Novel Technologies for Artificial Intelligence: prospects and challenges

Stefano Ambrogio
Pritish Narayanan
Robert M. Shelby
Hsinyu Tsai
Geoffrey W. Burr
IBM Research – Almaden
Outline

- Introduction
- A brain-inspired algorithm: **Spike Timing Dependent Plasticity**
- A machine learning algorithm: **Back-Propagation**
- Analog memory for training Neural Networks
- Software-equivalent accuracy with novel unit cell
- Circuit design considerations
- Conclusion
Outline

- Introduction
- A brain-inspired algorithm: Spike Timing Dependent Plasticity
- A machine learning algorithm: Back-Propagation
- Analog memory for training Neural Networks
- Software-equivalent accuracy with novel unit cell
- Circuit design considerations
- Conclusion
What is AI?

Artificial Intelligence

Machine Learning

Neural Networks

Deep Learning

Brain Inspired Algorithms
2012: AI foundations

The Deep Learning Explosion

YouTube
400 hours of video uploaded every minute

Walmart
2.5 petabytes of customer data hourly

Facebook
350 million images uploaded daily

ImageNet Classification Error

2010 2011 2012 2013 2014 2015 Human

Deep Neural Networks
GPU Hardware Accelerators
Tasks performed by specialized AI

- Language Translation
- Speech Transcription
- Language Processing
- Object Detection
- Face Recognition
Example AI challenges now tackled

**Design Automation**
- AI based design

**Industrial**
- Guide me through fixing malfunctioning components

**Healthcare**
- Improve the accuracy of breast cancer screening

**Visual Inspection**
- Find rust on electric towers, using drones

**Customer Care**
- Bot that can guide a user through buying the right insurance policy

**Marketing / Business**
- Summarize the strategic intent of a company based on recent news articles

**IoT**
- Predict yield of field based on images and sensor data

**Compliance**
- Is my organization compliant with latest regulatory documents

Stefano Ambrogio, IBM Research - Almaden
30 November 2018
The evolution of AI

Narrow AI
Emerging

广义人工智能
革命性的

宽泛人工智能
颠覆性与普及性

我们在这里
2050年及以后
The evolution of AI

Narrow AI
Single task, single domain
Superhuman accuracy and speed for certain tasks

Broad AI
Multi-task, multi-domain
Multi-modal
Distributed AI
Explainable

General AI
Cross-domain learning and reasoning
Broad autonomy
Neuromorphic hardware

The choice of which hardware and algorithm to use strongly depends on the desired application.
Which Algorithm?

Different algorithms provide different solutions

Brain Inspired Algorithms

- Provide unsupervised online learning
  $\rightarrow$ Learning of novel unseen classes

- Not very good at classification, accuracy on datasets generally lower than Back-propagation

- The brain-inspired conception is not useful as a feature, more interest in the result we can achieve

Back-Propagation

- Provides highest achievable accuracy
  Algorithm very well known

- Classification of novel unseen classes not available (Catastrophic forgetting)
  Need for medium/large training datasets

- The choice of the network type (Fully Connected, Convolutional, …) depends on the available power, area and complexity constraints

Stefano Ambrogio, IBM Research - Almaden
30 November 2018
Spike or continuous implementation?

It is generally believed that Spike implementations (whether bio-inspired or not) provide less power consumption due to chip activation only when needed.

It depends on several factors…

In case the information is NATURALLY spike based
  → typically very power-efficient

In case the information is NON-Spike based
  → the circuitry to convert information into spike consumes power, which sometimes is not taken into consideration

Spike implementation is generally slower to react, since it needs to capture many spikes to provide right answers.
Outline

- Introduction
- A brain-inspired algorithm: Spike Timing Dependent Plasticity
- A machine learning algorithm: Back-Propagation
- Analog memory for training Neural Networks
- Software-equivalent accuracy with novel unit cell
- Circuit design considerations
- Conclusion
Brain Inspired: Spike Timing Dependent Plasticity

- Biological protocol for synapse weight update

Long Term Potentiation $\Delta t > 0$

Long Term Depression $\Delta t < 0$

$
\Delta G$
$
\Delta t$

Unsupervised learning of features

- The network enables robust implementation of pattern learning and recognition.

