2020 Research Day

MTJ-Based Hardware Synapse Design for Ternary Deep Neural Networks

Tzofnat Greenberg, Ben Perach, Daniel Soudry, and Shahar Kvatinsky

Ternary Neural Networks

Training DNN
Matrix-vector multiplication
Real value of weights and neurons

discretization

Training TNN
Ternary space $\{-1,0,1\}$
Logic operations between weights and neurons (GXNOR)

DNN- High overhead- hardware computation and memory intensive, TNN constrains the weights to $\{-1,0,1\}$ and **replaced the multiply-accumulate operations with Gated-XNOR operation**.

-1	-1	**1**
-1	0	**0**
-1	1	**-1**
0	-1	**0**
0	0	**0**
0	1	**0**
1	-1	**-1**
1	0	**0**
1	1	**1**

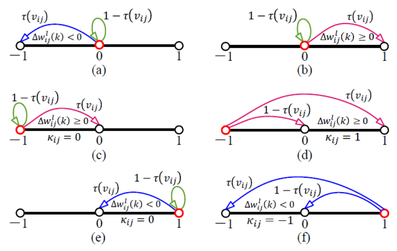
Gated XNOR- If one of the inputs is zero then the output is zero else XNOR

$u \,\&\, w\,! = 0$

u	w	***out***
-1	-1	**1**
-1	1	**-1**
1	-1	**-1**
1	1	**1**

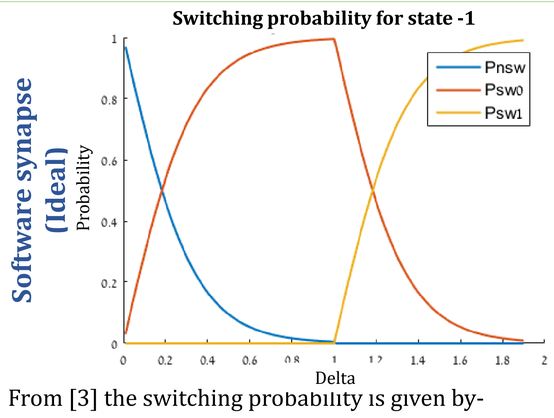
Weight Update-
Based on the work "Gated XNOR Networks: Deep Neural Networks with Ternary Weights and Activations under a Unified Discretization Framework" [3]

- The weight update value are restricted to the discrete ternary space.
- For calculating the update there is no need to keep the full precision value of the weights.
- The weight increment $\Delta w_{ij}(k)$ is a stochastic function of the Gradient $\Delta W_{ij}^{l}(k) = -\eta \frac{\partial E(W(k),Y(k))}{\partial W_{ij}^{l}(k)}$.



From [3]- discrete state transition in ternary weight space

Switching Probability



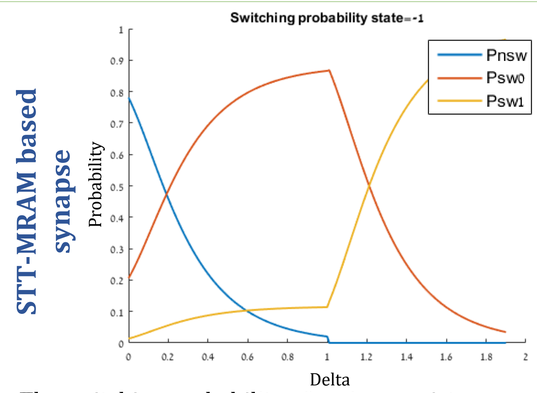
From [3] the switching probability is given by-

$$P\left(\Delta w_{ij}(k) = k_{ij} + sign\left(\varrho\left(\Delta W_{ij}^{l}(k)\right)\right)\right) = \tau(v_{ij})$$

$$P(\Delta w_{ij}(k) = k_{ij}) = 1 - \tau(v_{ij})$$

$\tau - state\ transition\ probability$

$\tau(v) = \tanh(m|v|)$



The switching probability of each memristor

$$P\left(\Delta t \frac{u}{R} > \epsilon\right) = 1 - \mathrm{erf}\left(\frac{\pi}{2\sqrt{2}\exp\left(\frac{\Delta t x}{cR}\right)}\right)$$

$$P_{right} \propto f\left(\Delta W_{ij}^{l}(k) - \lfloor \Delta W_{ij}^{l}(k) \rfloor\right)$$

