Neural-inspired computing algorithms and hardware for image analysis and cybersecurity applications

Conrad D. James, Ph.D.
Sandia National Laboratories

The Salishan Conference on High-Speed Computing, April 2017
Acknowledgments

**Algorithms:** Brad Aimone, Ojas Parekh, Nadine Miner, Sandra Faust, Steve Verzi, Frances Chance, Tu-Thach Quach, Chris Lamb, Meghan Galiardi, Samuel Mulder, William Severa, Kristofor Carlson, Michael Smith, Cynthia Phillips, Jacob Hobbs, Robert Abbott; Jeffrey Piersol,

**Architecture:** John Naegle, Alex Hsia, Sapan Agarwal, Craig Vineyard, Fred Rothganger, Jonathon Donaldson, Gabriel Popoola, Aaron Hill

**Learning Hardware:** Matt Marinella, Thomas Beechem, Ron Goeke, Alec Talin, Paul Kotula, Farid El Gabaly, Elliot Fuller, Jim Stevens, David Hughart, Andy Armstrong, David Henry, Gaadi Haase, Steve Wolfley, Seth Decker, Christopher Saltonstall, Jamison Wagner, John Niroula, Derek Wilke, Michael Van Heukelom, Patrick Finnegan, Carl Smith, Robin Jacobs-Gedrim

**Modeling & Applications:** Steve Plimpton, Justin Doak, Richard Schiek, Brian Tierney, Robert Bondi, Harry Hjalmarson, Tim Draelos, Jonathan Cox, Joe Ingram, Jason Wheeler

**Partnerships:** David Follett, Duncan Townsend (Lewis Rhodes Labs); Pamela Follett; Isaac Richter, Engin Ipek (U. Rochester); Felix Wang (UIUC); David Lidsky, Marek Osinski (UNM);
Data-driven computing (machine learning) is necessary for real-world problems

C. Lampert, VRML 2013

Conventional numerical computing

Data-driven computing

yann.lecun.com
Data-driven (neural-inspired) computing has a complicated history…and mixed results
Neural-inspired algorithms are achieving success but several challenges remain

Karpathy etc. NIPS 2014, 1889

Kemelmacher et al., CVPR 2016
Neural computing at Sandia Labs leverages a large research foundation.
Hardware Acceleration of Adaptive Neural Algorithms (HAANA)

Algorithms
- Non-spiking
- Spiking

Architectures
- Filamentary

Learning Hardware
- Electrochemical
- Imaging
- Cyber

Neurogenesis deep learning: Draelos et al, IJCNN 2017
Spiking network algorithms: Severa et al., ICRC 2016
Digital neuromorphic architecture: Smith et al., IJCNN 2017

Resistive switching model: Mickel et al, Adv Mater 2014
Electrochemical transistor: Fuller et al., Adv Mater 2016
Resistive crossbar accelerator: Agarwal et al., IJCNN 2016
Translating neuroscience into the next generation of computing – Neural Machine learning (NML)

Identify neurobiological circuits of interest

Simulate at high level of neural fidelity

Formalize & optimize neural algorithms

Translate into NML algorithm

Identify critical aspects of computation
Example cyber problem: file identification using deep neural networks

\[ \begin{align*}
  f(z_1) &= \tanh(z_1) \\
  f(z_2) &= \tanh(z_2) \\
  f(z_3) &= \text{softmax}(z_3)
\end{align*} \]

\( N_1 = 512 \)

\( N_2 = 256 \)
Limitations of supervised machine learning

- Supervised learning requires subject matter experts to hand-craft features
- Data-driven algorithms are limited...by the data

Cox, Aimone, James; Complex Adaptive Systems, Nov. 2015; Procedia Comp Sci 61, 349
Translating neuroscience into the next generation of computing

1. Identify neurobiological circuits of interest
2. Simulate at high level of neural fidelity
3. Identify critical aspects of computation
4. Formalize & optimize neural algorithms
5. Translate into NML algorithm
Leveraging computational neuroscience models to develop new algorithms

Vineyard et al., IJCNN 2016, DOI: 10.1109/IJCNN.2016.7727884
Modeling the pattern separation function of the hippocampus

Similar EC inputs lead to the dentate gyrus.

