment. In this method, the filters with the smallest L1 norm are considered less important and are pruned. This method assumes that filters with lower magnitude contribute less to the output, and hence are removed as a part of the L1 pruning process. The formula of L1 pruning is mentioned below for further clarity.

---

[4] **Krizhevsky, A.** (2009). *Learning multiple layers of features from tiny images* (Technical Report). University of Toronto. https://www.cs.toronto.edu/~kriz/cifar.html

[5] **Wang, B., & Kindratenko, V.** (2024). *RL-Pruner: Structured pruning using reinforcement learning for CNN compression and acceleration* [Preprint]. arXiv. https://arxiv.org/abs/2411.06463

[6] **Wang, B., & Kindratenko, V.** (2024). *RL-Pruner: Structured pruning using reinforcement learning for CNN compression and acceleration* [Preprint]. arXiv. https://arxiv.org/abs/2411.06463

$$\|x\|_1 = \sum_{i=1}^{n} |x_i|$$

3.6 L2 pruning

L2 pruning is the second pruning technique explored in this project. Unlike L1 pruning, which treats all weight magnitudes equally, L2 pruning gives more weightage for larger values within each filter. It calculates the L2 norm (Euclidean norm) of each filter, defined as the square root of the sum of squares of all weights in the filter. Filters with the smallest L2 norms indicating low overall energy are considered less significant and are removed. This method tends to retain filters with a few large weights, making it less aggressive and more stable. The formula of L1 pruning is mentioned below for further clarity.

$$\|W\|_2 = \sqrt{\sum_{i=1}^{n} w_i^2}$$

4. Results and Discussion

The results are summarized in the form of an average of subcategories (ResNet and VGG), in addition the results will be displayed numerically showing specific details of the pre-pruning (baseline), pruning and post-pruning phase of results.

If we analyze and compare the performance of both L1 and L2 norm based pruning across various ResNet and VGG models. The accuracy of each model was recorded at four pruning levels: 10%, 20%, 30%, and 40%, and the models were fine-tuned post-pruning.

4.1 Fine-Tuning

During the fine-tuning phase post-pruning, a simplified setup was adopted with reduced epochs (typically 10), a lower learning rate (0.01), and the SGD optimizer was used to efficiently retrain the pruned models while maintaining consistency across experiments.

4.2 Baseline Accuracies

The baseline accuracies for both ResNet and VGG models are as follows:

Variant	Baseline Accuracy (%)
ResNet18	76.68
ResNet34	77.53
ResNet50	79.63
ResNet101	80.17
VGG11	68.44
VGG13	72.60
VGG16	72.47
VGG19	72.70

Based on the above displayed accuracy levels, we notice a higher average across ResNet models, as compared to the average accuracy across VGG models.

4.3 L1 Pruned Results/Performance

Post-Pruning (L1) Accuracy Interpretation

At 10% pruning, all ResNet models maintained strong accuracy (64–76%), especially ResNet101 (75.9%) and ResNet50 (74.63%), showing that light pruning has minimal negative effect on performance.

At 20%, accuracy begins to drop, with smaller architectures (e.g: ResNet18) suffering more than deeper ones indicating deeper models are more resilient to pruning.

Beyond 30% pruning, accuracy drastically drops across all ResNet models, showing that excessive pruning removes important filter weights and harms generalization.

At 40%, accuracies fall to unusable levels (<5%), confirming over-pruning and we have to fine-tune the models on a smaller range of epochs to regain the lost accuracy once again.

For VGG models:

At 20% and higher, all VGG models drop steeply, with VGG16 and VGG19 practically failing (<6%).

The shallow VGGs (11, 13) show some resilience up to 20% of pruning, but none of the VGGs handle high pruning well.

The detailed results across all models are mentioned below:

Table: ResNet L1 Pruned Accuracies

Pruning Level	ResNet18	ResNet34	ResNet50	ResNet101
10%	64.49%	72.65%	74.63%	75.90%
20%	39.38%	56.44%	51.65%	54.40%
30%	9.63%	13.76%	15.48%	13.00%
40%	2.47%	3.71%	3.07%	2.90%

Table: VGG L1 Pruned Accuracies

Pruning Level	VGG11	VGG13	VGG16	VGG19
10%	53.34%	57.71%	49.64%	42.96%
20%	22.74%	17.85%	5.80%	1.51%
30%	2.06%	1.61%	1.54%	0.99%
40%	1.01%	1.00%	1.00%	1.00%

4.4 Pruned Results/Performance

L2 pruning on ResNet models shows good accuracy at 10% pruning, especially for deeper models like ResNet101 (75.97%). However, accuracy drops sharply at higher pruning levels, especially beyond 20%. Deeper models handle pruning better than smaller ones.

VGG models on the other hand, under L2 pruning show moderate performance at 10% pruning, with VGG13 achieving the highest at 54.21%. However, all models experience a steep accuracy decline beyond 20%, dropping below 2% by 30–40% pruning. This suggests VGG architectures are highly sensitive to aggressive L2 pruning.

