Pruning Filters for Efficient ConvNets

Hao Li¹, Asim Kadav², Igor Durdanovic², Hanan Samet¹, Hans Peter Graf²
University of Maryland¹, NEC Labs America²

Motivation
- Low computation cost of CNNs is a crucial factor for mobile applications and cloud services.
- Convolutional layers dominate computation and storage costs in state-of-art CNNs [1].
- Pruning small weights [2] mostly reduces the storage cost of parameters from FC layers and require sparse convolutions.

Contributions
- Reduce the inference computation cost of CNNs by pruning filters avoiding the need for sparse convolution libraries.
- Simple criterion for filter selection, without examining each feature map’s importance [3,4].
- Prune multiple filters together and retrain once, avoiding iterative pruning and retraining.

Determine Filters’ Importance
- For each conv layer, we measure each filter’s relative importance by its absolute weight sum \( \sum |f_{ij}| \), i.e., its \( \| f \|_1 \) norm. This value also represents the average magnitude of its weights.
- Filters with small weights tend to produce feature maps with weak activations.
- Pruning the smallest filters works better in comparison with pruning the same number of random or largest filters.

Determine Single Layer’s Sensitivity to Pruning
- Pruning the smallest filters of single layer
- Regain accuracy by retraining

Pruning ratios
- Layers with the same input sizes often have similar sensitivity to pruning. We use the same pruning ratio for these layers to avoid tuning layer-specific meta-parameters.
- For layers that are sensitive to pruning, we use a small pruning rate or completely skip pruning them.

Prune Filters across Multiple Layers
- Independent pruning determines filters to be pruned at one layer independent of other layers.
- Greedy pruning does not count kernels connected with the previously pruned feature maps during filter selection.

Pruning residual blocks with projection shortcut
- The first layer of the residual block can be pruned without restrictions.
- The filters to be pruned in the second conv layer of the residual blocks is determined by the pruning result of the shortcut projection.

Retrain Pruned Networks
- Instead of iterative pruning and retraining, we adopt a one-shot pruning and retraining strategy (~1/4 of the original training time).

Experiments

<table>
<thead>
<tr>
<th>Model</th>
<th>Error</th>
<th>FLOP</th>
<th>Pruned</th>
<th>Parameters</th>
<th>Pruned</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG-16</td>
<td>6.75</td>
<td>3.13</td>
<td>1.6 x 10⁵</td>
<td>5.4 x 10⁴</td>
<td>64%</td>
</tr>
<tr>
<td>VGG-16-pruned-A</td>
<td>6.60</td>
<td>2.06 x 10⁴</td>
<td>34.2%</td>
<td>5.4 x 10⁴</td>
<td>64%</td>
</tr>
<tr>
<td>VGG-16-pruned-A scratch train</td>
<td>6.88</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ResNet-56</td>
<td>6.76</td>
<td>1.25 x 10⁵</td>
<td>8.5 x 10⁴</td>
<td>9.4%</td>
<td></td>
</tr>
<tr>
<td>ResNet-56-pruned-A</td>
<td>6.90</td>
<td>1.12 x 10⁴</td>
<td>10.4%</td>
<td>7.7 x 10⁴</td>
<td>9.4%</td>
</tr>
<tr>
<td>ResNet-56-pruned-B</td>
<td>6.94</td>
<td>9.09 x 10⁴</td>
<td>27.6%</td>
<td>7.3 x 10⁴</td>
<td>13.7%</td>
</tr>
<tr>
<td>ResNet-56-pruned-B scratch train</td>
<td>8.69</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ResNet-110</td>
<td>6.42</td>
<td>2.53 x 10⁵</td>
<td>1.72 x 10⁴</td>
<td>2.3%</td>
<td></td>
</tr>
<tr>
<td>ResNet-110-pruned-A</td>
<td>6.65</td>
<td>2.13 x 10⁴</td>
<td>15.9%</td>
<td>1.68 x 10⁴</td>
<td>2.3%</td>
</tr>
<tr>
<td>ResNet-110-pruned-B</td>
<td>6.70</td>
<td>1.55 x 10⁴</td>
<td>38.6%</td>
<td>1.16 x 10⁴</td>
<td>32.4%</td>
</tr>
<tr>
<td>ResNet-110-pruned-B scratch train</td>
<td>7.06</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ResNet-34</td>
<td>26.77</td>
<td>3.64 x 10⁵</td>
<td>2.16 x 10⁴</td>
<td>7.2%</td>
<td></td>
</tr>
<tr>
<td>ResNet-34-pruned-A</td>
<td>27.44</td>
<td>3.08 x 10⁴</td>
<td>15.5%</td>
<td>1.99 x 10⁴</td>
<td>7.6%</td>
</tr>
<tr>
<td>ResNet-34-pruned-B</td>
<td>27.83</td>
<td>2.76 x 10⁴</td>
<td>24.2%</td>
<td>1.93 x 10⁴</td>
<td>10.8%</td>
</tr>
<tr>
<td>ResNet-34-pruned-C</td>
<td>27.52</td>
<td>3.37 x 10⁴</td>
<td>7.5%</td>
<td>2.01 x 10⁴</td>
<td>7.2%</td>
</tr>
</tbody>
</table>

Overall results
- ~30% reduction in FLOPs for VGG-16 (on CIFAR-10) and ResNets without significant loss in accuracy.
- Training a pruned model from scratch performs worse than retraining a pruned model.
- Pruning the first layer of the residual block is more effective.

Sensitivity analysis
- For ResNets, layers that are sensitive to pruning are close to the residual blocks where the number of feature maps changes.

References

5th International Conference on Learning Representations (ICLR) 2017, Toulon, France