Outline

- Introduction
- A brain-inspired algorithm: **Spike Timing Dependent Plasticity**
- A machine learning algorithm: **Back-Propagation**
- Analog memory for training Neural Networks
- Software-equivalent accuracy with novel unit cell
- Circuit design considerations
- Conclusion
What is backpropagation?

- Global learning rule for training Fully Connected and Convolutional Neural Networks

\[ x_{bi} = f(\sum x_{ai}w_i) \]

- The neuron output is the weighted sum of the inputs, passed through a nonlinearity
Forward propagation

Cropped MNIST images

NN results

1 → y_1
2 → y_2
3 → y_3
... 
8 → y_8
9 → y_9
0 → y_{10}
Comparison with the correct answer

Cropped MNIST images

528 input neurons

250 hidden neurons

125 hidden neurons

10 output neurons

NN results

Correction

1 → $y_1$ - $g_1$ → $\delta_1$

2 → $y_2$ - $g_2$ → $\delta_2$

3 → $y_3$ - $g_3$ → $\delta_3$

8 → $y_8$ - $g_8$ → $\delta_8$

9 → $y_9$ - $g_9$ → $\delta_9$

0 → $y_{10}$ - $g_{10}$ → $\delta_{10}$

Correct answer
Backpropagation

- 528 input neurons
- 250 hidden neurons
- 125 hidden neurons
- 10 output neurons

Cropped MNIST images

1. $x_1 \rightarrow y_1 - g_1 \rightarrow \delta_1$
2. $x_2 \rightarrow y_2 - g_2 \rightarrow \delta_2$
3. $x_3 \rightarrow y_3 - g_3 \rightarrow \delta_3$
4. $\ldots$
5. $x_{250} \rightarrow y_8 - g_8 \rightarrow \delta_8$
6. $x_{528} \rightarrow y_9 - g_9 \rightarrow \delta_9$
7. $\rightarrow y_{10} - g_{10} \rightarrow \delta_{10}$

Correction

Correct answer

$\delta_w = h \times d$

30 November 2018
Stefano Ambrogio, IBM Research - Almaden
Weight update

528 input neurons → 250 hidden neurons → 125 hidden neurons → 10 output neurons

Cropped MNIST images

\[ \Delta W = \eta \times \delta \]

Correct answer

Learning rate

Neuron output correction

NN results

Correction

1 → \( y_1 \) - \( g_1 \) → \( \delta_1 \)
2 → \( y_2 \) - \( g_2 \) → \( \delta_2 \)
3 → \( y_3 \) - \( g_3 \) → \( \delta_3 \)
... → ... → ... → ... → ...
8 → \( y_8 \) - \( g_8 \) → \( \delta_8 \)
9 → \( y_9 \) - \( g_9 \) → \( \delta_9 \)
0 → \( y_{10} \) - \( g_{10} \) → \( \delta_{10} \)

Weight update
Hardware opportunities

Input data (images, raw speech data, etc.)
input to neural network

“MNIST” database
~1998
→ check-reading ATMs

Forward inference:

Hardware opportunity: Efficient, low-power deployment

Training:

Problem: It can take WEEKS to train these networks, even with many GPUs.

Hardware opportunity: Train big networks FASTER and at LOWER POWER.
Computation needed: “Multiply-accumulate”

With a GPU, matrix-multiplication is fast & parallel...

\[ y_j = f(\sum x_i w_{ij}) \]

... but \( x \) and \( w \) values must arrive from DRAM, and new \( y \) values sent back to DRAM.
AI hardware, present & near-future: high-level view

**Forward Inference**
(in the cloud & at the edge)

- TODAY
  - CPUs & GPUs
- VERY SOON
  - TPU1
  - Custom digital accelerators
- LATER ON…?
  - Analog-memory-based accelerators