Increased sparsity leads to decorrelated DG outputs.

\[ NDP(x_i, x_j) = \frac{x_i \cdot x_j}{\|x_i\| \times \|x_j\|} \]

Severa et al., Neural Computation 2017, 29, 94
Quantifying the sparsity transformation in the hippocampus

This approach is feasible for well-defined inputs – need to formalize an algorithm to account for unknown inputs.

Severa et al., Neural Computation 2017, 29, 94
Algorithm inspiration: neurogenesis in biological networks

Adult neurogenesis improves information capacity…?
Neurogenesis may provide flexible encoding strategies for particular brain regions.

Aimone, Deng and Gage, Neuron 2011, 70, 589

Uncoded novel input

Differentiated novel input

Severa et al., Neural Computation 2017, 29, 94
Translating neuroscience into the next generation of computing

Identify neurobiological circuits of interest

Simulate at high level of neural fidelity

Formalize & optimize neural algorithms

Translate into NML algorithm

Identify critical aspects of computation
Data-driven computing methods are limited… by data

Draelos et al, ICLR 2016, IJCNN 2017
“Neurogenic deep learning” enables adaptation to changing data

Draelos et al, ICLR 2016, IJCNN 2017
Translating neuroscience into the next generation of computing

Identify neurobiological circuits of interest

Formalize & optimize neural algorithms

Simulate at high level of neural fidelity

Identify critical aspects of computation

Translate into NML algorithm
Categorizing cyber data under imperfect conditions with sparse coding

- Training data is not always available, fragmented
- Limited expertise; hand-engineered features

\[ y \approx Ax \]
\[ \min_x \|y - Ax\|^2 + S(x) \]

 Fragment Learned Dictionary

Sparse Representation
Generating local and global features from file fragments

- Byte dictionary patches & sparse representations of fragments – local features

- Long short-term memory (LSTM) networks are used to improve long-range correlations – global features

Graves et al., ASRU2013
Hochreiter & Schmidhuber, Neur Comp 1997

Wang et al., in preparation
Averaged $F_1$ score = 53.12%

NLP (SVM) approach achieved $49.1 \pm 3.15\%$ (Fitzgerald et al., DI 2012)
Translating neuroscience into the next generation of computing

Identify neurobiological circuits of interest

Simulate at high level of neural fidelity

Identify critical aspects of computation

Formalize & optimize neural algorithms

Translate into NML algorithm
Optimization of algorithm performance

Neural algorithm operations are computationally expensive (energy and time) due to training; many matrix-vector operations

\[
\min_x \|y - Ax\|^2 + S(x)
\]

Sparse coding: \[
\Delta_k = \sum_k w_{jk} \delta_k
\]

Backpropagation: \[
 f \ast g = F^{-1}\{F\{f\} \cdot F\{g\}\}
\]

Hardware accelerate algorithms:

\[
\left| j \right\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega^{jk} \left| k \right\rangle
\]

Lee et al., Proc World Cong Eng Comp Sci 2013
Hardware acceleration of spiking algorithms for time-dependent data processing

Example: liquid state machine (LSM); a tool for data transformation; randomly connected spiking neurons encode complex *temporal dynamics*

Spiking algorithms are often inefficient on conventional hardware…
Hardware acceleration of algorithm operations

- Ovtcharov et al., (Microsoft), FPGA acceleration of CNNs, 2015
- Gokmen & Vlasov., (IBM), Resistive crossbar acceleration of DNNs, 2016
- Coates et al., (Nvidia, Stanford), Deep learning with GPUs, 2013
- Gokhale et al., (Purdue), nnX for accelerating DNNs with ARMs, 2013
Spiking Temporal Processing Unit (STPU)

