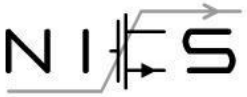
Department of Electronic Engineering, Tsinghua University

# Training Itself: Mixed-signal Training Acceleration for Memristor-based Neural Network

**Boxun Li**[1], Yuzhi Wang[1], Yu Wang[1], Yiran Chen[2], Huazhong Yang[1]

[1] Dept. of E.E., Tsinghua University, Beijing, China

[2] Dept. of E.C.E., University of Pittsburgh, Pittsburgh, USA



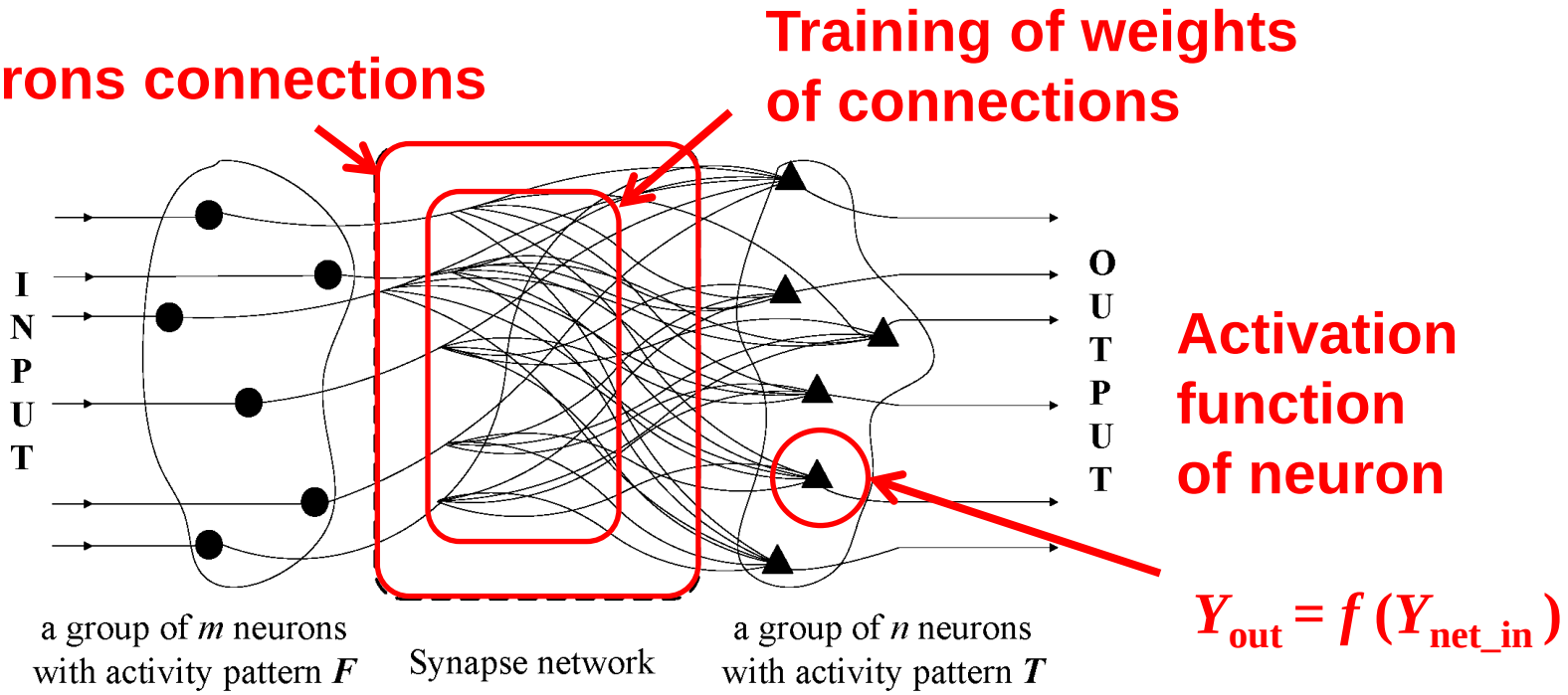
Nano-scale Integrated Circuit and System Lab.

# Outline

- Motivation
- Background Knowledge
- Proposed Framework
  - Modified Training Algorithm
  - Mixed-Signal  Training Architecture
- A Case Study: MNIST
- Future Work

# Motivation

- Artificial Neural Networks?
  - An information-processing system that has certain performance characteristics in common with biological neural network.
  - Characterized with three properties.

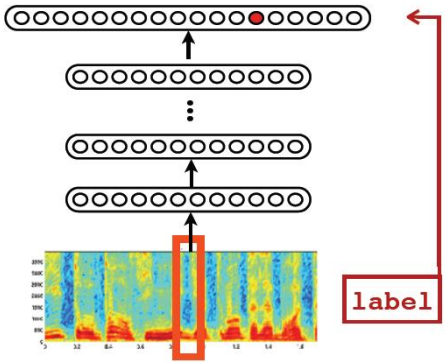



Miao Hu et al. , "Hardware realization of BSB recall function using memristor crossbar arrays" DAC 2012

# Motivation

- Artificial Neural Networks (ANNs)
  - Widely employed in data processing applications
  - State-of-the-art performance

Acoustic Modeling for Speech Recognition



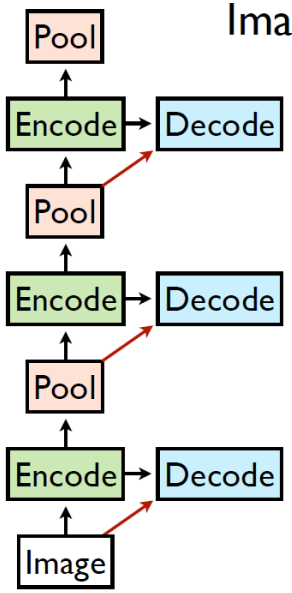

Close collaboration with Google Speech team

Trained in <5 days on cluster of 800 machines

30% reduction in Word Error Rate
("equivalent to 20 years of speech research")

Deployed in Jellybean release of Android

Semi-supervised Feature Learning in Images



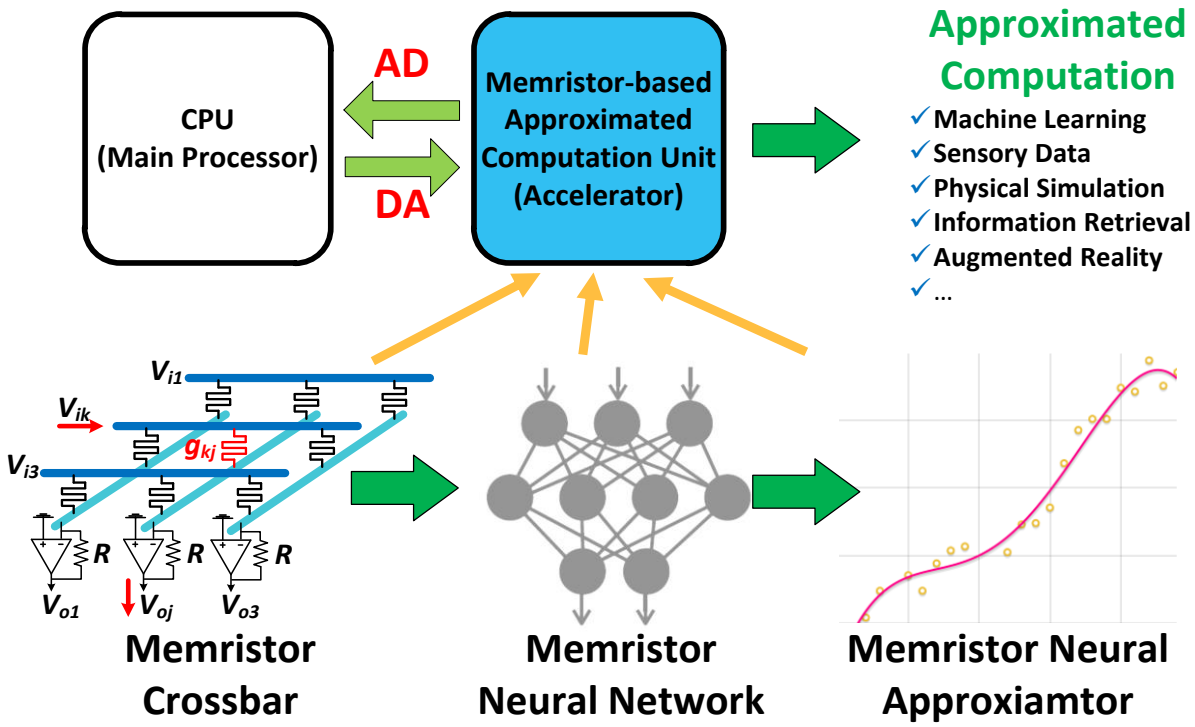

ImageNet Classification Results:

