A Configurable Multi-Precision CNN Computing Framework Based on Single Bit RRAM

Zhenhua Zhu\textsuperscript{1,2}, Hanbo Sun\textsuperscript{1,2}, Yujun Lin\textsuperscript{3}, Guohao Dai\textsuperscript{1,2}, Lixue Xia\textsuperscript{1,4}
Song Han\textsuperscript{3}, Yu Wang\textsuperscript{1,2}, Huazhong Yang\textsuperscript{1,2}

\textsuperscript{1}Department of Electronic Engineering, Tsinghua University, Beijing, China
\textsuperscript{2}Beijing National Research Center for Information Science and Technology (BNRist), Beijing, China
\textsuperscript{3}HAN Lab, Electrical Engineering and Computer Science Department, Massachusetts Institute of Technology (MIT), USA
\textsuperscript{4}Alibaba Group

Email: yu-wang@tsinghua.edu.cn
• Introduction and motivation
• Proposed Framework
  – **Software Level**: RRAM Computing Overhead Aware Quantization and Compression Algorithm
  – **Hardware Level**: A Configurable Multi-precision CNN Computing Architecture based on Single-bit RRAM
    – Optimal Hardware Configuration Search
• Simulations Results
• Conclusion
Convoluotional Neural Network

- CNNs are powerful.

Image Classification [A. Krizhevsky, et al. NIPS’12]

Object Detection [DeePhi Tech]

Automatic Driving [Bounini, et al. VPPC’15]
CNNs need more powerful computing platforms

• The computation of CNN **consumes high energy**
  – The inference of VGG-16 for one image needs **4.3J** energy consumption on GPU [Qiu J, et al. FPGA’16]

• **Large data movements** between PE and MEM cause high energy consumption
  – The data transfer in GPU consumes **2 orders of magnitude** more energy than a floating-point operation [Han S, et al. ISCA’16]

<table>
<thead>
<tr>
<th>Operation</th>
<th>Energy [pJ]</th>
<th>Relative Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>32 bit int ADD</td>
<td>0.1</td>
<td>1</td>
</tr>
<tr>
<td>32 bit float ADD</td>
<td>0.9</td>
<td>9</td>
</tr>
<tr>
<td>32 bit Register File</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>32 bit int MULT</td>
<td>3.1</td>
<td>31</td>
</tr>
<tr>
<td>32 bit float MULT</td>
<td>3.7</td>
<td>37</td>
</tr>
<tr>
<td>32 bit SRAM Cache</td>
<td>5</td>
<td>50</td>
</tr>
<tr>
<td>32 bit DRAM Memory</td>
<td>640</td>
<td>6400</td>
</tr>
</tbody>
</table>

Von Neumann Architecture

High Memory Access Energy Consumption
RRAM & PIM provide efficient solutions

- **RRAM and Processing-In-Memory** provide alternative solutions to realize better implementation of CNN.
## Existing RRAM-based accelerator

<table>
<thead>
<tr>
<th></th>
<th>RRAM Precision</th>
<th>Function</th>
<th>Performance</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISAAC</td>
<td>2 bits</td>
<td>CNN Inference</td>
<td><strong>14.8x</strong> Throughput and <strong>4.4x</strong> Energy Efficiency (Compared with DaDianNao)</td>
<td>Shafiee A, et al. ISCA’16</td>
</tr>
<tr>
<td>PRIME</td>
<td>4 bits</td>
<td>CNN Inference</td>
<td><strong>2360x</strong> Speedup and <strong>895x</strong> Energy Efficiency (Compared with DianNao)</td>
<td>Chi P, et al. ISCA’16</td>
</tr>
<tr>
<td>PipeLayer</td>
<td>5~6 bits</td>
<td>CNN Inference and Training</td>
<td><strong>42.45x</strong> Speedup and <strong>7.17x</strong> Energy Efficiency (Compared with GPU)</td>
<td>Song L, et al. HPCA’17</td>
</tr>
<tr>
<td>TIME</td>
<td>4 bits</td>
<td>CNN/DRL Inference and Training</td>
<td>CNN: <strong>1.3x</strong> Speedup and <strong>19.6x</strong> Energy Efficiency (Compared with DaDianNao) DRL: <strong>126x</strong> Energy Efficiency (Compared with GPU)</td>
<td>Cheng M, et al. TDAC’18</td>
</tr>
<tr>
<td>NTHU Chip-1</td>
<td>1 bit</td>
<td>Binary DNN/CNN</td>
<td>CNN: 14.8ns FCM: 15.6ns</td>
<td>Chen W, et al. ISSCC’18</td>
</tr>
<tr>
<td>—</td>
<td>1 bit</td>
<td>BCNN</td>
<td><strong>58.2%</strong> area energy consumption</td>
<td>Tang T, et al. ASPDAC’17</td>
</tr>
</tbody>
</table>
Design Challenges in RRAM Computing System