The detailed results across all models are mentioned below:

Table: ResNet L2 Pruned Accuracies

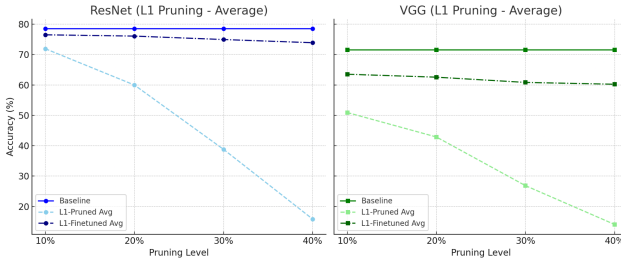
Pruning Level	ResNet18	ResNet34	ResNet50	ResNet101
10%	65.31%	72.65%	74.95%	75.97%
20%	36.95%	56.39%	50.98%	55.87%
30%	12.00%	12.10%	16.87%	13.01%
40%	2.81%	3.95%	2.67%	2.15%

Table: VGG L2 Pruned Accuracies

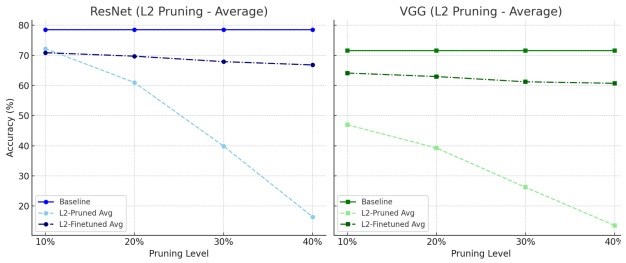
Pruning Level	VGG11	VGG13	VGG16	VGG19
10%	52.91%	54.21%	43.34%	37.40%
20%	10.17%	12.81%	2.69%	1.96%
30%	2.06%	1.32%	1.06%	1.00%
40%	1.09%	1.02%	1.00%	0.99%

The visual line graph below shows the above mentioned results as an average overall performances across all sub-categories of models across various pruning thresholds are discussed earlier.

L1 result line graph is mentioned below:



L2 result line graph is mentioned below:



Therefore, from these graphs we see that the baseline curve is at the topmost of the graph indicating the highest baseline accuracy achieved after initial training. The line below the baseline represents the fine-tuning accuracy after light re-training, and lastly the line below the fine-tuned curve represents the pruned accuracy across thresholds. This line graph visually indicated the process of the baseline to pruned to fine-tuned accuracies across L1 and L2 based pruning methods across multiple pruning thresholds.

5. Conclusion and Future Work

This project systematically evaluates the impact of structured pruning using both L1 and L2 norms across various CNN architectures, including ResNet (18, 34, 50, 101) and VGG (11, 13, 16, 19), on the CIFAR-100 dataset.

The findings show that L1 pruning generally preserved higher accuracy after fine-tuning compared to L2 pruning, especially at lower pruning levels. ResNet models demonstrated greater robustness to pruning than VGG models, maintaining competitive performance even at 30–40% pruning. Fine-tuning significantly recovered performance in both strategies, with optimal results observed at 10–20% pruning. These insights confirm that selective pruning when followed by appropriate retraining, can compress models while retaining meaningful accuracy, thereby reducing computational overhead without heavily compromising performance.

Based on the experiments conducted and based on the best of my abilities, it would be recommended to increasing training baseline epochs (>250), retaining Adam optimizer, and to experiment with slightly smaller learning rates (<0.01), combined with (>30) epochs for fine-tuning.

6. References

Dataset: **Krizhevsky, A.** (2009). *Learning multiple layers of features from tiny images* (Technical Report). University of Toronto. https://www.cs.toronto.edu/~kriz/cifar.html

Code: **Wang, B., & Kindratenko, V.** (2024). *RL-Pruner: Structured pruning using reinforcement learning for CNN compression and acceleration* [Preprint]. arXiv. https://arxiv.org/abs/2411.06463

Other references:

He, Y., Liu, P., Wang, Z., Hu, Z., & Yang, Y. (2019). *Filter pruning via geometric median for deep convolutional neural networks acceleration*. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 4340–4349. https://doi.org/10.1109/CVPR.2019.00448

Li, H., Kadav, A., Durdanovic, I., Samet, H., & Graf, H. P. (2017). *Pruning filters for efficient convnets*. International Conference on Learning Representations (ICLR). https://arxiv.org/abs/1608.08710

Han, S., Pool, J., Tran, J., & Dally, W. (2015). *Learning both weights and connections for efficient neural network*. Advances in Neural Information Processing Systems, 28. https://papers.nips.cc/paper_files/paper/2015/file/ae0eb3eed39d2bcef4622b2499a05fe6-Paper.pdf

Youtube: Han MIT HAN LAB (structured pruning lecture videos).