ImageNet 2011 (20k categories)
- Chance: 0.005%
- Best reported: 9.5%
- Our network: 16% (+70% relative)

**Jeff Dean, "Large-Scale Data and Computation: Challenges and Opportunities", January 19, 13**

# Motivation

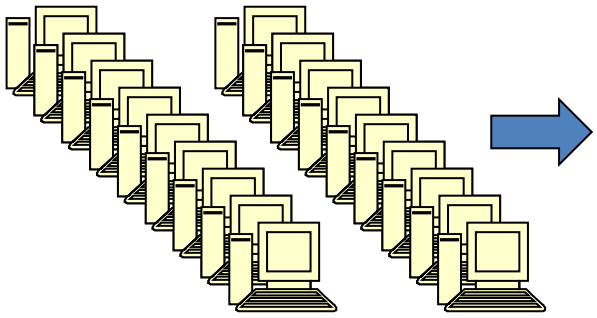
- Combine ANNs and emerging devices
  - Promising solutions to ultra-high power efficient computing
  - Memristor-based approximated computation: **> 400GFlops/W**



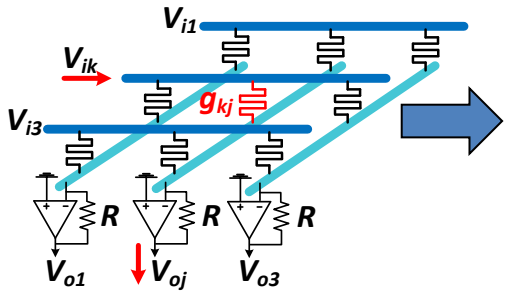

**Boxun Li et al. , "Memristor-based Approximated Computation", ISLPED 2013**

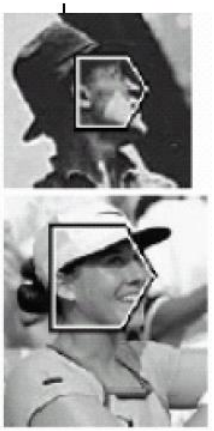
# Motivation

- ANNs must get trained for efficient data processing!
- Training Phase:
  - Critical operation to obtain an ANN for a specific task.
  - **Time consuming and resource intensive**
  - **Confined to digital (CPU/GPU/FPGA) systems**



| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 0.5377 | 0.8404 | 0.1832 |
| 2 | 1.8339 | -0.8880 | -1.0298 |
| 3 | -2.2588 | 0.1001 | 0.9492 |
| 4 | 0.8622 | -0.5445 | 0.3071 |
| 5 | 0.3188 | 0.3035 | 0.1352 |
| 6 | -1.3077 | -0.6003 | 0.5152 |
| 7 | -0.4336 | 0.4900 | 0.2614 |
| 8 | 0.3426 | 0.7394 | -0.9415 |
| 9 | 3.5784 | 1.7119 | -0.1623 |
| 10 | 2.7694 | -0.1941 | -0.1461 |
| 11 | -1.3499 | -2.1384 | -0.5320 |
| 12 | 3.0349 | -0.8396 | 1.6821 |



**Training on digital systems**

**Network's parameters**

**Tuning memristors to specific states**

**Data Processing**

**Digital Training**

**Analog Processing**

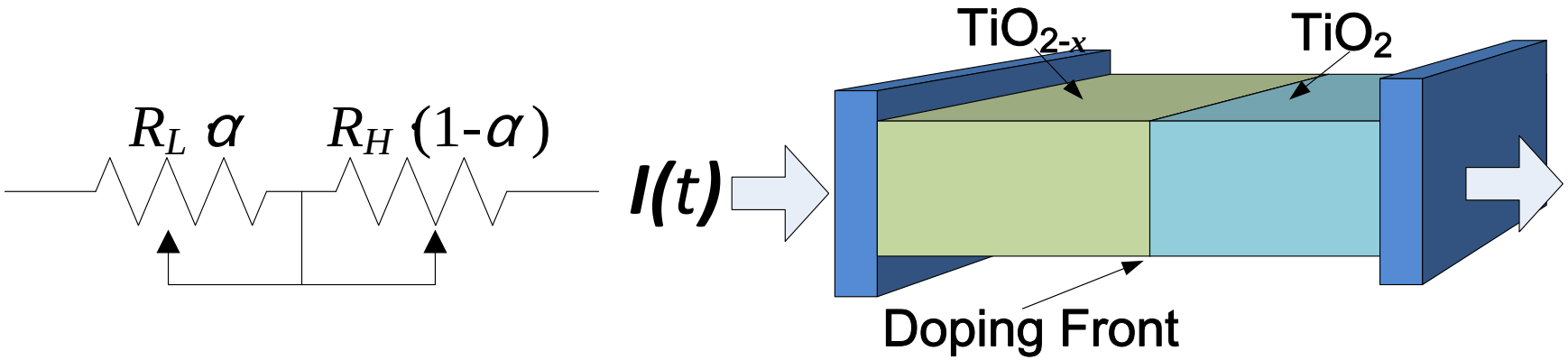
# Training Itself

- Realize the self-training of memristor-based neural network through mixed-signal system
  - Analog unit:
    - ✓ work out the training calculations efficiently
    - ✓ directly configure the memristor **without** state tuning
  - Digital unit:
    - ✓ only help control the state of training
    - ✓ **NOT** performing a large amount of complicate calculations.
  - Performance and energy gains

**How to let the memristor train itself?**

# Background: Memristor

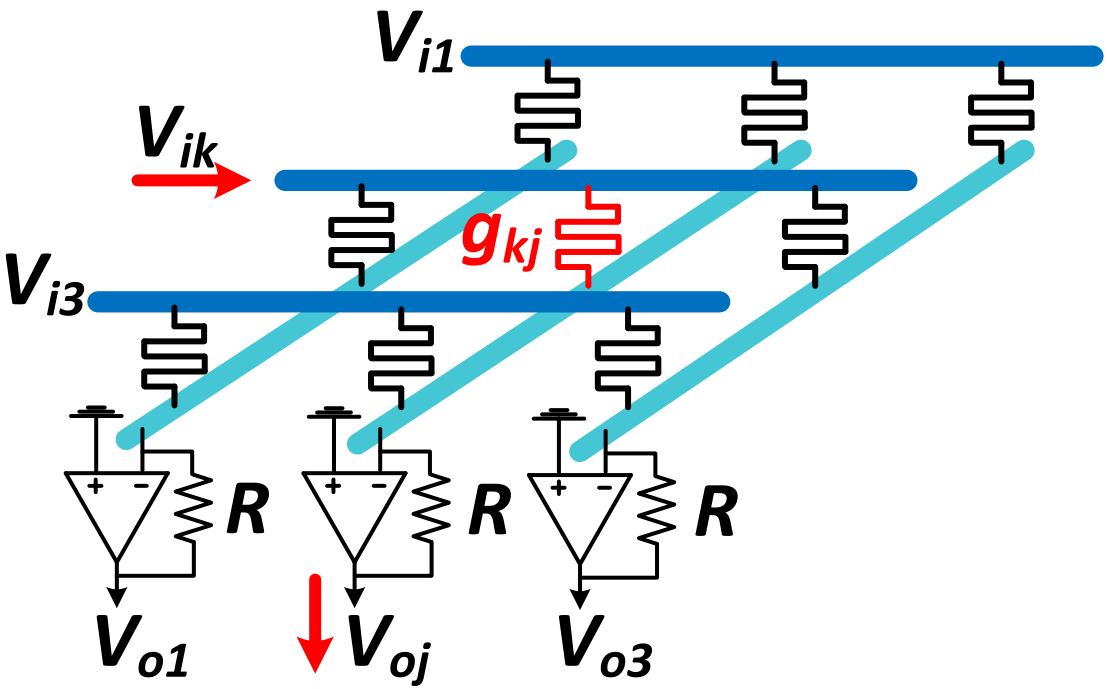
- Memristor
  - First physically realized by HP Labs in 2008 [Dmitri Nature 2008]
  - Ultra-integration density
  - The state of the memristor could be tuned by the current passing through itself
  - In this paper, we mainly take advantage of the **variable resistance states** of the memristor.