**Training**
(mostly in the cloud)

- TODAY
  - CPUs & GPUs
- VERY SOON
  - TPU2
- LATER ON…?
  - ?
Outline

- Introduction
- A brain-inspired algorithm: Spike Timing Dependent Plasticity
- A machine learning algorithm: Back-Propagation
- Analog memory for training Neural Networks
  - Software-equivalent accuracy with novel unit cell
  - Circuit design considerations
- Conclusion
Emerging devices for memory and computing

- Resistive Memory (RRAM)
- Phase-Change Memory (PCM)
- Magnetic Memory (MRAM)
- Ferro-Electric Memory (FeRAM)

Information encoded in the device conductance

**NVM** (Non-Volatile Memory): usually for storing digital data (0s and 1s)

NVM technologies include:
- **MRAM** (Magnetic RAM)
- **PCM** (Phase-Change Memory)
- **RRAM** (Resistance RAM)

Like conventional memory (SRAM/DRAM/Flash), an NVM is addressed one row at a time, to retrieve previously-stored digital data.

- Analog resistors
- Selector device
- Analog amplifiers (analog current $\rightarrow$ 0s and 1s)
- Sense-Amplifiers

\[ V_{\text{read}} \]

Address decoder

0 1 0 1 0 1
Multiply-accumulate with NVM: computed at the data, by physics...

1) Different peripheral circuitry
2) Weights $w \rightarrow$ conductances $G^+, G^-$
   (Ohm’s Law: $V = IR \rightarrow I = GV$)
3) Apply “$x$” voltages to every row
   (Kirchhoff’s Current Law $\rightarrow \Sigma I$)
4) Analog measurement
Vision: NVM-based Deep Learning Chip

- Support multiple deep learning algorithms
- Reconfigurable routing: Map different neural net topologies to the same chip
- Weight override mechanism for distributed learning
Maximizing the future business case (vs. a GPU)

<table>
<thead>
<tr>
<th>Low Power</th>
<th>Accuracy</th>
<th>Faster</th>
</tr>
</thead>
<tbody>
<tr>
<td>(inherent in the physics, but possible to lose in the engineering...)</td>
<td>(essential that final Deep-NN performance be indistinguishable from GPUs – hardest technical challenge)</td>
<td>Of zero interest</td>
</tr>
<tr>
<td>Of zero interest</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Still of interest for power-constrained situations: learning-in-cars, etc.</td>
<td>Of zero interest</td>
<td>(circuitry must be massively parallel)</td>
</tr>
<tr>
<td>Sweet spot: rather than buy GPUs, people buy this chip instead for training of Deep-NN’s</td>
<td>Of zero interest</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Still of interest for some situations: learning-in-server-room</td>
<td></td>
</tr>
</tbody>
</table>

Stefano Ambrogio, IBM Research - Almaden
Outline

- Introduction
- A brain-inspired algorithm: **Spike Timing Dependent Plasticity**
- A machine learning algorithm: **Back-Propagation**
- Analog memory for training Neural Networks
- Software-equivalent accuracy with novel unit cell
  - Circuit design considerations
- Conclusion
Our journey towards high DNN accuracy

Where we were in June 2014

- Experiments on MNIST Dataset
- **82% accuracy** w/ 5,000 examples,
- Too slow for 60,000 examples

“What a GPU would get” with this network…

- **97-98% TEST accuracy** w/ 60,000 examples
- **94% TEST accuracy** w/ 5,000 examples

Non-idealities in Real PCM Devices

Study: 2-PCM: Asymmetric Conductance Response

- 2-PCM unit cell is non-linear and asymmetric
- Symmetry is crucial to balance UP and DOWN steps and accurately implement open-loop weight update
- Strong impact on Neural Network training accuracy

\[ W = G^+ - G^- \]

\[ G^+ + G^- = W \]

Non-linear behavior

Median

Number of pulses

Asymmetric update

Number of pulses

G^+ \: [\mu S]

G^- \: [\mu S]
2-PCM scheme: dependence on applied pulses

- $\Sigma \Delta W$ distributions are overlapped, preventing a clear distinction of increase and decrease weight requests
- MNIST accuracy is lower than accuracy achieved with TensorFlow on a same size network

