Exploring Bit-Slice Sparsity in Deep Neural Networks for Efficient ReRAM-Based Deployment

Jingyang Zhang¹, Huanrui Yang¹, Fan Chen¹, Yitu Wang², Hai Li¹
¹Duke University, ²Fudan University

Motivation

- Deep Neural Network (DNN) models are powerful, but are costly to deploy.
- Resistive random-access memory (ReRAM)-based DNN accelerators (Shafiee et al. 2016, Song et al. 2017)
- In-situ matrix-vector multiplications
- Two-order magnitude advantage in energy, performance and chip footprint

Challenges of ReRAM-based accelerators

- Limited cell bit density: operands (i.e. weights) are bit-sliced across multiple ReRAM bitlines
- High bit-resolution ADC accounts for high power (>60%) and area (>30%) overhead
- ADC resolution dictated by accumulated currents on bitlines
- Need higher sparsity in each bit-slice to reduce the accumulated current on bitlines, therefore reducing ADC overhead

Methods

- Dynamic fixed-point quantization
  - For each layer, preserve dynamic range
    \[ S(W_i) = \log_2(\max_w |w_i|) \]
  - Uniform n-bit quantization after scaling
    \[ Q_{\text{step}} = 2^{-S(W_i)} \]
  - \(B(w_i)\) will be stored on ReRAM crossbar
  - Dynamic range recovery
    \[ Q(w_i) = B(w_i) \cdot Q_{\text{step}} \]
  - Recovery can be done efficiently with shifting
  - \(Q(w_i)\) will be used for computation

- Bit-slice L1
  - Binary representation of the quantized weight
  - Slice into 2-bit slices
  - L1 regularization across all bit-slices of all elements within a weight matrix/tensor
  \[ B(w_i) = \sum_{j=0}^{2^{n-1}} B_1^{(j)} \]
  \[ B_1^{(j)} = \begin{cases} \frac{1}{2^{n/2}} & \text{if } j \leq 2^{n/2} - 1 \\ \frac{1}{2^n} & \text{else} \end{cases} \]

Overall training routine

- Forward and backward pass with quantized weight, gradient update on full-precision weight
- Add bit-slice L1 to the objective

Results

- Significant improvement on bit-slice sparsity

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
<th>(B_1^{(j)})</th>
<th>(B_2^{(j)})</th>
<th>(B_1^{(j)})</th>
<th>(B_2^{(j)})</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pruned</td>
<td>97.99%</td>
<td>1.08%</td>
<td>5.87%</td>
<td>8.42%</td>
<td>17.42%</td>
<td>8.20%</td>
</tr>
<tr>
<td>(l_1)</td>
<td>97.99%</td>
<td>1.19%</td>
<td>5.31%</td>
<td>7.01%</td>
<td>13.56%</td>
<td>6.19%</td>
</tr>
<tr>
<td>(B_1)</td>
<td>97.67%</td>
<td>0.84%</td>
<td>4.62%</td>
<td>4.27%</td>
<td>9.58%</td>
<td>4.68%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
<th>(B_1^{(j)})</th>
<th>(B_2^{(j)})</th>
<th>(B_1^{(j)})</th>
<th>(B_2^{(j)})</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pruned</td>
<td>88.93%</td>
<td>0.86%</td>
<td>28.30%</td>
<td>33.39%</td>
<td>24.17%</td>
<td>13.05%</td>
</tr>
<tr>
<td>(l_1)</td>
<td>89.33%</td>
<td>0.30%</td>
<td>9.37%</td>
<td>18.45%</td>
<td>22.19%</td>
<td>12.59%</td>
</tr>
<tr>
<td>(B_1)</td>
<td>89.33%</td>
<td>0.21%</td>
<td>3.57%</td>
<td>7.09%</td>
<td>10.71%</td>
<td>5.40%</td>
</tr>
</tbody>
</table>

Methods (cont’d)

- Bit-slice L1
- Binary representation of the quantized weight
- Slice into 2-bit slices
- L1 regularization across all bit-slices of all elements within a weight matrix/tensor

Conclusion

- We propose bit-slice L1 regularizer, the first algorithm to induce bit-slice sparsity during the training of dynamic fixed-point DNNs
- Up to 2x or more sparsity improvement on bit slices comparing to traditional L1 regularizer
- When deployed on ReRAM-based accelerator, the achieved bit-slice sparsity allows the ADC resolution to be reduced to 1-bit of the most significant bit-slice and down to 3-bit for the others bits, which significantly reduces power and area overhead.

Acknowledgement

This work was supported in part by NSF CNS-1822085 and NSF CSR-1717885.

References


Codes available at: https://github.com/zjysteven/bit-slice_sparsity