A Deep Convolutional Neural Network Based on Nested Residue Number System

Hiroki Nakahara\textsuperscript{1} Tsutomu Sasao\textsuperscript{2}

\textsuperscript{1}Ehime University, Japan
\textsuperscript{2}Meiji University, Japan
Outline

- Background
- Deep convolutional neural network (DCNN)
- Residue number system (RNS)
- DCNN using nested RNS (NRNS)
- Experimental results
- Conclusion
Background

• **Deep Neural Network**
  – Multi-layer neuron model
  – Used for embedded vision system

• **FPGA realization is suitable for real-time systems**
  – faster than the CPU
  – Lower power consumption than the GPU
  – Fixed point representation is sufficient

• **High-performance per area is desired**
Deep Convolutional Neural Network (DCNN)
Artificial Neuron

\[ y = f(u) \]

\[ u = \sum_{i=0}^{N} w_i x_i \]

- \( x_i \): Input signal
- \( w_i \): Weight
- \( u \): Internal state
- \( f(u) \): Activation function (Sigmoid, ReLU, etc.)
- \( y \): Output signal
Deep Convolutional Neural Network (DCNN) for ImageNet

- 2D convolutional layer, pooling layer, and fully connection layer
2D Convolutional Layer

- Consumes more than 90% of the computation time
  - Multiply–accumulation (MAC) operation is performed

\[ z_{ij} = y_{ij} + \sum_{m=0}^{K-1} \sum_{n=0}^{K-1} x_{i+m, j+n} w_{mn} \]

- \( x_{ij} \): Input signal
- \( y_{ij} \): Bias
- \( w_{mn} \): Weight
- \( K \): Kernel size
- \( z_{ij} \): Output signal
Realization of 2D Convolutional Layer

• Requires more than billion MACs!
• Our realization
  – Time multiplexing
  – Nested Residue Number System (NRNS)
Residue Number System (RNS)
Residue Number System (RNS)

- Defined by a set of L mutually prime integer constants \(<m_1, m_2, \ldots, m_L>\)
  - No pair modulus have a common factor with any other
  - Typically, prime number is used as moduli set

- An arbitrary integer X can be uniquely represented by a tuple of L integers \((X_1, X_2, \ldots, X_L)\), where \(X_i \equiv X \pmod{m_i}\)

- Dynamic range

\[ M = \prod_{i=1}^{L} m_i \]
Multiplication on RNS

- Moduli set $\langle 3, 4, 5 \rangle$, $X = 8$, $Y = 2$
- $Z = X \times Y = 16 = (1, 0, 1)$
- $X = (2, 0, 3)$, $Y = (2, 2, 2)$
- $Z = (4 \mod 3, 0 \mod 4, 6 \mod 5) = (1, 0, 1) = 16$

Binary2RNS Conversion
Parallel Multiplication
RNS2Binary Conversion
# Binary2RNS Converter

<table>
<thead>
<tr>
<th>X</th>
<th>mod 2</th>
<th>mod 3</th>
<th>mod 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Functional Decomposition

Bound variables
\[ X_1 = (x_1, x_2) \]

Free variables
\[ X_2 = (x_3, x_4) \]

<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>01</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>01</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

| x1    | 0  | 0  | 1  | 1  |
| x2    | 0  | 1  | 0  | 1  |
| h(X1) | 0  | 1  | 0  | 1  |

\[ h(X_1) = 0101 \]

Column multiplicity = 2

\[ 2^4x1 = 16 \text{ [bit]} \]

\[ 2^2x1 + 2^3x1 = 12 \text{ [bit]} \]
# Decomposition Chart for $X \mod 3$

The chart shows the decomposition of $X$ into $X_1$ and $X_2$ for $X = (x_1, x_2, x_3, x_4, x_5)$.

<table>
<thead>
<tr>
<th>$X_1 = (x_1, x_2)$</th>
<th>00</th>
<th>01</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>01</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$X_2 = (x_3, x_4, x_5)$</th>
<th>000</th>
<th>001</th>
<th>010</th>
<th>011</th>
<th>100</th>
<th>101</th>
<th>110</th>
<th>111</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>001</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>010</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>011</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

- $0 \mod 3 = 0$
- $1 \mod 3 = 1$
- $2 \mod 3 = 2$
- $3 \mod 3 = 0$
- $4 \mod 3 = 1$
- $5 \mod 3 = 2$
- $6 \mod 3 = 0$
- $7 \mod 3 = 1$
- $8 \mod 3 = 2$
- $9 \mod 3 = 0$
- $10 \mod 3 = 1$
- $\vdots$
Decomposition Chart for $X \mod 3$

<table>
<thead>
<tr>
<th>$x_3$</th>
<th>0 0 0 0 1 1 1 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_4$</td>
<td>0 0 1 1 0 0 1 1</td>
</tr>
<tr>
<td>$x_5$</td>
<td>0 1 0 1 0 1 0 1</td>
</tr>
<tr>
<td>$h(X_2)$</td>
<td>0 1 2 0 1 2 0 1</td>
</tr>
</tbody>
</table>

| 0 mod 3 = 0 |
| 1 mod 3 = 1 |
| 2 mod 3 = 2 |
| 3 mod 3 = 0 |
| 4 mod 3 = 1 |
| 5 mod 3 = 2 |
| 6 mod 3 = 0 |
| 7 mod 3 = 1 |
| 8 mod 3 = 2 |
| 9 mod 3 = 0 |
| 10 mod 3 = 1 |
| ... |
Binary2RNS Converter

LUT cascade for \( X \mod m_1 \)

LUT cascade for \( X \mod m_2 \)

\( X_n \)  

\( k \)

\( k-r \)