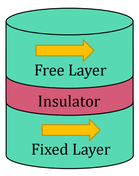
$$P_{left} \propto f\left(\lfloor \Delta W_{ij}^{l}(k) \rfloor\right)$$

*assuming high current density

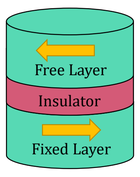
STT-MTJ

Magnetic tunnel junction (MTJ)

- Two ferromagnetic electrodes separated by an insulator barrier
- Free layer- this layer polarization can be set by the current flowing through it (**Spin-transfer torque**)
- Fixed layer ("Pinned")- fixed polarization, used as reference layer



Parallel polarization- **Low resistance R_{on}**



Ani-parallel polarization- **High resistance R_{off}**

Properties

- Nonvolatile
- High endurance
- CMOS compatible
- Low power consumption
- High write and read speed
- **Stochastic switching delay**

Stochastic switching delay

Critical current of the device

$$I_{c_0} = \frac{2|e|}{\hbar}\frac{\alpha V(1 \pm P)}{P}\frac{\mu_0 M_s M_{eff}}{2}$$

For $I \ll I_{c0}$ low current regime

$$P_{sw} = 1 - \exp\left(-\frac{\Delta t}{\langle\tau\rangle}\right)$$

$$\langle\tau\rangle = f_0^{-1}\exp\left(\frac{E_0}{k_B T}\left(1 - \frac{I}{I_{c0}}\right)\right)$$

For $I \gg I_{c0}$ high current regime

$$\tau = C\frac{1}{I - I_{c_0}}\log\left(\frac{\pi}{2|\theta|}\right)$$

$$\theta \sim N(0,\theta_0), \theta_0 = \sqrt{\frac{K_B T}{\mu_0 H_K M_s V}}$$

Training With STT-MTJ Based Synapse

Initial results

Plugging the STT-MTJ based synapse switching behavior to CNN network training algorithm

Setup

- Dataset- MNIST 28x28, 60000 training set, 10000 validation set
- Batch size-10000
- Number of epochs- 1200
- CNN architecture-

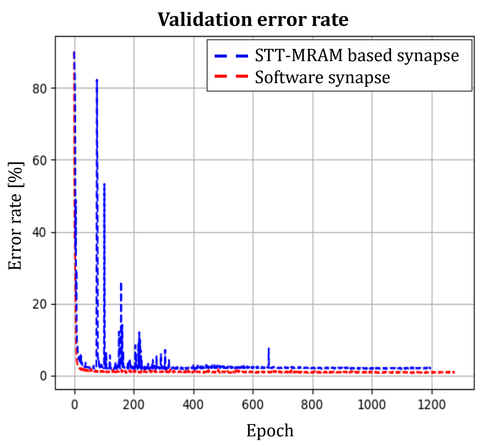
#layer	Type	Size
1	Input	28x28
2	Conv	32×5× 5
3	Pooling	2 × 2
4	Conv	64 ×5× 5
5	Pooling	2x2
6	FC	512
7	FC	10

Results

- The network manage to converge to similar error rate as the software performance
- The convergence suffers from "Noise"

Future steps

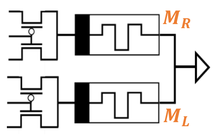
- Improve the STT-MTJ switching models
- "Play" with the STT-MTJ properties



Ternary Synapse Based STT-MTJ

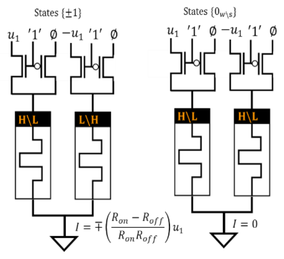
Need to support 3 operations:

1. GXNOR ("read")
2. Inverse read
3. Write



M_L	M_R	State
R_{off}	R_{off}	0
R_{off}	R_{on}	-1
R_{on}	R_{off}	1
R_{on}	R_{on}	0