# Background: Memristor Crossbar

- Memristor Crossbar Array [Miao DAC2012]
  - Realize matrix-vector multiplication operation efficiently

$$V_{oj} = r \cdot \sum_k (V_{ik} \cdot g_{kj}),\ g_{kj} = \frac{1}{M_{kj}}$$

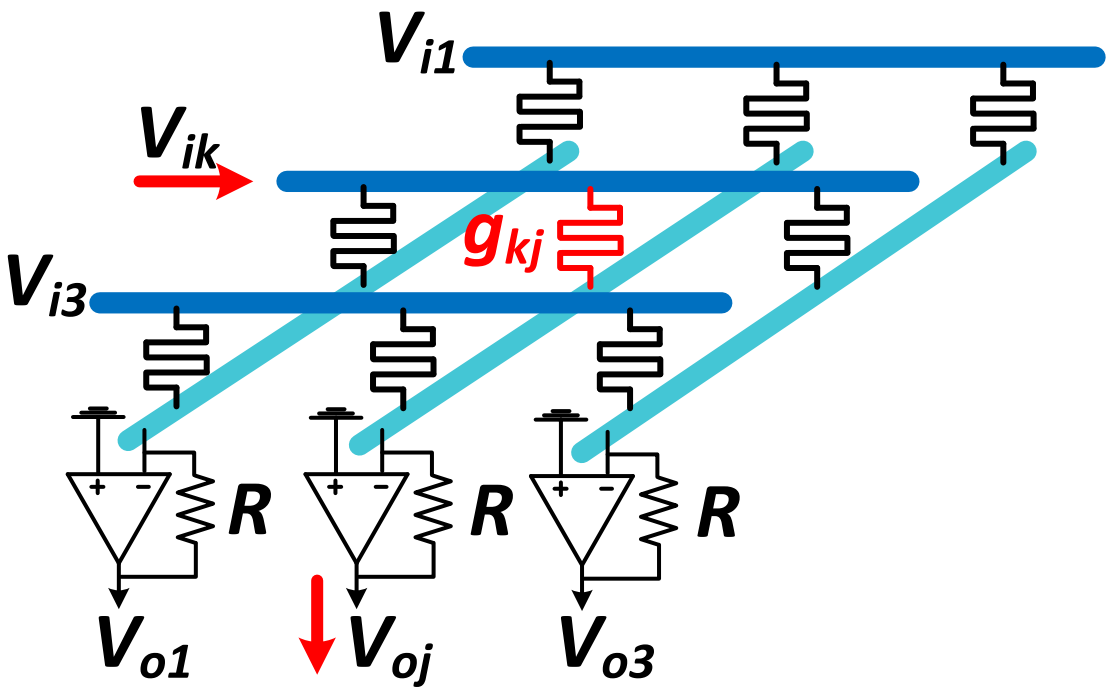


# Background: Memristor Crossbar

- Memristor Crossbar Array [Miao DAC2012]
  - Realize matrix-vector multiplication operation efficiently

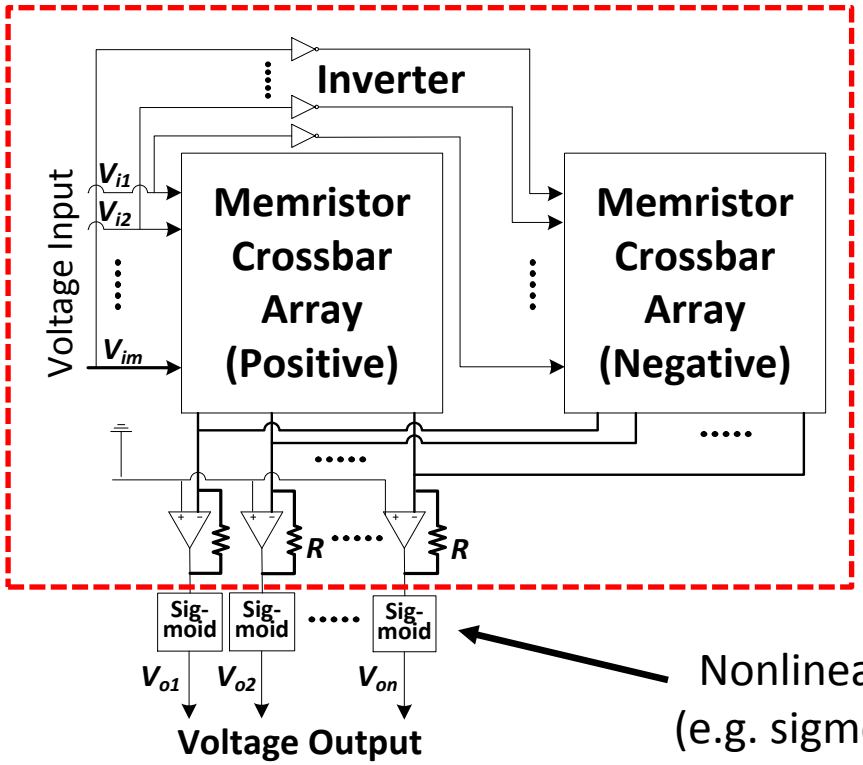
Basic Operations of ANNs

$$V_{oj} = r \cdot \sum_k (V_{ik} \cdot g_{kj}),\ g_{kj} = \frac{1}{M_{kj}}$$



# Background: Memristor-based Neural Net

- Two crossbar arrays & Sigmoid circuit [G. Khodabandehloo TVLSI2012]
  - Since both R and M can only be positive, two crossbars are needed to represent the positive and negative weights of a network.
  - A multilayer neural network can be realized by combining several such architectures together




$$V_{oj} = \sum_k (V_{ik} \cdot w_{kj}) \qquad g_{kj} = \frac{1}{M_{kj}}$$

$$w_{kj} = R \cdot (g_{kj(postive)} - g_{kj(negative)})$$

# Outline

- Motivation
- Background Knowledge
- Proposed Framework
  - Modified Training Algorithm
  - Mixed-Signal Training Architecture
- A Case Study: MNIST
- Future Work

# Original Training Algorithm

- Stochastic Gradient Descent (SGD)
  - One of the most common algorithms for neural network training
  - An **iterative** process and hard to be parallelized
  - The update of each weight ($w_{ji}$) between Node i (this layer) and Node j (next layer) is:

$$w_{ji} \leftarrow w_{ji} + \eta \cdot \delta_j \cdot x_i$$

Error

Input of Node i

Learning Rate

  - For the Node p in the output layer:

$$\delta_p = o_p \cdot (1 - o_p) \cdot (t_p - o_p)$$

Actual Output

Ideal Output

  - For the Node h in the hidden layers:

$$\delta_h = o_h \cdot (1 - o_h) \cdot \sum_{k \in nextlayer} w_{kh}\delta_k$$

Output of the Node

Calculations are difficult for analog systems to accomplish!