\( \log_2 m_L \)

LUT cascade for \( X \mod m_L \)

pipeline registers
RNS2Binary Converter \((m=30)\)

\[
\begin{array}{c|c}
\hline
x_1 & y_1 \\
\hline
0 & 0 \\
1 & 15 \\
\hline
\end{array}
\]

\[
\begin{array}{c|c}
\hline
x_2 & y_2 \\
\hline
0 & 0 \\
1 & 10 \\
2 & 20 \\
\hline
\end{array}
\]

\[
\begin{array}{c|c}
\hline
x_3 & y_3 \\
\hline
0 & 0 \\
1 & 6 \\
2 & 12 \\
3 & 18 \\
4 & 24 \\
\hline
\end{array}
\]

\[
X + Y \quad 2^{j-m} \\
\]

\[
(X+Y) \mod m \\
\]
Problem

- Moduli set of RNS consists of mutually prime numbers
  - sizes of circuits are all different
- Example: \(<7,11,13>\)
DCNN using Nested RNS
Nested RNS

- \((Z_1, Z_2, \ldots, Z_i, \ldots, Z_L) \rightarrow (Z_1, Z_2, \ldots, (Z_{i_1}, Z_{i_2}, \ldots, Z_{i_j}), \ldots, Z_L)\)
- Ex: \(<7, 11, 13> \times <7, 11, 13>\)

\(<7, <5, 6, 7>_{11}, <5, 6, 7>_{13}> \times <7, <5, 6, 7>_{11}, <5, 6, 7>_{13}>\)

1. Reuse the same moduli set
2. Decompose a large modulo into smaller ones
Example of Nested RNS

- $19 \times 22 (=418)$ on $\langle 7, \langle 5,6,7 \rangle_{11}, \langle 5,6,7 \rangle_{13} \rangle$

$19 \times 22$

$= \langle 5,8,6 \rangle \times \langle 1,0,9 \rangle$

$= \langle 5, \langle 3,2,1 \rangle_{11}, \langle 1,0,6 \rangle_{13} \rangle \times \langle 1, \langle 0,0,0 \rangle_{11}, \langle 4,3,2 \rangle_{13} \rangle$

$= \langle 5, \langle 0,0,0 \rangle_{11}, \langle 4,0,5 \rangle_{13} \rangle$

$= \langle 5,0,2 \rangle$

$= 418$

Diagram:

- Binary2NRNS Conversion
- Modulo Multiplication
- Bin2RNS on NRNS
- RNS2Bin
Realization of Nested RNS

Realized by BRAMs

Binary 2NRNS

NRNS2 Binary

LUTs

BRAMs and DSP blocks
Moduli Set for NRNS

- Conventional RNS (uses 23 moduli)
  \(<3,4,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83>\>

- Applied the NRNS to moduli that are greater than 15
  \(<3,4,5,7,11,13,\<3,4,5,7,11,13>\>_{17},\<3,4,5,7,11,13>_{19},\<3,4,5,7,11,13,\<3,4,5,7,11,13>\>_{17}\>_{23},\<3,4,5,7,11,13,\<3,4,5,7,11,13>\>_{17}\>_{29},\ldots,\<3,4,5,7,11,13,\<3,4,5,7,11,13>\>_{17}\>_{83}>\>

All the 48-bit MAC operations are decomposed into 4-bit ones
Experimental Results
Implementation Setup

- FPGA board: Xilinx VC707
  - FPGA: Virtex7 VC485T
  - 1GB DDR3SODIMM
    (Bus@800MHz, 64 bit width)
- Realized the pre-trained ImageNet by Convnet2
  - 48-bit fixed precision
- Synthesis tool: Xilinx Vivado2014.1
  - Timing constrain: 400MHz
## Comparison with Other Implementations

<table>
<thead>
<tr>
<th>Precision</th>
<th>Max. Freq. [MHz]</th>
<th>FPGA</th>
<th>Performance [GOPS]</th>
<th>Performance per area [GOPS/Slice x 10^{-4}]</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASAP2009</td>
<td>16bit fixed</td>
<td>115 Viretex5 LX330T</td>
<td>6.7</td>
<td>1.3</td>
</tr>
<tr>
<td>PACT2010</td>
<td>--- fixed</td>
<td>125 Viretex5 SX240T</td>
<td>7.0</td>
<td>1.9</td>
</tr>
<tr>
<td>FPL2009</td>
<td>48bit fixed</td>
<td>125 Spartax3A DSP3400</td>
<td>5.3</td>
<td>2.2</td>
</tr>
<tr>
<td>ISCA2010</td>
<td>48bit fixed</td>
<td>200 Viretex5 SX240T</td>
<td>16.0</td>
<td>4.3</td>
</tr>
<tr>
<td>ICCD2013</td>
<td>--- fixed</td>
<td>150 Viretex6 LVX240T</td>
<td>17.0</td>
<td>4.5</td>
</tr>
<tr>
<td>FPGA2015</td>
<td>32bit float</td>
<td>100 Virtex7 VX485T</td>
<td>61.6</td>
<td>8.1</td>
</tr>
<tr>
<td>Proposed</td>
<td>48bit fixed</td>
<td>400 Virtex7 VX485T</td>
<td>132.2</td>
<td>25.2</td>
</tr>
</tbody>
</table>
Conclusion

• Realized the DCNN on the FPGA
  – Time multiplexing
  – Nested RNS
    • MAC operation is realized by small LUTs
    • Functional decomposition are used as follows:
      – Bin2NRNS converter is realized by BRAMs
      – NRNS2Bin converter is realized by DSP blocks and BRAMs

• Performance per area (GOPS/Slice)
  – 5.86 times higher than ISCA10’s