- Higher storage/computing density
- Hard to implement and tape out
- Affected by non-ideal factors severely
- Larger quantization errors at interfaces

Multi-bit RRAM

RRAM Accelerators

- [Cheng M, et al. TDAC’18]
- [Song L, et al. HPCA’17]
- [Shafiee A, et al. ISCA’16]
- [Chi P, et al. ISCA’16]
- [Cheng M, et al. TDAC’18]
- [Chen W, et al. ISSCC’18]
- [Xue C, et al. ISSCC’19]

Neural Network Algorithm

Cat
Panda
Elephant
Dog
Design Challenges in RRAM Computing System

Neural Network Algorithm

RRAM Accelerators

- More reliable and accurate
- Limited precision causes accuracy loss in BCNN
- Huge parameter number needs more devices in CNN

Single-bit RRAM

- [Shafiee A, et al. ISCA’16]
- [Chi P, et al. ISCA’16]
- [Cheng M, et al. TDAC’18]
- [Song L, et al. HPCA’17]
- [Chen W, et al. ISSCC’18]
- [Xue C, et al. ISSCC’19]
- [Liu R, et al. TNS’15]
Design Challenges in RRAM Computing System

Our Target

**Design an efficient CNN computing framework based on single bit RRAM**

- Higher storage/computing density
- Hard to implement and tape out
- Non-ideal factors severely affect performance
- Larger quantization errors at interfaces
- Multibit RRAM
  - [Zhao L, et al. Nanoscale’14]
  - [Liu R, et al. TNS’15]
- More reliable and accurate
- Limited precision causes accuracy loss in BCNN
- Huge parameter number needs more devices in CNN

**Neural Network Algorithm**
- Panda
- Elephant
- Dog

**RRAM Accelerators**
- [Chen W, et al. ISSCC’18]
- [Xue C, et al. ISSCC’19]

**Single-bit RRAM**
- [Liu R, et al. TNS’15]
Proposed Framework Overview

**RRAM-Aware Quantization**
- Analysis of the RRAM computing deviation
- Optimization model of RRAM-aware quantization
- Multi-precision CNN quantization scheme

**Configurable Multi-Precision Architecture**
- Multi-bit data splitting implementation
- Configurable architecture design
- Algorithm mapping strategy
I. RRAM-Aware Quantization

• Source of RRAM computing deviation:
  – Device-level and circuit-level non-ideal factors (e.g., resistance variations)
  – Quantization error of the analog/digital interfaces

\[ Q \text{ v.s. } Q_{\text{ideal}} = m + \log_2(N) + 1 \]

Input data: \( m - \text{bit}, \) crossbar size: \( N, \) precision of SA&ADC: \( Q - \text{bit} \)

• Directly mapping well-trained model on RRAM suffers from accuracy loss

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>GPU Quantization</th>
<th>GPU Quant. Model on RRAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>92.02%</td>
<td><strong>88.80%</strong></td>
<td><strong>83.80%</strong></td>
</tr>
</tbody>
</table>
I. RRAM-Aware Quantization