# Proposed Method

- Approximate calculations:
  - For the error calculations of Node p in the output layer:

$$\delta_p = o_p \cdot (1 - o_p) \cdot (t_p - o_p) \longrightarrow \delta_p = t_p - o_p$$

  - For the Node h in the hidden layers:

$$\delta_h = o_h \cdot (1 - o_h) \cdot \sum_{k \in nextlayer} w_{kh}\delta_k \longrightarrow \delta_h = filter(o_h) \cdot \sum_{k \in nextlayer} w_{kh}\delta_k$$

$$filter(o_h) = \begin{cases} 0, & \varepsilon < o_h < 1-\varepsilon \\ 1, & \text{others} \end{cases}$$

Small number (e.g. 0.1)

Output of sigmoid function $f(x) = \frac{1}{1+e^{-x}}$

$0 < o_h, o_p < 1$

  - The update of each weight ($w_{ji}$) between Node i (this layer) and Node j (next layer) is:

$$w_{ji} \leftarrow w_{ji} + \eta \cdot \delta_j \cdot x_i \longrightarrow w_{ji} \leftarrow w_{ji} + \eta \cdot sign(\delta_j) \cdot |\delta_j| \cdot x_i$$

**Still difficult for analog systems**

# Proposed Method

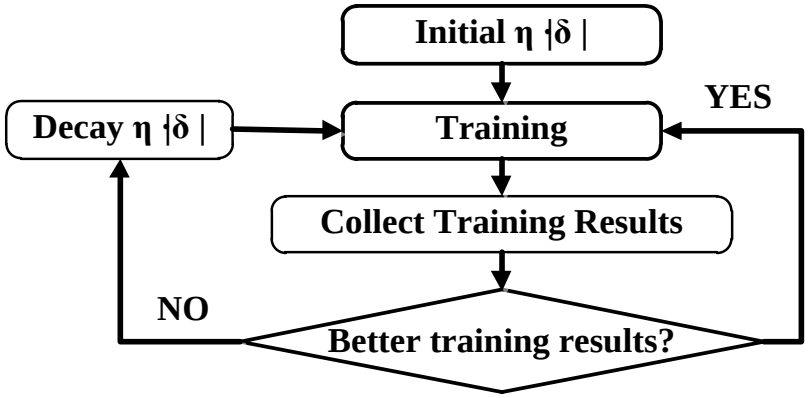
- Approximate calculations:
  - The update of each weight ($w_{ji}$) is still difficult to calculate

$$w_{ji} \leftarrow w_{ji} + \eta \cdot |\delta_j| \cdot sign(\delta_j) \cdot x_i$$

**Update Amplitude**
**Difficult to compute**

**Update Direction**
**Easy to calculate**

**Do NOT calculate**
**Estimation & Automatically Adjust**




$$w_{ji} \leftarrow w_{ji} + \gamma \cdot ConvergenceRate_j \cdot sign(\delta_j) \cdot x_i$$

**Random Number**

$\eta \cdot |\delta_j|$

**Automatic Convergence Rate**

# Supporting Architecture

- After approximation, the calculations can be accomplished with analog circuits.
- Approximate calculations:
  - For the error calculations of Node p in the output layer:

$$\delta_p = t_p - o_p$$ → Analog Subtractor

  - For the Node h in the hidden layers:

$$\delta_h = filter(o_h) \cdot \sum_{k \in nextlayer} w_{kh}\delta_k$$ → Copy Crossbar

$$filter(o_h) = \begin{cases} 0, & \varepsilon < o_h < 1-\varepsilon \\ 1, & \text{others} \end{cases}$$

  - The update of each weight ($w_{ji}$) between Node i (this layer) and Node j (next layer) is:

$$w_{ji} \leftarrow w_{ji} + \gamma \cdot ConvergenceRate_j \cdot sign(\delta_j) \cdot x_i$$

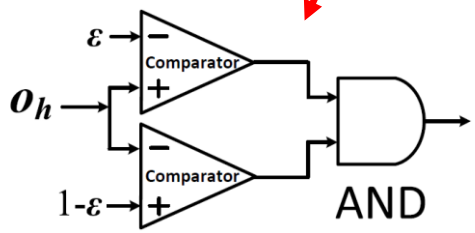



Fig. 3. Implementation of the Filter Operation

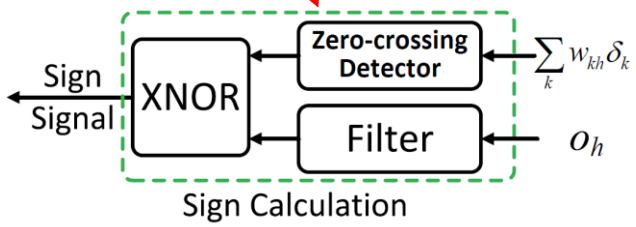
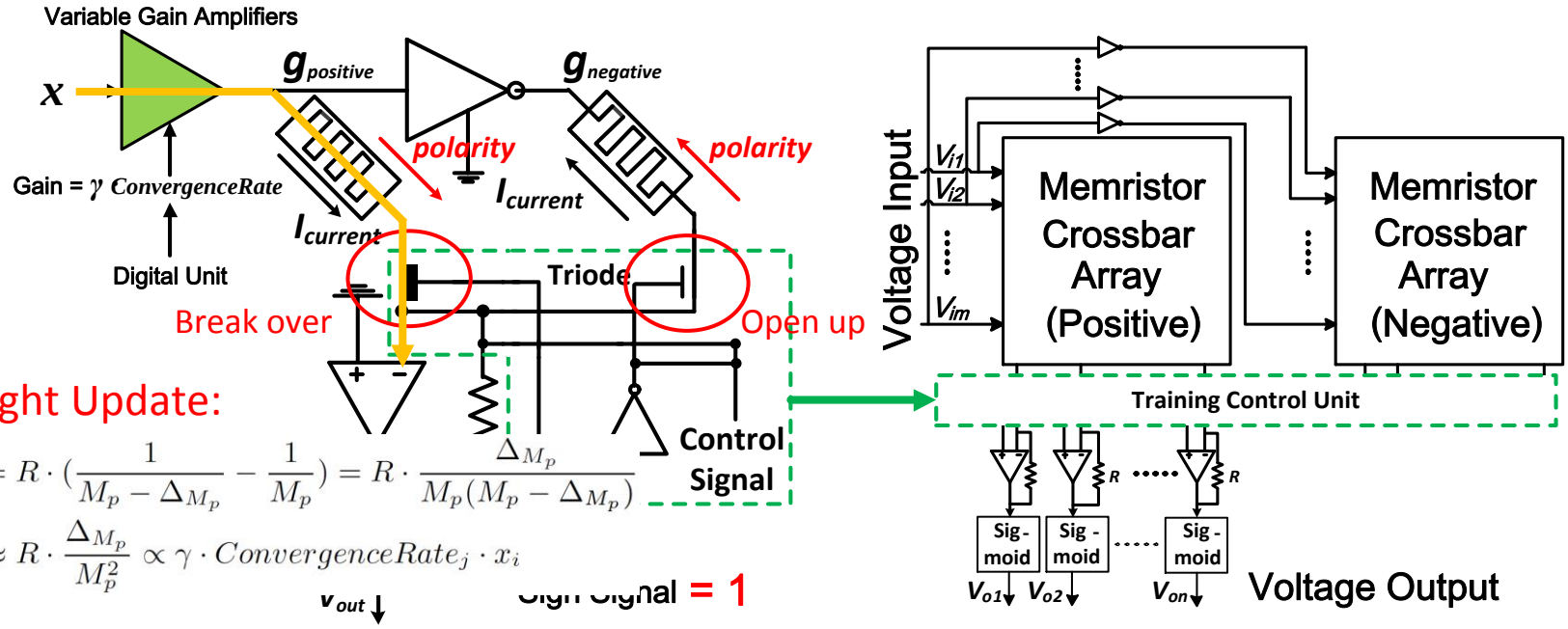



