NETWORK COMPRESSION
Hung-yi Lee 李宏毅
Resource-limited Devices

Limited memory space, limited computing power, etc.
Outline

• Network Pruning
• Knowledge Distillation
• Parameter Quantization
• Architecture Design
• Dynamic Computation

We will not talk about hard-ware solution today.
Network Pruning
Network can be pruned

• Networks are typically over-parameterized (there is significant redundant weights or neurons)
• Prune them!
Network Pruning

• Importance of a weight:
  L1, L2 ......

• Importance of a neuron:
  the number of times it wasn’t zero on a given data set ......

• After pruning, the accuracy will drop (hopefully not too much)

• Fine-tuning on training data for recover

• Don’t prune too much at once, or the network won’t recover.

Pre-trained Network

Evaluate the Importance

Remove

Fine-tune

Are you happy?

yes

Smaller Network

no

large

smaller
Why Pruning?

• How about simply train a smaller network?
• It is widely known that smaller network is more difficult to learn successfully.
  • Larger network is easier to optimize? [YouTube Video](https://www.youtube.com/watch?v=_VuWvQUMQVk)
• Lottery Ticket Hypothesis [Paper](https://arxiv.org/abs/1803.03635)
Why Pruning?
Lottery Ticket Hypothesis

Random init weights
Trained weight
Another random init weights
Why Pruning?

• Rethinking the Value of Network Pruning

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Model</th>
<th>Unpruned</th>
<th>Pruned Model</th>
<th>Fine-tuned</th>
<th>Scratch-E</th>
<th>Scratch-B</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIFAR-10</td>
<td>VGG-16</td>
<td>93.63 (±0.16)</td>
<td>VGG-16-A</td>
<td>93.41 (±0.12)</td>
<td>93.62 (±0.11)</td>
<td><strong>93.78</strong> (±0.15)</td>
</tr>
<tr>
<td></td>
<td>ResNet-56</td>
<td>93.14 (±0.12)</td>
<td>ResNet-56-A</td>
<td>92.97 (±0.17)</td>
<td>92.96 (±0.26)</td>
<td><strong>93.09</strong> (±0.14)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ResNet-56-B</td>
<td>92.67 (±0.14)</td>
<td>92.54 (±0.19)</td>
<td><strong>93.05</strong> (±0.18)</td>
</tr>
<tr>
<td></td>
<td>ResNet-110</td>
<td>93.14 (±0.24)</td>
<td>ResNet-110-A</td>
<td>93.14 (±0.16)</td>
<td><strong>93.25</strong> (±0.29)</td>
<td>93.22 (±0.22)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ResNet-110-B</td>
<td>92.69 (±0.09)</td>
<td>92.89 (±0.43)</td>
<td><strong>93.60</strong> (±0.25)</td>
</tr>
<tr>
<td>ImageNet</td>
<td>ResNet-34</td>
<td>73.31</td>
<td>ResNet-34-A</td>
<td>72.56</td>
<td>72.77</td>
<td><strong>73.03</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ResNet-34-B</td>
<td>72.29</td>
<td>72.55</td>
<td><strong>72.91</strong></td>
</tr>
</tbody>
</table>

• Real random initialization, not original random initialization in “Lottery Ticket Hypothesis”
• Pruning algorithms could be seen as performing network architecture search
Network Pruning - Practical Issue

• Weight pruning

The network architecture becomes irregular.

Prune some weights

Hard to implement, hard to speedup ......
Network Pruning - Practical Issue

- Weight pruning

Network Pruning - Practical Issue

• Neuron pruning

The network architecture is regular.

Prune some neurons

Easy to implement, easy to speedup ......
Knowledge Distillation
Knowledge Distillation

Knowledge Distillation
Do Deep Nets Really Need to be Deep?

Learning target
“1”: 0.7, “7”: 0.2, “9”: 0.1

Cross-entropy minimization

Teacher Net (Large)

Providing the information that “1” is similar to “7”

Student Net (Small)
Knowledge Distillation

Knowledge Distillation
Do Deep Nets Really Need to be Deep?