GXNOR



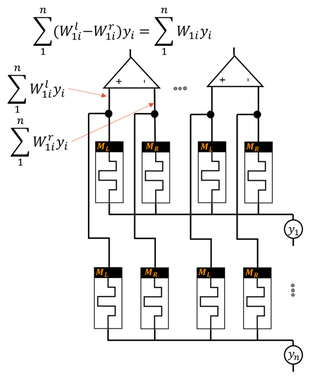
- At the feedforward stage the voltage supply ("the previous layer neuron") of both memristor are $u_{l,i} = -u_{r,i} = x_i^l$
- The current is given by

$$I = \left(\frac{M_R - M_L}{M_L M_R}\right)u$$

- Example u=-1

w	M_L	M_R	Out
0	R_{off}	R_{off}	0
-1	R_{off}	R_{on}	1
1	R_{on}	R_{off}	-1
0	R_{on}	R_{on}	0

Inverse read

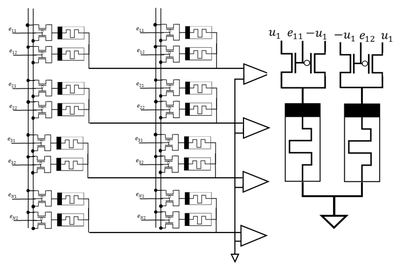


For propagating the error back through the network the value $W^T y$ need to be calculated.

The rows acts as the input with voltage level y

The output current per column per memristor (H\L) is summed and then compared.

Write



The gradient value $\Delta W = x^T y$
Notice- $|x| \in \{0,1\}$

The left memristor is updated with respect to $\lfloor \Delta W \rfloor$

$$e_{i1} = \begin{cases} sign(\lfloor y_i \rfloor), & 0 < t < abs(\lfloor y_1 \rfloor) \\ 0, & abs(\lfloor y_1 \rfloor) < t < T_{wr} \end{cases}$$

The right memristor is updated with respect to $\Delta W - \lfloor \Delta W \rfloor$

$$e_{i2} = \begin{cases} sign(y_i - \lfloor y_i \rfloor), & 0 < t < abs(y_i - \lfloor y_i \rfloor) \\ 0, & abs(y_i - \lfloor y_i \rfloor) < t < T_{wr} \end{cases}$$

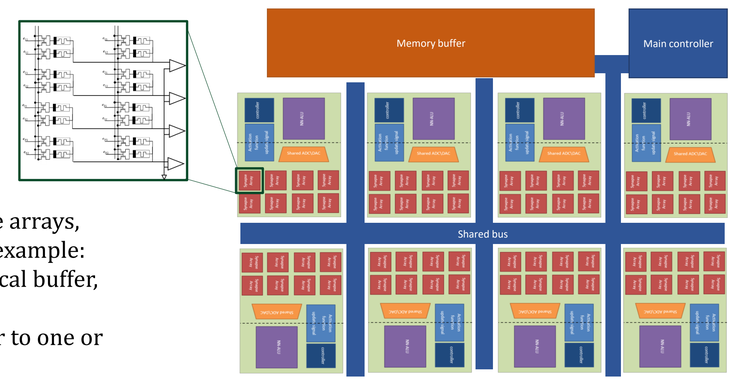
TNN Hardware Architecture

Improve power consumption and run-time

- Reduce memory accesses
- In-memory computation of the GXNOR (= dot-product)

Main architecture concept

- Tile based architecture
- Each tile contain: several synapse arrays, shared local computation units (example: activation and its' derivatives), local buffer, etc…
- Main controller to map each layer to one or more synapse arrays



Reference:

1) Tzofnat Greenberg-Toledo, Ben Perach, Daniel Soudry, Shahar Kvatinsky "MTJ-Based Hardware Synapse Design for Quantized Deep Neural Networks". CoRR abs/1912.12636 (2019)
2) Vincent, Adrien F., et al. "Spin-transfer torque magnetic memory as a stochastic memristive synapse for neuromorphic systems." IEEE transactions on biomedical circuits and systems 9.2 (2015): 166-174
3) Vincent, Adrien F., et al. "Analytical macrospin modeling of the stochastic switching time of spin-transfer torque devices." IEEE Transactions on Electron Devices 62.1 (2015): 164-170.
4) Deng, Lei, et al. "Gated XNOR Networks: Deep Neural Networks with Ternary Weights and Activations under a Unified Discretization Framework." arXiv preprint arXiv:1705.09283 (2017).