• Motivation of multi-bit quantization
  – The precision requirements of each layer are different
  – Layer-wise quantization can achieve fewer storage devices without accuracy loss

The accuracy increment of VGG as the precision of (a) activations or (b) weights increases
I. RRAM-Aware Quantization

- **Optimization model** of RRAM-aware quantization:
  - Storage: the number of crossbars to store weights
    - $L$: layer number, $W_i$: weight precision, $C_{in}$&$C_{out}$: Input&Output channel, $k$: kernel size

\[
Num_c = \sum_{i=1}^{L} 2 \times W_i \times \left[ \frac{K_i^2 C_{in_i}}{N} \right] \times \left[ \frac{C_{out_i}}{N} \right]
\]
I. RRAM-Aware Quantization

- **Optimization model** of RRAM-aware quantization:
  - Latency: mainly consider the RRAM crossbar computation time
    - $M$: activation precision, $m$: DAC precision, $S_t$: sliding times
    - $\text{Latency} = \sum_{i=1}^{L} \frac{M_i}{m} \times S_{t_i}$

- RRAM-Aware Quantization

---

p. 14
I. RRAM-Aware Quantization

- **Optimization model** of RRAM-aware quantization:
  
  - **Storage:** the number of crossbars to store weights
    
    \[ \text{Num}_c = \sum_{i=1}^{L} 2 \times W_i \times \left\lfloor \frac{K_i^2 C_{in_i}}{N} \right\rfloor \times \left\lfloor \frac{C_{out_i}}{N} \right\rfloor \]
    
  - **Latency:** mainly consider the RRAM crossbar computation time
    
    \[ \text{Latency} = \sum_{i=1}^{L} \frac{M_i}{m} \times S_{ti} \]
    
  - **Optimization target:**
    
    \[ \min_{\{W_i\}, \{M_i\}} (\alpha \text{Num}_c(\{W_i\}, \{M_i\}) + \beta \text{Latency}(\{W_i\}, \{M_i\})) \]
    
    \[ \text{s.t. } \text{Loss}(\{W_i\}, \{M_i\}) \leq \text{Loss\_Threshold} \]
I. RRAM-Aware Quantization

- RRAM computing deviation aware quantization and compression scheme
II. Configurable Multi-Precision Architecture

• Data splitting implementation:
  – Use multiple crossbars to store one multi-bit weights
  • Hardware friendly for multi-precision algorithms
II. Configurable Multi-Precision Architecture

- Data splitting implementation:
  - Use multiple crossbars to store one multi-bit weights
  - Hardware friendly for multi-precision algorithms
  - Rewrite the convolution equation:

\[
f_o(x, y, z) = \sum_{i=0}^{K-1} \sum_{j=0}^{K-1} \sum_{k=1}^{C_{in}} f_i(x + i, y + j, k)k_z(i, j, k)
\]

\[
= \sum_{c=0}^{M/m-1} \left[ 2^{mc} \sum_{w=0}^{W-1} 2^w \sum_{i=0}^{K-1} \sum_{j=0}^{K-1} \sum_{k=1}^{C_{in}} f_{ic}(x + i, y + j, k)k_{zw}(i, j, k) \right]
\]
II. Configurable Multi-Precision Architecture

• Configurable architecture design:

Right: The overall architecture design;
Left: Details of the Process Element (PE), the PE Slice, and the Joint Model
II. Configurable Multi-Precision Architecture

• Mapping strategy:
  – Determine # of PE slices and RRAM bank for each layer
    \[ N_{PES} = \left\lceil \frac{K^2 \times C_{in}}{N} \times \frac{W}{8} \right\rceil \quad ; \quad N_{Bank} = \left\lceil \frac{C_{out} \times N}{2^S / N_{PES}} \right\rceil \]
  – Use H-Tree structure to optimize the PE slice allocation:

The schematic diagram of the allocation strategy, the number represents the order of allocations
III. Optimal Hardware Configuration Search

