Algorithm Innovations of Enhancing Scalability and Adaptability of Learning Systems

Yiran Chen
Evolutionary Intelligence Lab (EI-Lab)
University of Pittsburgh

Neuromorphic Computing Workshop 2016
@Oak Ridge National Laboratory
## Mismatch: Hardware vs. Software

<table>
<thead>
<tr>
<th></th>
<th>Hardware</th>
<th>Software</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model/Component scale</strong></td>
<td>Small/Moderate</td>
<td>Large</td>
</tr>
<tr>
<td><strong>Re-configurability</strong></td>
<td>Hard</td>
<td>Easy</td>
</tr>
<tr>
<td><strong>Accuracy vs. Power</strong></td>
<td>Tradeoff</td>
<td>Accuracy</td>
</tr>
<tr>
<td><strong>Training implementation</strong></td>
<td>Hard</td>
<td>Easy</td>
</tr>
<tr>
<td><strong>Precision vs. Limited programmability</strong></td>
<td>Low precision (often a few bits)</td>
<td>Double (high) precision</td>
</tr>
<tr>
<td><strong>Connectivity realization</strong></td>
<td>Hard</td>
<td>Easy</td>
</tr>
</tbody>
</table>
Precision & Limited programmability

Decrease precision without re-training

Re-train low precision AlexNet on ImageNet

<table>
<thead>
<tr>
<th>Precision</th>
<th>Top-5 Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>32-bit floating point</td>
<td>80.3%</td>
</tr>
<tr>
<td>16-bit floating point</td>
<td>80.3%</td>
</tr>
<tr>
<td>8-bit fixed point</td>
<td>80.1%</td>
</tr>
<tr>
<td>4-bit fixed point</td>
<td>14.0%</td>
</tr>
<tr>
<td>2-bit fixed point</td>
<td>0.9%</td>
</tr>
</tbody>
</table>

Example: TrueNorth
TrueNorth Chipset Architecture

The **IBM TrueNorth Dev Board.**

- 4,096 neurosynaptic cores;
- 1 million neurons;
- 256 million synapses;
- A 65mW real-time neurosynaptic processor.

---

A Network of Neurosynaptic Cores

A 256×256 Synaptic Crossbar

Low-resolution Integer Weight

Spike Communication
Overview of TrueNorth Operation

Database

Raw Pixels Testing

Spike Encoder

Binary Spikes

TrueNorth model with low resolution synaptic integers

Raw Pixels Training

Tea learning in Caffe (Traditional CPUs/GPUs)

A network with floating-point weights

Tea deploying in CPE (Corelet Programming Environment)

* S. K. Esser, etc., NIPS’15
Learning and Deploying of **TrueNorth**

**Mapping Neural Networks in IBM TrueNorth**

![Diagram of TrueNorth architecture]

**Traditional Neural Networks**

Floating-point Precision

\[ y = w \cdot x + b \]
\[ z = h(y) \]

**Neural Networks with TrueNorth**

Binary/low Integer precision

McCulloch-Pitts neuron model:

\[ y' = w' \cdot x' - \lambda \]
\[ z' = \begin{cases} 
1, & \text{Reset } y'=0; \\
0, & \text{Reset } y'=0; \\
1, & \text{If } y'>=0 \\
0, & \text{If } y'<0 . 
\end{cases} \]

**MNIST with TrueNorth**

![MNIST database example]

Block Size (16)

256

Block Stride
Learning and Deploying of TrueNorth

Connectivity Probabilities $P$

![Connectivity Probabilities Diagram]

Connectivity Samples

![Connectivity Samples Diagram]

(a) Tea learning
Traditional Floating Point Precision

(b) Tea deploying
Binary/low integer precision sampled by float-point probability

MNIST Accuracy:
- 95.27% in Caffe
- 90.04% @1 NN copy & 1 spf in TrueNorth
- 92.74% @ 1 NN copy & 4 spf in TrueNorth
- 94.63% @16 NN copies & 1 spf in TrueNorth

