Training of a Spiking Neural Network on spintronics based analog hardware for handwritten digit recognition

SUBMISSION DATE / POSTED DATE

09-06-2021 / 11-06-2021

CITATION

Sahu, Upasana; Goyal, Kushaagra; Bhowmik, Debanjan (2021): Training of a Spiking Neural Network on spintronics based analog hardware for handwritten digit recognition. TechRxiv. Preprint. https://doi.org/10.36227/techrxiv.14757972.v1

DOI

10.36227/techrxiv.14757972.v1

Training of a Spiking Neural Network on spintronics based analog hardware for handwritten digit recognition

Upasana Sahu
Department of Electrical Engineering
Indian Institute of Technology Delhi
New Delhi, India
Upasana.Sahu@ee.iitd.ac.in

Kushaagra Goyal
Department of Mechanical Engineering
Indian Institute of Technology Delhi
New Delhi, India
kg13344135@gmail.com

Debanjan Bhowmik
Department of Electrical Engineering
Indian Institute of Technology Delhi
New Delhi, India
debanjan@ee.iitd.ac.in

***Abstract*—Spiking Neural Network (SNN) has been shown to consume very low power for an inference task, i.e., forward computation on the test data with a pre-trained model. However, training of such SNN has remained a challenge. In this paper, we use Spike Time Dependent Plasticity (STDP) based learning to train the SNN through implementation on an analog hardware. We design and simulate the synapses and neurons in the hardware using a combination of ferromagnetic metal- heavy metal based spintronic devices and transistor based electronic circuits. We train the SNN on the MNIST dataset of handwritten digits. The architecture used by us has less number of network layers that that used previously for MNIST classification. We use two different modes (both STDP enabled but with different level of control over spiking of the output stage neurons or post-neurons) for the training/learning: completely unsupervised learning and partially supervised learning. Finally, we report the time consumed and total energy consumed in our designed synapse circuits to enable the learning.**

***Index Terms*—Neuromorphic Computing, Analog Hardware Neural Network, Spiking Neural Network, Spintronics**

I. Introduction

A. Motivations for training Spiking Neural Network (SNN) on analog hardware

Spiking Neural Network (SNN) algorithm is considered to be of special interest [1]–[5] among the different Neural Network (NN) algorithms proposed for data classification. SNN processes information in the form of spikes much like the brain [6]–[9]. As a result of the use of spikes, SNN has been shown to consume very low power for inference, i.e. forward computation of pre-trained model with already tuned weights on test data [10]–[12]. This has led to the interest in SNN. However training the SNN has remained a challenge [10].

One possible way of training the SNN is to train a Non-Spiking NN first and then convert it to SNN, but this method has several limitations [10], [13], [14]. The alternate way is to train the SNN itself. SNN has already been trained on CPU-s and GPU-s by simulating biologically inspired processes like Leaky Integrate Fire (LIF) property of neuron, Spike Time Dependent Plasticity (STDP) of synapse, etc. on the CPU or GPU. SNN has also been trained on customized digital neuromorphic chips [15]–[17]. The time dynamics of LIF and STDP processes need to be solved for on the CPU, GPU or customized neuromorphic chip for the purpose. This is highly time and energy consuming because such computing units are fundamentally clock driven. A process like LIF or STDP has to be split into a multitude of time steps and solved for each time step on such a clock driven computing unit [10]. A more efficient way to train the SNN is hence on analog hardware with emerging devices where the physics of the devices mimics the time dynamics of the synapses (STDP) or the neurons (LIF). Hence such time dynamics is computed automatically. Spintronic devices, coupled with transistor based electronic circuits, form one such class of emerging devices [18]–[24].

Another motivation to train SNN on analog hardware designed with emerging devices is the in-memory computing architecture it offers. The CPU follows the Von Neumann architecture and hence has memory and computing separated from each other. Hence a lot of time and energy is consumed in shuffling data between the memory and computing units. This is known as the Von Neumann bottleneck. It becomes particularly critical in the case of training NN-s because of the frequent need to tune the weight parameters [10], [25]. Though the GPU or the specialized neuromorphic chips [15]–[17] have a multitude of cores, memory and computing units are still physically separated inside each core. Hence the Von Neumann bottleneck for computation is not completely eliminated in them either. However analog hardware designed with emerging non-volatile devices, e.g. Resistive Random Access Memory (RRAM), Phase Change Memory (PCM) and spintronic devices, mostly follows a crossbar architecture and offers complete memory-computing intertwining [25]–[35] and eliminates the Von Neumann bottleneck altogether. Since the SNN is trained here using STDP property of synapses as opposed to optimization of a global loss function as in non-spiking NN [36], the training is very local in nature. Thus it is particularly suitable for implementation on such analog in-memory computing hardware [11].

B. Motivation for choosing spintronic hardware SNN

In this paper, we use heavy metal/ ferromagnetic metal heterostructure based spin orbit torque driven domain wall devices, coupled with transistor based electronic circuits, both as synapses and neurons in our designed analog hardware SNN. We choose such spintronic devices as opposed to other emerging devices [25]–[27], [29]–[33] for the following two reasons.