• Tradeoff between **accuracy** and **hardware performance**:  
  – DAC:  
    • More cycles are needed for **loading data** @ lower DACs’ precision  
    • The area and power of DACs grow exponentially with precision  
  – Crossbar size:  
    • Crossbar size influences **the quantization error**  
    • Different sizes cause different area and energy overhead  
  – SA and ADC:  
    • **The quantization precision** affects the computing accuracy  
    • Higher precision means larger area and higher power consumption  
    • Larger area means lower parallelism -> slower
### Simulation results

- **Tradeoff between accuracy and hardware overhead**

#### Table: Simulation results with different ADC precisions

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
<th>Area/mm²</th>
<th>Energy/mJ</th>
<th>Latency/ms</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-bit</td>
<td>91.92%</td>
<td>319.16</td>
<td>0.465</td>
<td>5.76</td>
</tr>
<tr>
<td>2-bit</td>
<td>91.80%</td>
<td>319.21</td>
<td>0.239</td>
<td>2.88</td>
</tr>
</tbody>
</table>

#### Table: Simulation results with different DAC precisions

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
<th>Area/mm²</th>
<th>Energy/mJ</th>
<th>Latency/ms</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-bit</td>
<td>91.92%</td>
<td>319.16</td>
<td>0.465</td>
<td>5.76</td>
</tr>
<tr>
<td>2-bit</td>
<td>91.80%</td>
<td>319.21</td>
<td>0.239</td>
<td>2.88</td>
</tr>
</tbody>
</table>

#### Table: Simulation results with different ADC precisions and crossbar sizes

- LeNet
  - Crossbar size = 128
    - Q=6: 76.00%
    - Q=8: 76.64%
    - Q=10: 76.44%
  - Crossbar size = 256
    - Q=6: 75.96%
    - Q=8: 76.52%
    - Q=10: 76.54%
  - Crossbar size = 512
    - Q=6: 75.36%
    - Q=8: 76.64%
    - Q=10: 76.26%

- VGG-16
  - Crossbar size = 128
    - Q=6: 91.86%
    - Q=8: 92.04%
    - Q=10: 92.36%
  - Crossbar size = 256
    - Q=6: 91.80%
    - Q=8: 92.20%
    - Q=10: 92.32%
  - Crossbar size = 512
    - Q=6: 92.12%
    - Q=8: 91.70%
    - Q=10: 92.20%

- ResNet-18
  - Crossbar size = 128
    - Q=6: 94.62%
    - Q=8: 94.90%
    - Q=10: 95.04%
  - Crossbar size = 256
    - Q=6: 94.78%
    - Q=8: 94.94%
    - Q=10: 95.06%
  - Crossbar size = 512
    - Q=6: 94.42%
    - Q=8: 94.94%
    - Q=10: 94.88%

#### Optimal Hardware Design:
- Crossbar Size: 256x256
- SA&ADC Precision: 6-bit
- DAC Precision: 2-bit
Simulation results

- Tradeoff between accuracy and hardware overhead

VGG16 Accuracy on Cifar10 dataset and Hardware Performance under Different Accuracy Loss Thresholds
• Performance analysis
  – Equivalent energy efficiency of RRAM computing units: 3.44TOps/W (8.6x @ ISAAC and 1.6x @ PRIME)
  – Energy and area breakdown:

The Energy (left) and Area (right) Breakdown
Conclusion

• **Challenges:**
  - For multi-bit RRAM architecture: it is hard to implement and fabricate a chip due to the immature process technology
  - For single-bit RRAM chip: existing work only focuses on BCNN with accuracy loss

• **Solutions:**
  - A configurable multi-precision CNN computing framework based on single bit RRAM
  - Software: an RRAM computing over- head aware network quantization algorithm
  - Hardware: a configurable multi-precision CNN computing architecture based on single bit RRAM

• **Future Work:**
  - Improve and optimize the mapping strategy
  - Design the buffer structure
References

Thanks for your attention