(FF - neural networks; spf – spikes per frame)
Learning and Deploying of TrueNorth

Connectivity Probabilities $P$

- $x$: 0.75
- $y$: 0.20
- $z$: 0.60

Spike Probabilities:
- $x$: 0.8
- $y$: 0.5
- $z$: 0.1

(a) Tea learning

Connectivity Samples

- $w'$
- $x'$
- $y'$

(b) Tea deploying

$\Delta y = y' - y = \sum_{i=0}^{n-1} w' x' - \sum_{i=0}^{n-1} w x_i$

$E\{\Delta y\} = 0$

$\text{var}\{\Delta y\} = \sum_{i=0}^{n-1} \text{var}\{w' x_i\}$

Unbiased approximation $w/\text{variance}$ is affected by both synaptic and spiking randomness.

Spiking Randomness:
Determined by External Data

Synaptic Randomness:

$\text{var}\{w_i\} = E\{w_i^2\} - E\{w_i\}^2 = p_i(1-p_i)$
Minimizing Deployment Variance

**Minimization target:**

\[
\hat{E}(w) = E_D(w) + \alpha \times E_p(P)
\]

- **Data Loss**
- **Probability Regulation**

\[
E_{p_1}(P) = |||P - a| - b||
\]

\[
= \sum_{i=1}^{M} |p_i - a| - b
\]

**Variance Evaluation**

- **Baseline**
  - @ 1 Network copy & 1 spf
  - Accuracy = 90.04%

- **Probability Regularization**
  - Accuracy = 92.78%

**Synaptic Deviation**

- Number of Synapses
- Probability
- Synaptic Deviation

**Graphs:**

- Baseline vs. Probability Regularization
- Accuracy comparison

- Color scale: 0.00 to 1.00
**Experiment Results**

<table>
<thead>
<tr>
<th>Test Bench</th>
<th>Dataset</th>
<th>Block stride</th>
<th>Hidden Layer #</th>
<th>Cores per Layer</th>
<th>Accuracy in Caffe</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MNIST</td>
<td>12</td>
<td>1</td>
<td>4</td>
<td>95.27%</td>
</tr>
</tbody>
</table>

- **33.4% Avg. Core Reduction without Accuracy loss**
- **Only Tea learning is changed. No hardware modification**

*16 and 9 correspond to the cores utilized by 1st and 2nd hidden layer.*
Sparsity & Computation Cost

<table>
<thead>
<tr>
<th>Serving Layer</th>
<th>Dense</th>
<th>Random sparsity</th>
<th>Structured sparsity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network</td>
<td><img src="image1.png" alt="Dense Network" /></td>
<td><img src="image2.png" alt="Random sparsity Network" /></td>
<td><img src="image3.png" alt="Structured sparsity Network" /></td>
</tr>
<tr>
<td>Deployed weight matrix</td>
<td><img src="image4.png" alt="Dense Weight" /></td>
<td><img src="image5.png" alt="Random sparsity Weight" /></td>
<td><img src="image6.png" alt="Structured sparsity Weight" /></td>
</tr>
<tr>
<td>Memory occupancy</td>
<td>Full &amp; Large</td>
<td>Ineffective &amp; Large</td>
<td>Full &amp; Small</td>
</tr>
<tr>
<td>Power</td>
<td>High</td>
<td>Still high</td>
<td>Low</td>
</tr>
</tbody>
</table>
• Neuromorphic engineering, also known as **neuromorphic computing**, is a concept developed by Carver Mead in the late 1980s, describing the use of **very-large-scale integration (VLSI) systems** containing electronic analog circuits to mimic **neuro-biological** architectures present in the nervous system. (wikipedia.org)
  – Hardware (computing) and Software (bio-model and algorithm)

• “Chicken and Egg”
  – Another hardware-software co-design problem?
  – Harder than that as we do not have a solid theory and implementation foundation.

• Coordination and Standardization are needed in research of neuromorphic computing.
Sponsors