10605 Mini Project Report

HsiangYu Lee Leyla Li Shizhuo Li
{hsiangyl, leylal, shizhuol}@andrew.cmu.edu
Carnegie Mellon University

1 Methodology

In this project, we implemented and compared three neural network pruning methods: Wanda (Weight and Activation Pruning), L1-norm filter pruning, and Network Slimming.

1.1 L1-norm Based Filter Pruning

The L-1 norm based filter pruning (Li et al., 2017) is to compute the L-1 score for each filter and setting the lowest L-1 scored filters to weight zero. The L-1 norm can be computed by the following function:

$$\text{L1 norm}_i = \sum_{c,h,w} |W_{i,c,h,w}|$$

.

In our L-1 norm base filter pruning, we adapt the method step by step by a rate - that is, prune a small proportion for each step and repeat for many steps (auto skipping pruned filters). In each step, we re-compute the L-1 score for each filter and prune the smallest scored filters. In our implementation, for each step, we prune 0.005 proportion of the filters, afterwhich we can choose an optional finetuning. This prune will be repeated for many steps. We experimented several steps ranging from 8 to 26.

1.2 WANDA

We focused our study on a single unstructured pruning method called **Wanda** (Weight and Activation-aware pruning), which evaluates weight importance by combining weight magnitude with input activation strength.

- **Wanda Pruning (by Weight and Activation Importance)**: This method ranks weights based on the element-wise product of their absolute value and the ℓ_2 norm of the corresponding input activation channels, i.e., $S_{ij} = |W_{ij}| \cdot \|\mathbf{X}_j\|_2$. This strategy allows us to prune weights that are both small in value and tied to weak activations, thus avoiding the removal of influential weights.
- **Layer-wise Pruning Configurations**: Although the pruning technique remained the same, we evaluated several layer selection strategies. Some configurations pruned only dense layers, while others included early and late convolutional layers. Sparsity ratios ranged from 10% to 60% across different experiments, enabling a thorough exploration of trade-offs between accuracy and sparsity.
- **Incremental and Deterministic Pruning Pipeline**: We implemented a reproducible pruning framework where each layer is pruned one at a time from a clean checkpoint. For each pruning step, we use the same fixed input batch to extract activation statistics. This avoids inter-layer contamination and ensures that pruning outcomes are deterministic and comparable across models.

1.3 Network Slimming

Network Slimming (Liu et al., 2017) leverages the γ scaling factors in BatchNorm layers as importance indicators. The method proceeds as follows:

1. Train a CNN with BatchNorm layers.
2. Collect γ values and compute a pruning threshold based on a pruning ratio (e.g., 40% percentile).
3. Prune channels (filters) whose corresponding γ values fall below the threshold.
4. Fine-tune the pruned model to recover performance.
5. Optionally, apply weight thresholding to zero out small weights post fine-tuning.

This method is elegant and effective, especially for structured channel pruning. Since it builds on the already existing BatchNorm layers, it introduces minimal overhead.

1.3.1 Computing Pruning Threshold

After training, we determine a pruning threshold value based on the distribution of γ values. See the Figure 1 Using a specified pruning ratio (e.g., 30%), we compute the threshold using a percentile function. Table below shows the impact of various pruning ratios on model sparsity and validation accuracy.

After computing the threshold percentage, we then proceed with filter pruning whose corresponding γ values is below the threshold by force iteration.

After pruning, we fine-tune the model using standard training procedures to recover the performance lost during pruning.

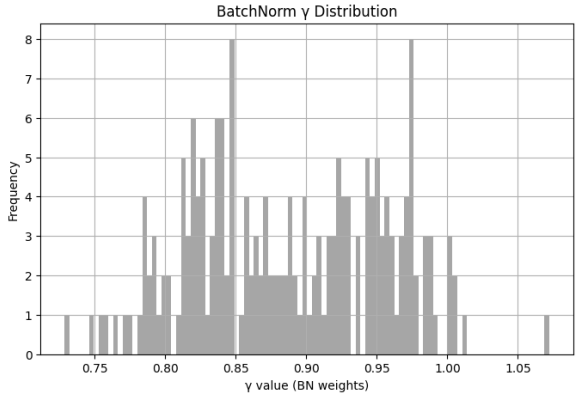

Figure 1: Impact of pruning ratio on model accuracy

2 Comparison

Below is the comparison between the three ways of pruning:

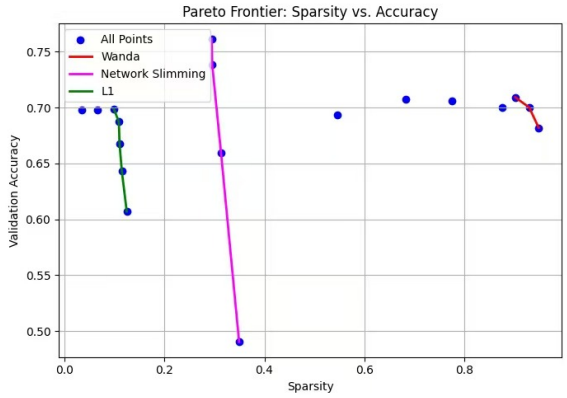

Figure 2: Pareto Frontier of Sparsity vs. Accuracy for L1, Wanda, and Network Slimming

We compare three pruning methods: **Wanda**, **L1-norm pruning**, and **Network Slimming**, focusing on their trade-offs between model sparsity and validation accuracy. As shown in Figure 2, each method yields a different balance of sparsity and performance.

Overall Performance

Among the three methods, **Wanda demonstrates the best overall performance**. For a comparable level of validation accuracy (around 70%), Wanda achieves the highest sparsity. This indicates that Wanda is most effective at compressing the model while maintaining competitive performance. The curve formed by Wanda's results dominates the other two methods on the Pareto frontier.

L1-norm Pruning

The L1-norm method ranks all weights by their absolute magnitude and removes a specified percentage of the smallest ones. It is conceptually straightforward and requires minimal structural modification. However, at higher sparsity levels, its accuracy tends to drop more steeply than Wanda, indicating less effective pruning in practice.

Network Slimming

Network Slimming uses the scaling coefficients in Batch Normalization layers as importance indicators. During training, L1 regularization is added to these coefficients, and channels with small scaling factors are pruned afterward. Although this method maintains reasonable accuracy under moderate sparsity, it requires re-training the model with regularization and additional thresholding steps, increasing the implementation workload.

Ease of Implementation

Assuming all methods are implemented from scratch:

- **L1-norm pruning** is the simplest. It only requires sorting weights by magnitude and applying masks, making it the easiest to implement with minimal engineering effort.
- **Wanda** is moderately complex. It involves computing weight importance scores based on gradient or second-order information, but can be done post-training without retraining the model.
- **Network Slimming** is the most complex. It modifies the training process by adding regularization, and requires post-training processing and tuning of sparsity thresholds.

3 Reflection

We initially expected **Network Slimming** to perform best due to its training-time integration. However, **Wanda**, a simpler post-training method, achieved up to **90% sparsity** with comparable accuracy, showing that direct pruning can be highly effective.

Overall, we observed a trade-off between sparsity and validation accuracy that varied across methods. Many pruned models preserved the accuracy of the original, supporting the over-parameterization theory behind the *Lottery Ticket Hypothesis*. We also found that combining techniques — such as L1 pruning with Network Slimming and applying gradual, multi-step pruning with fine-tuning — can yield better results than one-shot approaches.

References

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. 2017. Pruning filters for efficient convnets.

Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. 2017. Learning efficient convolutional networks through network slimming.