Fig. 4. Sign Calculation Unit

# Supporting Architecture

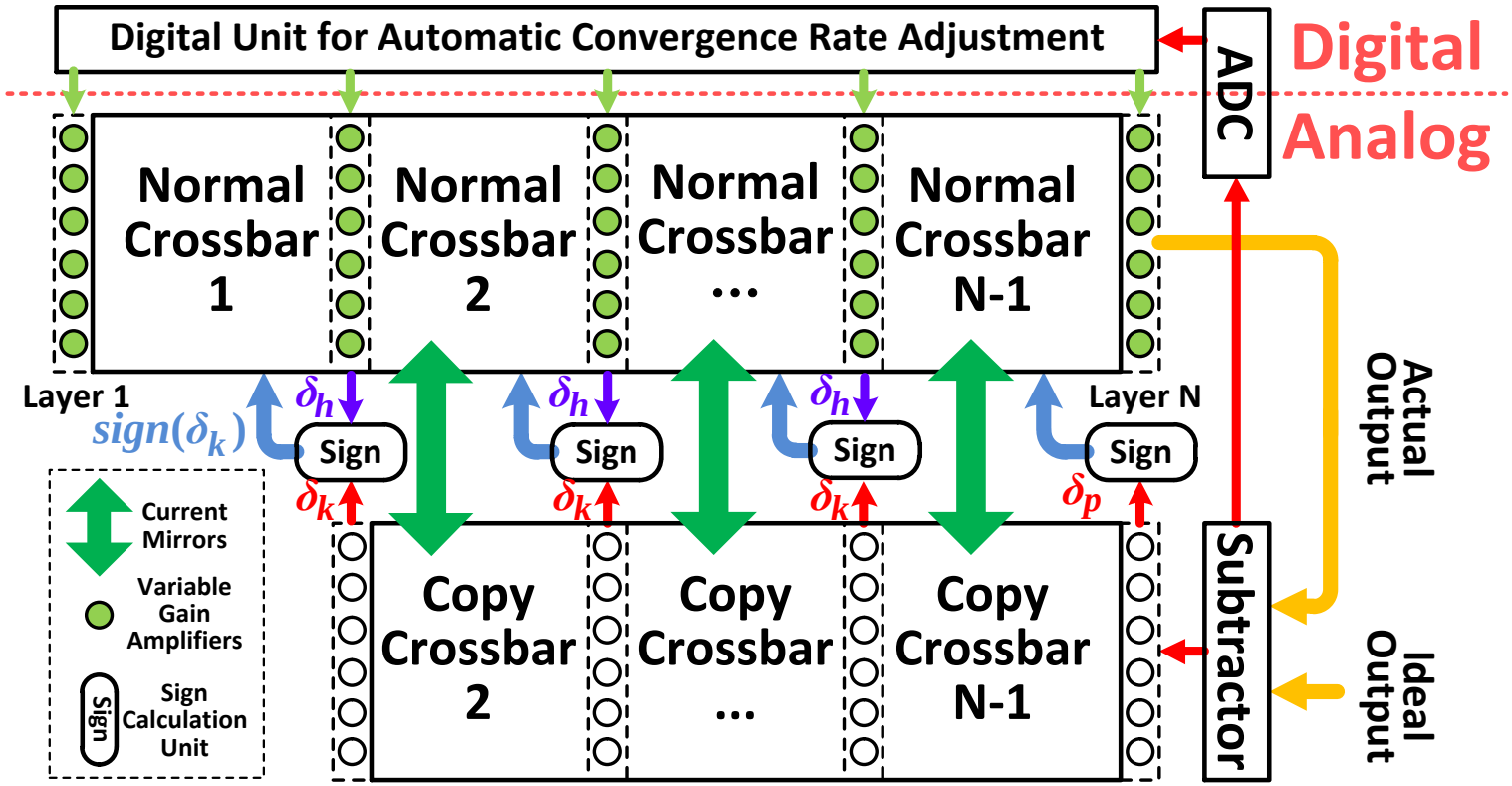
- After approximation, the calculations can be accomplished with analog circuits.
- Approximate calculations:
  - The update of each weight ($w_{ji}$) between Node i (this layer) and Node j (next layer) is:

$$w_{ji} \leftarrow w_{ji} + \gamma \cdot ConvergenceRate_j \cdot sign(\delta_j) \cdot x_i$$



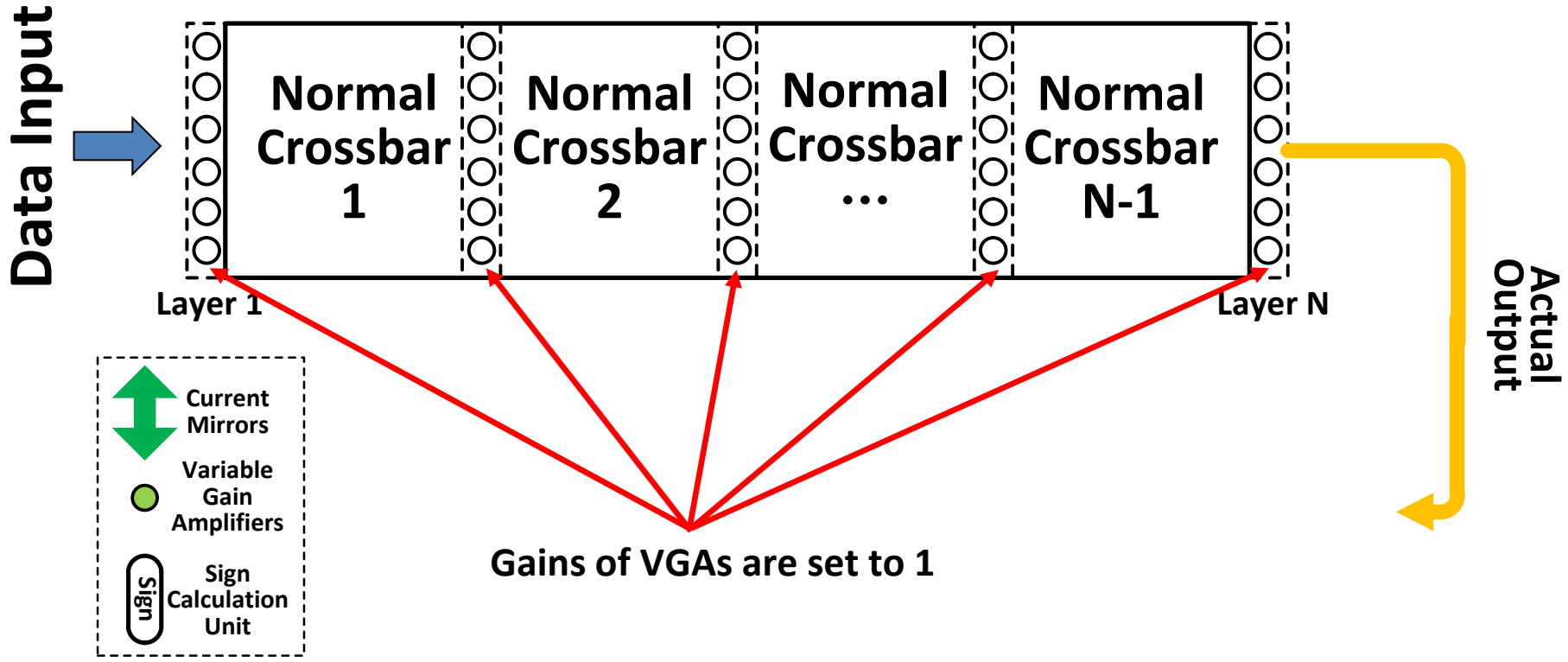
# Mixed-Signal Training Framework

- Mixed-Signal Training Framework
  - Normal Crossbar: feedforward calculations for actual output
  - Copy Crossbar: backpropagate calculations of errors for weight update
  - Digital Unit: monitor training process and automatically adjust convergence rate