MNIST Accuracy
TensorFlow: 97.94%
2-PCM: 93.77%
Novel 2T2R + 3T1C unit cell

\[ W = F \times (G^+ - G^-) + g^+ - g^- \]

- **Symmetry** → Weight update performed on \( g^+ \) only
  - \( g^- \) shared among many columns (e.g. 128 columns)
- **Dynamic Range** → Gain factor \( F \) (e.g. \( F = 3 \))
- **Non-Volatility** → Weight transferred to PCMs infrequently (every 1000s of images)

Novel unit cell: 2T2R + 3T1C, nominal behavior

- PMOS charges the capacitor, increasing $g^+$ and $W$
- NMOS discharges the capacitor, decreasing $g^+$ and $W$
- Read MOS shows a linear dependence of $g$ on $V_C$
- PMOS and NMOS provide the same current, balancing UP and DOWN weight updates

2T2R+3T1C scheme: dependence on applied pulses

- Higher number of requested pulses due to very small $g^+$ update
- MNIST accuracy is equivalent to accuracy achieved with TensorFlow on a same size network

MNIST Accuracy
TensorFlow: 97.94%
2T2R+3T1C: 98.10%
Novel unit cell: 2T2R + 3T1C, CMOS variability

- PMOS charges the capacitor, increasing $g^+$ and $W$
- NMOS discharges the capacitor, decreasing $g^+$ and $W$
- Read MOS shows a linear dependence of $g$ on $V_C$
- PMOS and NMOS never provide the same current, causing UP and DOWN weight updates asymmetry

2T2R+3T1C scheme: impact of CMOS variability

- Asymmetry in PMOS and NMOS strongly broadens $\Sigma \Delta W$ distributions
- MNIST accuracy is highly degraded with respect to accuracy achieved with TensorFlow

MNIST Accuracy

TensorFlow: 97.94%
2T2R+3T1C: 98.10%
+Variability: 92.42%
2T2R+3T1C scheme: polarity inversion

Polarity inversion: Invert the sign of the lower significance conductance between transfers to higher significance pair

\[ W = F \times (G^+ - G^-) + g^+ - g^- \]

\[ W = F \times (G^+ - G^-) - (g^+ - g^-) \]

Stronger PFET
Equal PFET and NFET
Stronger NFET
+100 net pulses

W = \( F \times (G^+ - G^-) + g^+ - g^- \)

Increase weight
DECREASE weight
INCREASE weight

W = \( F \times (G^+ - G^-) - (g^+ - g^-) \)

Read current ADDs to weight
Read current SUBTRACTS from weight

Transfer
2T2R+3T1C scheme: CMOS variability, polarity inversion

- Asymmetry in PMOS and NMOS is averaged by polarity inversion
- MNIST accuracy is equivalent to accuracy achieved with TensorFlow

MNIST Accuracy
Tensorflow: 97.94%
Polarity Inv: 97.95%

Dynamic Range Fraction

-100 net pulses
+100 net pulses

PDF

-50 -40 -30 -20 -10 0 10 20 30 40 50

ΣΔW [μS]
Accuracy on MNIST and MNIST backrand

Mixed hardware-software experiment: every synaptic weight $\rightarrow$ 2 real PCM devices

Transfer learning from ImageNet to CIFAR-10/100

Mixed hardware-software experiment

Transfer Learning: Use pre-trained, scaled weights from ImageNet for convolution layers

Only train last fully-connected layer

Full 2-Analog Memory structure

\[ W = F \times (G^+ - G^-) + g^+ - g^- \]

- Single pair of devices performing the entire training
Single device requirements

- Several specifications are requested to single resistive device in order to obtain software-equivalent accuracies

- A minimum of 1000 different conductance steps are required → extremely hard to obtain

- A maximum 5% of asymmetry between up and down conductance updates → need for very linear and symmetric devices

Our solution → Multiple conductances of varying significance, diversification of requirements