Impart complex temporal dynamics into neural networks

$\sum_{k} i_k(t) w_{k,j}(t)$

$R_{D-1}$

$R_0$

LIF

$t = t_1$

$t = t_2$

$W_{ij} e^{-\frac{(t - \Delta t)}{\tau}}$

Smith et al, “A Novel Digital Neuromorphic Architecture…”, IJCNN 2017
Emulation of a LSM mapped onto an STPU architecture

Assemble an array of LIF neurons and combine with a synaptic map $W$

• Test data: spoken digits (0-9)
• Implement the liquid on the STPU
• Use a classifier to categorize the spoken digits

<table>
<thead>
<tr>
<th>Linear Model</th>
<th>3x3x15</th>
<th>5x5x5</th>
<th>4x5x10</th>
<th>2x2x20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear SVM</td>
<td>0.906</td>
<td>0.900</td>
<td>0.900</td>
<td>0.914</td>
</tr>
<tr>
<td>LDA</td>
<td>0.921</td>
<td>0.922</td>
<td>0.922</td>
<td>0.946</td>
</tr>
<tr>
<td>Ridge Regress</td>
<td>0.745</td>
<td>0.717</td>
<td>0.717</td>
<td>0.897</td>
</tr>
<tr>
<td>Logistic Regress</td>
<td>0.431</td>
<td>0.254</td>
<td>0.254</td>
<td>0.815</td>
</tr>
</tbody>
</table>

Smith et al, “A Novel Digital Neuromorphic Architecture…”, IJCNN 2017
Implementing synaptic connections in hardware for non-spiking neural algorithms

Use variable resistors to implement neural network weights in hardware – saves energy $O(n^3)$ to $O(n^2)$

$w_{ij} = G_{ij} = (R_{ij})^{-1}$

Agarwal et al., Front. Neuroscience 2016, 9, 484
Agarwal et al, IJCNN 2016, DOI: 10.1109/IJCNN.2016.7727298
Designing, modeling, and fabricating devices with improved neural computing characteristics

**Filament surface temperature ($T_s$):**

$$T_s = T_{RT} + \sigma V^2 \frac{d_E}{2k_E d_o} \left[ 1 - \frac{k_E}{k_F} \frac{r_F^2}{4d_E d_o} \right]$$

**References:**

- Mickel et al., Adv Mater, 26, 4486, 2014
- Landon et al., APL 2015, 107, 023108
- Fuller et al., Adv Mater 2016, 10.1002/adma.201604310
- van de Brugt et al., Nat Materials 2017, 10.1038/nmat4856
Model hardware-acceleration to assess the impact on algorithm performance

Agarwal et al, IJCNN 2016, DOI: 10.1109/IJCNN.2016.7727298
Translating neuroscience into the next generation of computing

Identify neurobiological circuits of interest

Simulate at high level of neural fidelity

Identify critical aspects of computation

Formalize & optimize neural algorithms

Translate into NML algorithm
Thanks for your time!
Questions?
Backup Slides
Applications in imaging and cybersecurity

- The real world is filled with massive amounts of data
- Data needs to be filtered to capture relevant information
- Signatures or features need to be extracted from data
- Features can then be used to interpret activities
Spiking network algorithm for computing cross-correlations

- Motivation: determine the local velocity in a flow field
- SNN algorithms are highly parallel and can leverage the time/neuron tradeoff
- Neural algorithms can match or best traditional ‘big O’
Trading neurons for time and vice versa

- $O(n^2)$ neurons, constant time
- $O(n)$ neurons, $O(n)$ time
Matrix operations are at the core of many neural computing operations.