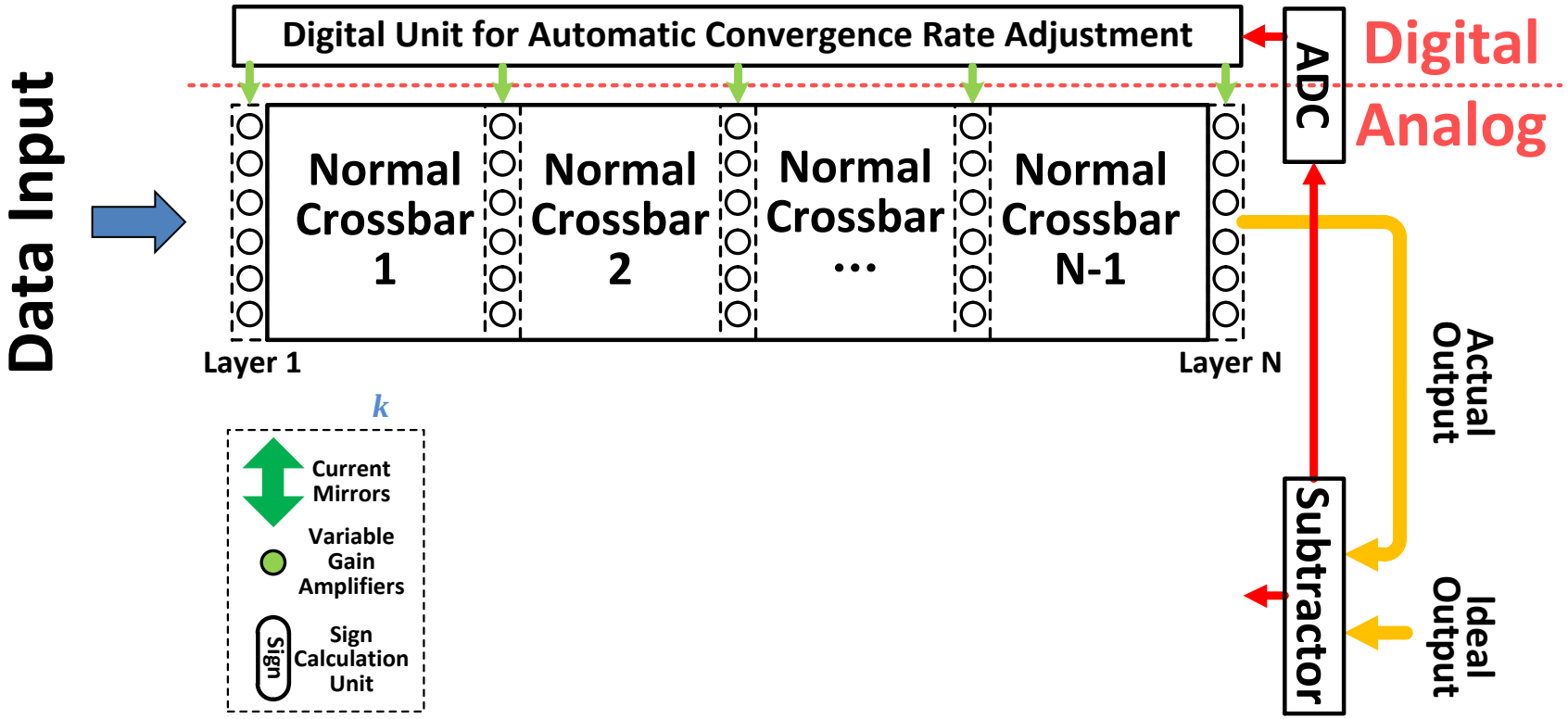
# Mixed-Signal Training Framework

- Mixed-Signal Training Framework
  - Calculate the actual output of the network
  - Calculate the error for weight update and automatically adjust convergence rate
  - Update weight (states of memristor in the crossbar)



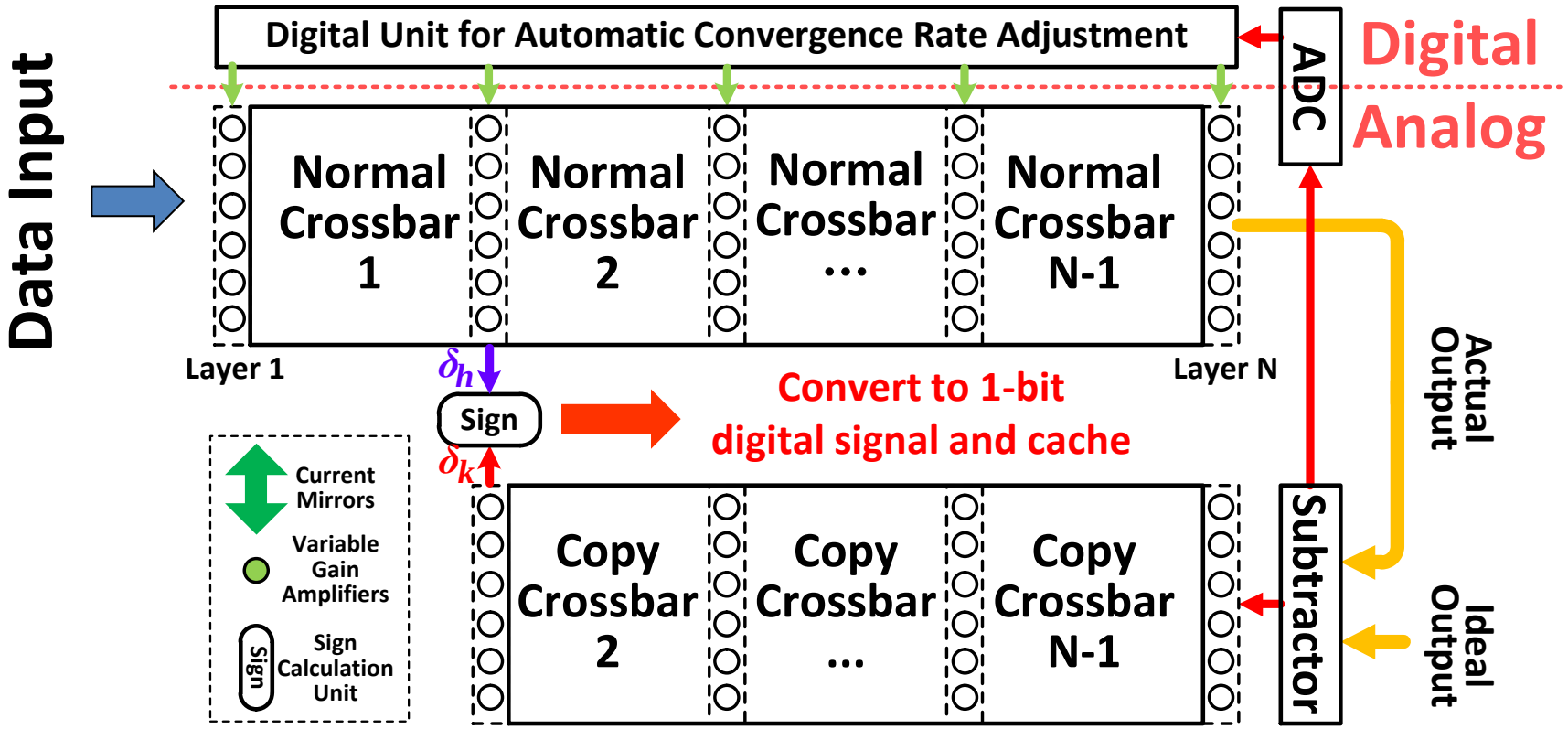
# Mixed-Signal Training Framework

- Mixed-Signal Training Framework
  - Calculate the actual output of the network
  - Calculate the error for weight update and automatically adjust convergence rate
  - Update weight (states of memristor in the crossbar)



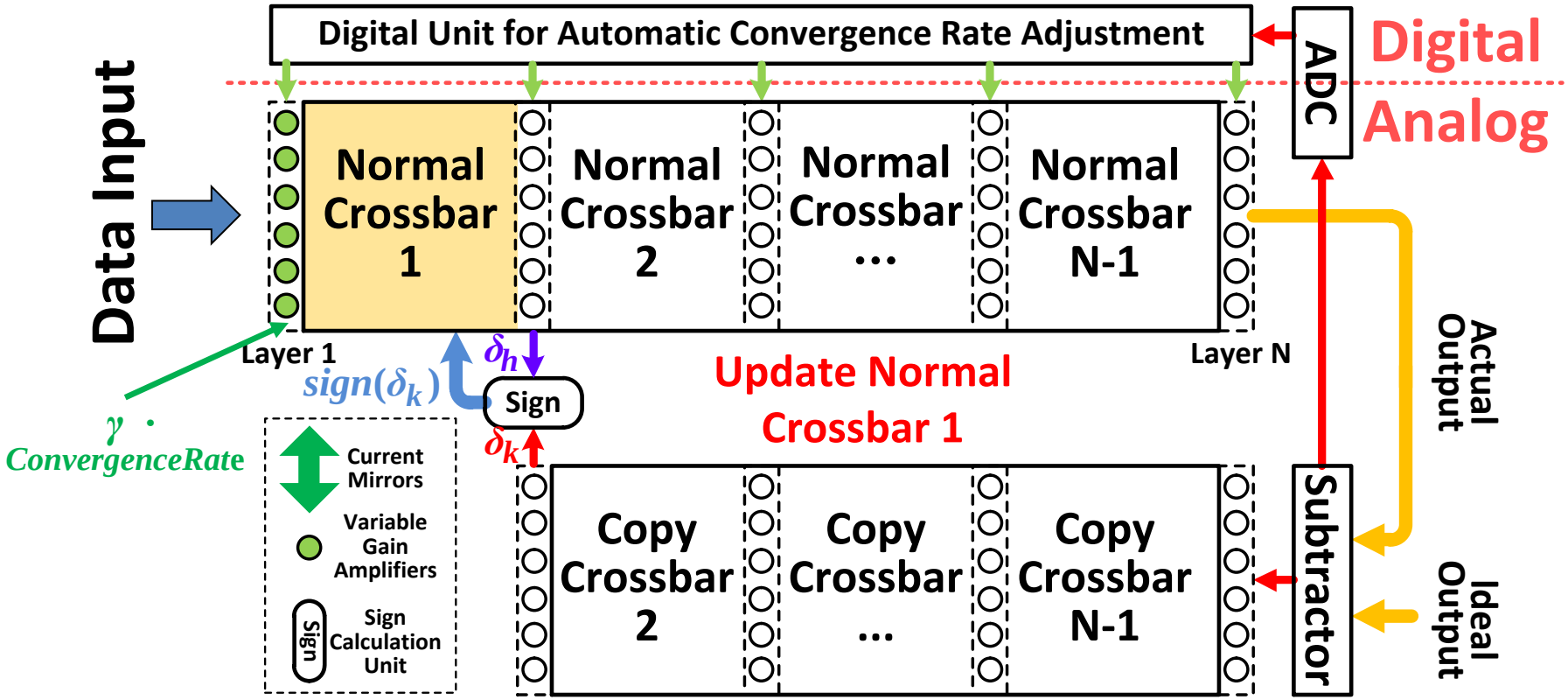
# Mixed-Signal Training Framework

- Mixed-Signal Training Framework
  - Calculate the actual output of the network
  - Calculate the error for weight update and automatically adjust convergence rate
  - Update weight (states of memristor in the crossbar)