<table>
<thead>
<tr>
<th>Spec</th>
<th>Parameter</th>
<th>Value</th>
<th>Tolerance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pulse duration</td>
<td>$\pm V_S$</td>
<td>1 ns</td>
<td></td>
</tr>
<tr>
<td>Operating voltage</td>
<td>$\pm V_S$</td>
<td>1 V</td>
<td></td>
</tr>
<tr>
<td>Maximum device area</td>
<td>$R_{device}$</td>
<td>0.04 $\mu$m$^2$</td>
<td>24 M$\Omega$</td>
</tr>
<tr>
<td>Average device resistance</td>
<td>$R_{device}$</td>
<td>24 M$\Omega$</td>
<td>7 M$\Omega$</td>
</tr>
<tr>
<td>Maximum device resistance</td>
<td>$\max (g_{ij})$</td>
<td>112 M$\Omega$</td>
<td>7 M$\Omega$</td>
</tr>
<tr>
<td>Minimum device resistance</td>
<td>$\min (g_{ij})$</td>
<td>14 M$\Omega$</td>
<td>7 M$\Omega$</td>
</tr>
<tr>
<td>Resistance on/off ratio</td>
<td>$\max (g_{ij})/\min (g_{ij})$</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Resistance change at $\pm V_S$</td>
<td>$\Delta g^\pm_{min}$</td>
<td>100 K$\Omega$</td>
<td>30 K$\Omega$</td>
</tr>
<tr>
<td>Resistance change at $\pm V_S/2$</td>
<td>$\Delta g^\pm_{min}$</td>
<td>10 K$\Omega$</td>
<td></td>
</tr>
<tr>
<td>Storage capacity</td>
<td>$(\max (g_{ij}) - \min (g_{ij})) / \Delta g_{min}$</td>
<td>1000 levels</td>
<td></td>
</tr>
<tr>
<td>Device up/down asymmetry</td>
<td>$\Delta g^+<em>{min} / \Delta g^-</em>{min}$</td>
<td>1.05</td>
<td>2%</td>
</tr>
</tbody>
</table>

Note that these numbers are derived from the radar diagram in Figure 4A and correspond to the shaded area. "Global asymmetry in up/down responses can be to a large extend compensated by proper adjustment of pulse widths and/or pulse amplitude.

Full 4-Analog Memory structure

\[ W = F \times (G^+ - G^-) + g^+ - g^- \]

- **Most Significant Pair**: Infrequent, **Closed Loop Programming** Operation
- **Least Significant Pair**: Frequent, **Open Loop Programming** Operation
Suggestions for new analog memory devices

- Larger unit cell with two components
  1. More-significant pair of non-volatile conductances (e.g., PCM) stores “most” of the weight info
    - Non-linear conductance update $\rightarrow$ OK
    - DOES need to be able to tune these conductances rapidly in a CLOSED-LOOP manner
  2. We perform all the OPEN-LOOP programming using a “less-significant” pair of conductances
    - Poor retention $\rightarrow$ OK
    - Significant device-to-device fixed variabilities $\rightarrow$ OK
    - DOES need to offer highly linear conductance update

$\rightarrow$ Reduces the difficulty of device requirements

Comparison of device specifications for MSP and LSP

<table>
<thead>
<tr>
<th>Specifications</th>
<th>Parameter</th>
<th>MSP</th>
<th>LSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Step-size</td>
<td>$\Delta G_0$</td>
<td>$&lt; 21 \ \mu S$ (42%)</td>
<td>$&lt; 1.4 \ \mu S$ (2.8%)</td>
</tr>
<tr>
<td>Intra-device Variability</td>
<td>$\sigma_{\text{intra}}$</td>
<td>$&lt; 1.5 \ \mu S$</td>
<td>$&lt; 0.8 \ \mu S$</td>
</tr>
<tr>
<td>Inter-device Variability</td>
<td>$\sigma_{\text{Gmax}}$</td>
<td>$&lt; 10 \ \mu S$</td>
<td>$&lt; 12 \ \mu S$</td>
</tr>
<tr>
<td></td>
<td>$\sigma_{G_0}^{*}$</td>
<td>$&lt; 200%$</td>
<td>$&lt; 95%$</td>
</tr>
<tr>
<td>Faulty devices</td>
<td>Dead C.R.</td>
<td>$&lt; 7%$</td>
<td>$&lt; 7%$</td>
</tr>
<tr>
<td></td>
<td>Stuck On C.R.</td>
<td>$&lt; 35%$</td>
<td>$&lt; 10%$</td>
</tr>
<tr>
<td>Dynamic range</td>
<td>Number of levels</td>
<td>$&gt; 13$</td>
<td>$&gt; 110$</td>
</tr>
<tr>
<td>Retention</td>
<td>Time before data loss</td>
<td>Higher</td>
<td>Lower</td>
</tr>
<tr>
<td>Endurance</td>
<td>Number of Set/Reset</td>
<td>Lower</td>
<td>Higher</td>
</tr>
</tbody>
</table>