Learning target
“1”: 0.7, “7”: 0.2, “9”: 0.1

Cross-entropy minimization

Average of a set of models
Ensemble

Student Net (Small)
Knowledge Distillation

• Temperature

\[
y_i = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \quad \rightarrow \quad y_i = \frac{\exp(x_i/T)}{\sum_j \exp(x_j/T)}
\]

\[
x_1 = 100 \quad y_1 = 1 \quad x_1/T = 1 \quad y_1 = 0.56
\]
\[
x_2 = 10 \quad y_2 \approx 0 \quad x_2/T = 0.1 \quad y_2 = 0.23
\]
\[
x_3 = 1 \quad y_3 \approx 0 \quad x_3/T = 0.01 \quad y_3 = 0.21
\]
Parameter Quantization
Parameter Quantization

- 1. Using less bits to represent a value
- 2. Weight clustering

<table>
<thead>
<tr>
<th>weights in a network</th>
<th>0.5</th>
<th>1.3</th>
<th>4.3</th>
<th>-0.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>-0.2</td>
<td>-1.2</td>
<td>0.3</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>3.0</td>
<td>-0.4</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>-0.5</td>
<td>-0.1</td>
<td>-3.4</td>
<td>-5.0</td>
<td></td>
</tr>
</tbody>
</table>

Clustering
Parameter Quantization

• 1. Using less bits to represent a value
• 2. Weight clustering

weights in a network

<table>
<thead>
<tr>
<th></th>
<th>0.5</th>
<th>1.3</th>
<th>4.3</th>
<th>-0.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>-0.2</td>
<td>-1.2</td>
<td>0.3</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>3.0</td>
<td>-0.4</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>-0.5</td>
<td>-0.1</td>
<td>-3.4</td>
<td>-5.0</td>
<td></td>
</tr>
</tbody>
</table>

Clustering

Only needs 2 bits

• 3. Represent frequent clusters by less bits, represent rare clusters by more bits
  • e.g. Huffman encoding

Table

<table>
<thead>
<tr>
<th></th>
<th>-0.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>2.9</td>
<td></td>
</tr>
<tr>
<td>-4.2</td>
<td></td>
</tr>
</tbody>
</table>
Binary Weights

Your weights are always +1 or -1

• Binary Connect

Network with binary weights

Network with real value weights

Negative gradient (compute on binary weights)

Update direction (compute on real weights)
# Binary Connect

<table>
<thead>
<tr>
<th>Method</th>
<th>MNIST</th>
<th>CIFAR-10</th>
<th>SVHN</th>
</tr>
</thead>
<tbody>
<tr>
<td>No regularizer</td>
<td>1.30 ± 0.04%</td>
<td>10.64%</td>
<td>2.44%</td>
</tr>
<tr>
<td>BinaryConnect (det.)</td>
<td>1.29 ± 0.08%</td>
<td>9.90%</td>
<td>2.30%</td>
</tr>
<tr>
<td>BinaryConnect (stoch.)</td>
<td>1.18 ± 0.04%</td>
<td>8.27%</td>
<td>2.15%</td>
</tr>
<tr>
<td>50% Dropout</td>
<td>1.01 ± 0.04%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

https://arxiv.org/abs/1511.00363
Architecture Design
Low rank approximation

\[ W \approx U V \]

Less parameters

K < M, N
Review: Standard CNN

Input feature map

2 channels

3 × 3 × 2 × 4 = 72 parameters
**Depthwise Separable Convolution**

1. Depthwise Convolution

- Filter number = Input channel number
- Each filter only considers one channel.
- The filters are $k \times k$ matrices
- There is no interaction between channels.
Depthwise Separable Convolution

1. Depthwise Convolution

2. Pointwise Convolution

\[ 3 \times 3 \times 2 = 18 \]

\[ 1 \times 1 \text{ filter} \]

\[ 2 \times 4 = 8 \]
$I$: number of input channels

$O$: number of output channels

$k \times k$: kernel size

$$\frac{k \times k \times I + I \times O}{k \times k \times I \times O}$$

$$= \frac{1}{O} + \frac{1}{k \times k}$$
To learn more ......

• SqueezeNet
  • https://arxiv.org/abs/1602.07360

• MobileNet
  • https://arxiv.org/abs/1704.04861

• ShuffleNet
  • https://arxiv.org/abs/1707.01083

• Xception
  • https://arxiv.org/abs/1610.02357
Dynamic Computation
Dynamic Computation

• Can network adjust the computation power it need?

資源充足，那麼就做到最好

減少運算量，先求有再求好（但也不要太差）
Possible Solutions

• 1. Train multiple classifiers
• 2. Classifiers at the intermedia layer

https://arxiv.org/abs/1703.09844
Multi-Scale Dense Networks

https://arxiv.org/abs/1703.09844
Concluding Remarks

• Network Pruning
• Knowledge Distillation
• Parameter Quantization
• Architecture Design
• Dynamic Computation