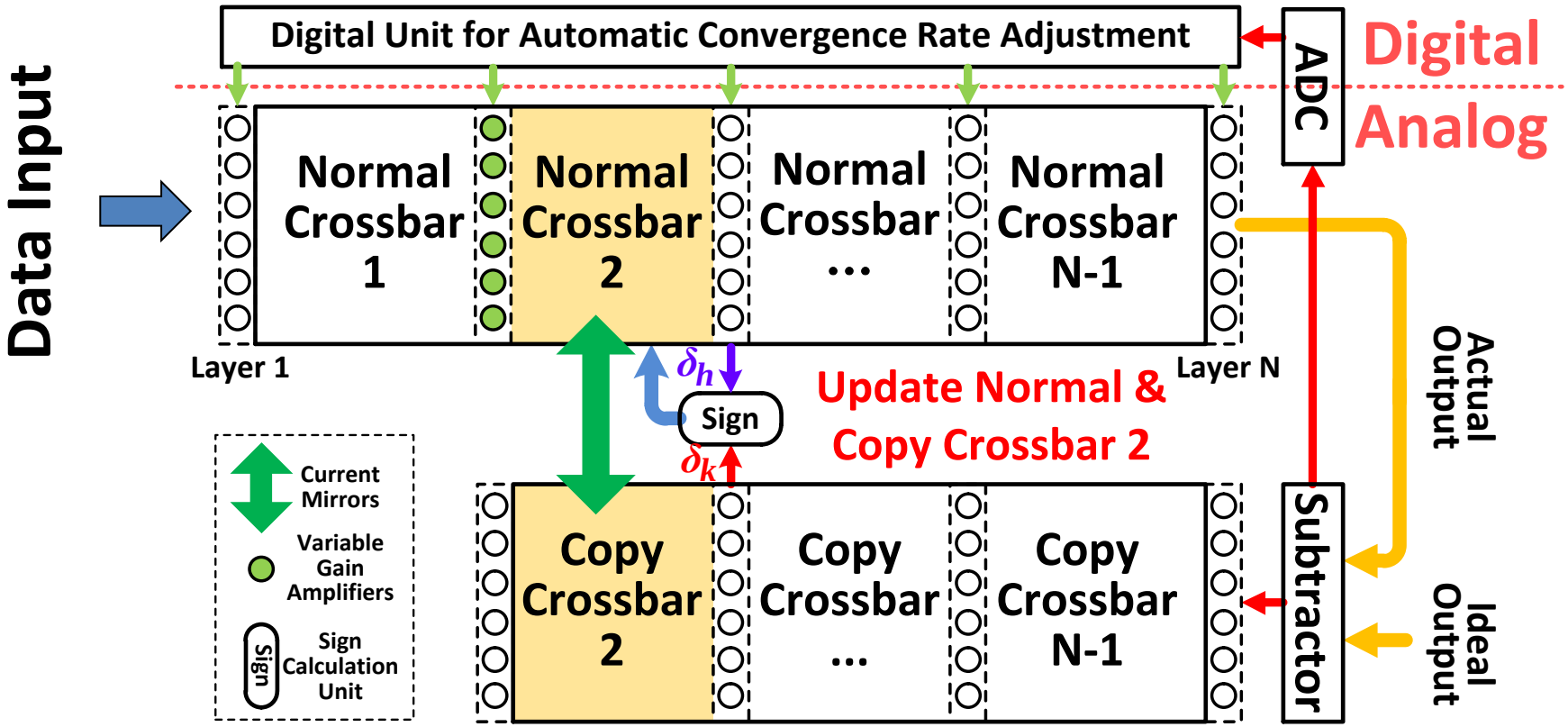
# Mixed-Signal Training Framework

- Mixed-Signal Training Framework
  - Calculate the actual output of the network
  - Calculate the error for weight update and automatically adjust convergence rate
  - Update weight (states of memristor in the crossbar)



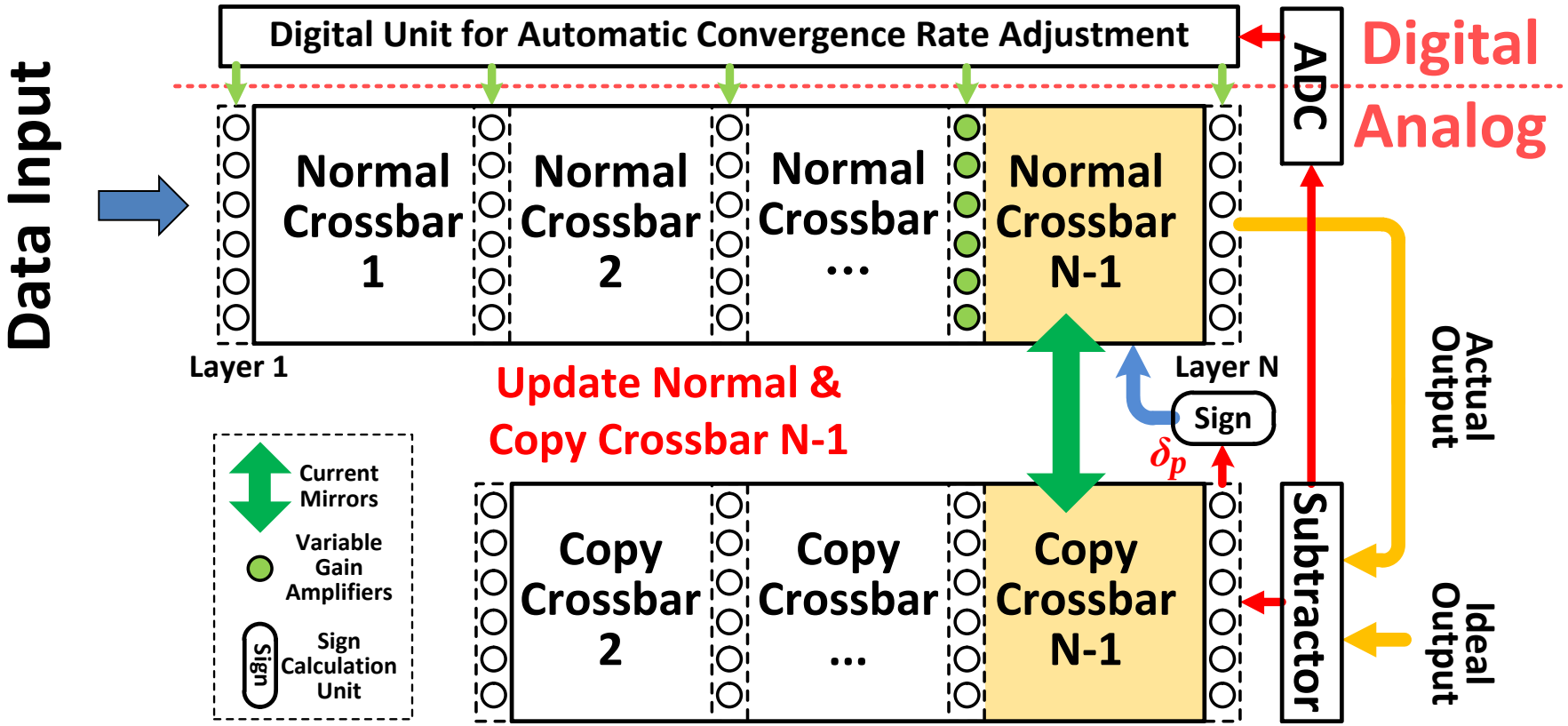
# Mixed-Signal Training Framework

- Mixed-Signal Training Framework
  - Calculate the actual output of the network
  - Calculate the error for weight update and automatically adjust convergence rate
  - Update weight (states of memristor in the crossbar)