Perspective on Training Fully Connected Networks with Resistive Memories: Device Requirements for Multiple Conductances of Varying Significance

Giorgio Cristiano,1,2 Massimo Giordano,1,2 Stefano Ambrogio,3 Louis P. Romero,1 Christina Cheng,1 Pritish Narayanan,1 Hsinyu Tsai,1 Robert M. Shelby,3 and Geoffrey W. Burr1,4

1IBM Research AI, IBM Research-Almaden, 650 Harry Road, San Jose, CA USA 95120
2EPFL, Route Cantonale, 1015 Lausanne, Switzerland

Outline

- Introduction
- A brain-inspired algorithm: **Spike Timing Dependent Plasticity**
- A machine learning algorithm: **Back-Propagation**
- Analog memory for training Neural Networks
- Software-equivalent accuracy with novel unit cell
- Circuit design considerations
- Conclusion
**Long-term: maximizing the future business case (vs. a GPU)**

<table>
<thead>
<tr>
<th><strong>Low Power</strong></th>
<th><strong>Accuracy</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>(inherent in the physics, but possible to lose in the engineering…)</td>
<td>(essential that final Deep-NN performance be indistinguishable from GPUs – hardest technical challenge)</td>
</tr>
<tr>
<td><strong>Sweet spot:</strong> rather than buy GPUs, people buy this chip instead for training of Deep-NN’s</td>
<td>Still of interest for some situations: learning-in-server-room</td>
</tr>
<tr>
<td><strong>Faster</strong></td>
<td>(circuitry must be massively parallel)</td>
</tr>
</tbody>
</table>

Still of interest for power-constrained situations: learning-in-cars, etc.
Suggestions from circuit design work

1) Parallelism is key

2) Avoiding ADC (Analog-to-Digital Conversion) saves time, power and area

3) Do the necessary computations (squashing functions) but be as “approximate” as you can (get away with)

4) Need to get vectors of data from the bottom of one array to the edge of the next one

5) Digital accelerators are at their best w/ **convolutional** layers; Analog-memory accelerators are at their best w/ **fully-connected** layers.
Impact on Convolutional Neural Networks

- Only the last layers in a Convolutional Neural Network are Fully Connected due to memory constraints
- Hardware accelerators could easily implement FC layers, what could be the impact on CNN topology and performance?

https://devblogs.nvidia.com/parallelforall/deep-learning-nutshell-core-concepts/
Outline

- Introduction
- A brain-inspired algorithm: **Spike Timing Dependent Plasticity**
- A machine learning algorithm: **Back-Propagation**
- Analog memory for training Neural Networks
- Software-equivalent accuracy with novel unit cell
- Circuit design considerations
- Conclusion
Conclusion

- AI is introducing novel tools to develop solutions to everyday challenges
  - Brain Inspired approach
  - Deep Learning approach

- NVM-based crossbar arrays can accelerate the training of Deep Machine Learning compared to GPU-based training
  - Multiply-accumulate performed at the data
  - Possible 500x speedup and orders-of-magnitude lower power

- Experimental results on a 2T2R+3T1C unit cell demonstrate software-equivalent training accuracy
  - MNIST, MNIST-backrand, CIFAR-10 and CIFAR-100 tested

- Need area-efficient peripheral circuitry
  - Tradeoffs balancing simplicity and area-efficiency against impact on ANN performance

stefano.ambrogio@ibm.com