Backpropagation:

$$\Delta_k = \sum_k w_{jk} \delta_k$$

Graph Analysis:

Naïve algorithm for matrix multiplication is $O(N^3)$.
Strassen matrix multiplication

\[
\begin{bmatrix}
A_{11} & A_{12} \\
A_{21} & A_{22}
\end{bmatrix}
\times
\begin{bmatrix}
B_{11} & B_{12} \\
B_{21} & B_{22}
\end{bmatrix}
= \begin{bmatrix}
C_{11} & C_{12} \\
C_{21} & C_{22}
\end{bmatrix}
\]

**Standard**

\[
\begin{align*}
C_{11} &= A_{11}B_{11} + A_{12}B_{21} \\
C_{12} &= A_{11}B_{12} + A_{12}B_{22} \\
C_{21} &= A_{21}B_{11} + A_{22}B_{21} \\
C_{22} &= A_{21}B_{12} + A_{22}B_{22}
\end{align*}
\]

**Strassen**

\[
\begin{align*}
M_1 &= (A_{11} + A_{22})(B_{11} + B_{22}) \\
M_2 &= (A_{21} + A_{22})B_{11} \\
M_3 &= A_{11}(B_{12} - B_{22}) \\
M_4 &= A_{22}(B_{21} - B_{11}) \\
M_5 &= (A_{11} + A_{12})B_{22} \\
M_6 &= (A_{21} - A_{11})(B_{11} + B_{12}) \\
M_7 &= (A_{12} - A_{22})(B_{21} + B_{22})
\end{align*}
\]

\[
\begin{align*}
C_{11} &= M_1 + M_4 - M_5 + M_7 \\
C_{12} &= M_3 + M_5 \\
C_{21} &= M_2 + M_4 \\
C_{22} &= M_1 - M_2 + M_3 + M_6
\end{align*}
\]

**Standard:** 8Ms, 4As $\rightarrow O(N^3)$

**Strassen:** 7Ms, 18A/Ss $\rightarrow O(N^{2+\varepsilon})$

*Strassen, Num Math, 1969*
“Neural” network for matrix multiplication

Strassen formulation of matrix multiply enables less than $O(N^3)$ neurons – resulting in less power consumption
Resistive switching devices

\[ V = I \times R \]
\[ I = G \times V \]

\[ I_1 = I_1 + I_2 \]

DRAM | NAND Flash | PC-RAM | STT-MRAM | FeRAM | ReRAM | CBRAM
--- | --- | --- | --- | --- | --- | ---
**Maturity** | Production (20 nm) | Production (16 nm) | Production (45 nm) | Production (65 nm) | Production (180 nm) | Production (180 nm) | Production (180 nm)
Min device feature F (nm) | 20 | 16 | <10 | 16 | 28 nm | 5 | 20 (5 est.)
Density \((F^2)\) | 6 | 16 (single layer) | 4 | 8-20 | 22 | 4 | 4
Write Time (ns) | < 10 | 10000 | 50 | 13 | <100 | 2 | 2
Write Energy (pJ/bit) | 0.005 | 100 | 6 | 4 | 270 | <1 | <1
Endurance (W/E Cycles) | \(>10^{16}\) | \(10^4\) | \(>10^9\) | \(10^{12}\) | \(10^{14}\) | \(10^{12}\) | \(10^{10}\)
Retention | 64 ms | 1 - 10 y | > 10 y | weeks | > 10 y | > 10 y | > 10 y
Stackable | No | Yes | Yes | No | No | Yes | Yes
Process complexity | High/FE | High/FE | Low/BE | High/BE | High/BE | Low/BE | Low/BE

***many of these numbers are not universally agreed on***
ReRAM is $O(N)$ better than SRAM in energy consumption for vector-matrix multiply computations

SRAMs must fetch each vector per dot product $\sim O(N^2 \times M)$

Analog computation: multiplier and adder at each intersection; $E \sim CV^2 \sim O(N \times M)$

$V_1 = x_1$  $- +$
$V_2 = x_2$  $- +$
$V_3 = x_3$  $- +$
$V_4 = x_4$  $- +$

$I_1 = x_1 * w_{11} + \ldots + x_4 * w_{41}$

Agarwal et al., Front. Neuroscience 2016, 9, 484