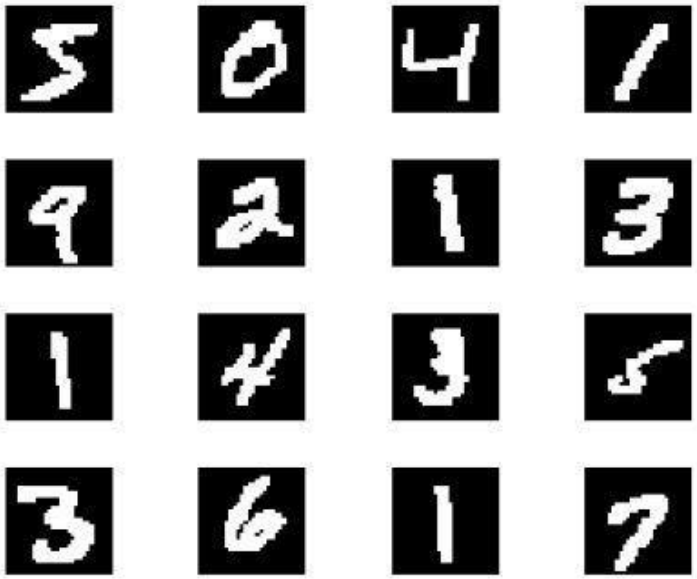
# Mixed-Signal Training Framework

- Mixed-Signal Training Framework
  - Calculate the actual output of the network
  - Calculate the error for weight update and automatically adjust convergence rate
  - Update weight (states of memristor in the crossbar)



# A Case Study: MNIST

- We use the MNIST dataset as a case study
  - A widely used database of handwritten digits for **optical character recognition**
  - 20,000 examples for training and 5,000 examples for testing
  - Training a 3-layer network
    - 784* (28*28 pixels) *300 (hidden layer)*10 (10 digits (‘0’~’9’) to recognize)
  - Compared with Intel i5-2320 @3.0GHz with MKL



# A Case Study: MNIST

- Results
  - The effect of training depends both on the ***Decay Rate*** and the ***Noise Rate***.
  - > 90% recognition accuracy
  - A slight decrease of the accuracy of recognition (< 5%)
  - 3 orders of magnitude speed-up
  - 4 orders of magnitude energy saving

TABLE I
EXPERIMENT RESULTS OF THE MEMRISTOR IMPLEMENTATION OF TRAINING SCHEME

| *DecayRate* | Noise Rate | Accuracy (%) | Iteration | Time (ms) | Energy (mJ) | Speed Up | Energy Saving | Accuracy Drop (%) |
|---|---|---|---|---|---|---|---|---|
| CPU | | 96.16 | 517,000 | 71067 | 6396030 | - | - | - |
| 1.2 | 0 | 94.84 | 140,000 | 21.0 | 146.89 | 3384 | 43544 | 1.37 |
| | 0.01 | 94.36 | 130,000 | 19.5 | 129.46 | 3644 | 49407 | 1.87 |
| | 0.05 | 94.40 | 134,000 | 20.1 | 132.97 | 3536 | 48102 | 1.83 |
| | 0.1 | 94.14 | 137,000 | 20.55 | 143.38 | 3458 | 44608 | 2.10 |
| 1.5 | 0 | 94.30 | 76,000 | 11.4 | 75.56 | 6234 | 84647 | 1.93 |
| | 0.01 | 93.96 | 77,000 | 11.55 | 81.43 | 6153 | 78543 | 2.29 |
| | 0.05 | 94.76 | 75,000 | 11.25 | 77.37 | 6317 | 82665 | 2.50 |
| | 0.1 | 93.34 | 66,000 | 9.9 | 67.31 | 7178 | 95021 | 2.93 |
| 1.8 | 0 | 93.02 | 50,000 | 7.5 | 49.53 | 9476 | 129128 | 3.27 |
| | 0.01 | 92.60 | 47,000 | 7.05 | 47.52 | 10080 | 134591 | 3.70 |
| | 0.05 | 92.70 | 46,000 | 6.9 | 47.04 | 10300 | 135984 | 3.60 |
| | 0.1 | 92.62 | 45,000 | 6.75 | 48.25 | 10528 | 132572 | 3.68 |

# Conclusion

- Mixed-signal training acceleration framework for memristor-based neural network
  - ✓ **Self-training**
  - ✓ Modified training algorithm
    - approximating calculations
    - designing an alternative computing method
  - ✓ Mixed-signal acceleration architecture
    - Copy crossbar technique
    - Weight update units
    - Sign calculation units ...
  - ✓ **3 orders** of magnitude speed-up
  - ✓ **4 orders** of magnitude energy saving
  - ✓ A slight decrease of the recognition accuracy (<5%).

# REFERENCES


- [1] Jeffrey Dean et al. Large scale distributed deep networks. In P. Bartlett, F.C.N. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, editors, Advances in Neural Information Processing Systems 25, pages 1232–1240, 2012.
- [2] Hadi Esmaeilzadeh, Adrian Sampson, Luis Ceze, and Doug Burger. Neural acceleration for general-purpose approximate programs. In Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture.
- [3] Russell Beale and Tom Jackson. Neural Computing-an introduction. CRC Press, 2010.
- [4] Dmitri B Strukov, Gregory S Snider, Duncan R Stewart, and R Stanley Williams. The missing memristor found. Nature, 453(7191):80–83, 2008.
- [5] Beiye Liu, Miao Hu, Hai Li, Zhi-Hong Mao, Yiran Chen, Tingwen Huang, and Wei Zhang. Digital-assisted noise-eliminating training for memristor crossbar-based analog neuromorphic computing engine. In Proceedings of the 50th Annual Design Automation Conference, 2013.
- [6] Miao Hu, Hai Li, Qing Wu, and Garrett S Rose. Hardware realization of bsb recall function using memristor crossbar arrays. In Proceedings of the 49th Annual Design Automation Conference, 2012.
- [7] Li Boxun, Shan Yi, Hu Miao, Wang Yu, Chen Yiran, and Yang Huazhong. Memristor-based approximated computation. In Proceedings of ISLPED 2013.
- [8] Rafael Men´endez de Llano and Jos´e Luis Bosque. Study of neural net training methods in parallel and distributed architectures. Future Generation Computer Systems, 26(2):267–275, 2010.

# REFERENCES


- [9] Simon S Haykin, Simon S Haykin, Simon S Haykin, and Simon S Haykin. Neural networks and learning machines, volume 3. Prentice Hall New York, 2009.
- [10] Jiquan Ngiam, Adam Coates, Ahbik Lahiri, Bobby Prochnow, Andrew Ng, and Quoc V Le. On optimization methods for deep learning. In Proceedings of the 28th International Conference on Machine Learning (ICML-11), pages 265–272, 2011.
- [11] Sieu D Ha and Shriram Ramanathan. Adaptive oxide electronics: A review. Journal of Applied Physics, 110(7):071101–071101, 2011.
- [12] Wei Yi, Frederick Perner, et al. Feedback write scheme for memristive switching devices. Applied Physics A, 102:973–982, 2011.
- [13] Robinson E Pino, Hai Li, Yiran Chen, Miao Hu, and Beiye Liu. Statistical memristor modeling and case study in neuromorphic computing. In Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE, pages 585–590. IEEE, 2012.
- [14] G. Khodabandehloo, M. Mirhassani, and M. Ahmadi. Analog implementation of a novel resistive-type sigmoidal neuron. TVLSI, 20(4):750 –754, april 2012.
- [15] Rainer Gemulla, Erik Nijkamp, Peter J Haas, and Yannis Sismanis. Large-scale matrix factorization with distributed stochastic gradient descent. In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 69–77. ACM, 2011.
- [16] Yann LeCun and Corinna Cortes. The mnist database of handwritten digits, 1998.
- [17] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. Science, 313(5786):504–507, 2006.
- [18] Intel. Intel math kernel library. http://software.intel.com/en-us/intelmkl/.

# Thank you !

# Q&A