1. The same kind of spintronic device can enable both neuron and synapse functionality [18], [21], [24]. The nature of the transistor based circuit connected to it just changes in either case.

2. Such a domain wall based spintronic device has a linear and symmetric synaptic characteristic unlike RRAM and PCM devices, making it easier to achieve learning in the spintronic hardware SNN since learning involves frequent positive and negative weight updates at the synapses [25], [37]–[40].

C. Work done in this paper

Through a combination of micromagnetic modeling of the spintronic devices on a micromagnetic package "mumax3", simulation of the transistor circuits on Cadence Virtoso SPICE circuit simulator and system level simulations of the SNN on a high level programming language, we demonstrate training of the designed SNN on the popular MNIST dataset of handwritten digits.

The novelties of our work compared to previous works are as follows:

1. Training on the MNIST dataset with spintronic hardware has been reported earlier [21]. The algorithm used in [21] is based on that proposed in [6]. It employs a SNN with three layers. On the other hand, the SNN we use here has only two layers - a layer of pre-neurons and a layer of post-neurons (Fig. 1(a),(b)). This makes hardware implementation simpler in our case.

2. Only a completely unsupervised learning method is used in [6], [21], with the synapses exhibiting biologically inspired STDP property and neurons exhibiting biologically inspired homoeostasis property. In our implementation, we train the SNN in two modes. One mode is the STDP and homoesostasis enabled completely unsupervised mode used in [6], [21] (Fig. 1(a)). The other mode is a partially supervised mode. In this mode, the synapses still update their weights via STDP contributing to the partially unsupervised nature of the learning. However the partially supervised nature comes from applying bias currents to the output stage neurons or post-neurons only during the learning phase to control them and make them fire selectively (Fig. 1(b)) [20], [24], [41], [42]. While SNN has been trained on simple datasets like the Fisher's Iris dataset and Wisconsin Breast Cancer dataset under the partially supervised mode in the previous reports [20], [24], [41], [42], training on a more complex dataset involving more input features and more output classes like the MNIST dataset has not been demonstrated before whether with spintronic hardware or otherwise, to the best of our knowledge.

3. Earlier works [24], [43] only report the energy consumed in the spintronic devices themselves for the STDP enabled learning. In this paper, we have reported the net energy consumed in the peripheral transistor circuits as well. Some other earlier works [11], [12] on spintronic implementation of SNN only report energy consumption for inference.

The paper is organized as follows. In Section II we discuss our SNN training algorithm, the layers used in the SNN and the two modes for training/ learning (Fig. 1). We also report the classification accuracy results obtained on the MNIST dataset with our algorithm (Fig. 2, Fig. 3, Fig. 4). In Section III we discuss our simulation of the spintronic devices (Fig. 5, Fig. 6) and associated transistor based electronic circuits (Fig. 7), which are used to obtain STDP property of synapses (Fig. 8) and LIF property of neurons. In Section IV we show our calculation of the total time taken and the net energy consumed in all the synapse circuits to achieve the learning (Table I).

II. Our SNN training algorithm

A. Description of the algorithm

The SNN, designed by us in this paper, has a layer of pre-neurons, to which the input is applied, and a layer of post-neurons, the spiking pattern of which determines the output label corresponding to the input generated by the SNN (Fig. 1). Since we classify the MNIST dataset of hand-written digits here, $N_1 = 784$ pre-neurons are used corresponding to the 784 pixels in each input image of the dataset. $N_2 = 400$ post-neurons are used with several neurons designed to spike corresponding to the different variations with which the same digit is written, using the homoeostasis property as discussed in [6], [21]. However, as mentioned in Section I, [6] and [21] use a separate layer of excitatory neurons and a separate layer of inhibitory neurons at the output stage, unlike here where we use only a single layer of post-neurons [24], [41], [42]. Thus our network is simpler in design than that in [6], [21].

Every pre-neuron and post-neuron in our designed SNN follows the biologically motivated Leaky Integrate Fire (LIF) model [6], [8], [24]. Following the LIF model, the membrane potential $v(t)$ of each neuron is governed by the following equation:

$$C\frac{\partial v(t)}{\partial t} = -G_L(v(t) - E_L) + I(t) \tag{1}$$

where $I(t)$ is input current to the neuron, G_L is membrane conductance, E_L is resting potential of neuron. Once $v(t)$ reaches the threshold potential (V_{th}), the neuron generates a spike and $v(t)$ drops to E_L. For our designed SNN, we choose G_L = 30 nS , $E_L = -70$ mV and C = 500 fF for the LIF model of both the pre-neurons and post-neurons.

For the pre-neurons, V_{th} is fixed at 20 mV. For the post-neurons, V_{th} is a function of time following the biologically plausible homoeostasis property of neurons [6], [9]. Every time the neuron spikes (t_{spike}), V_{th} increases by a fixed

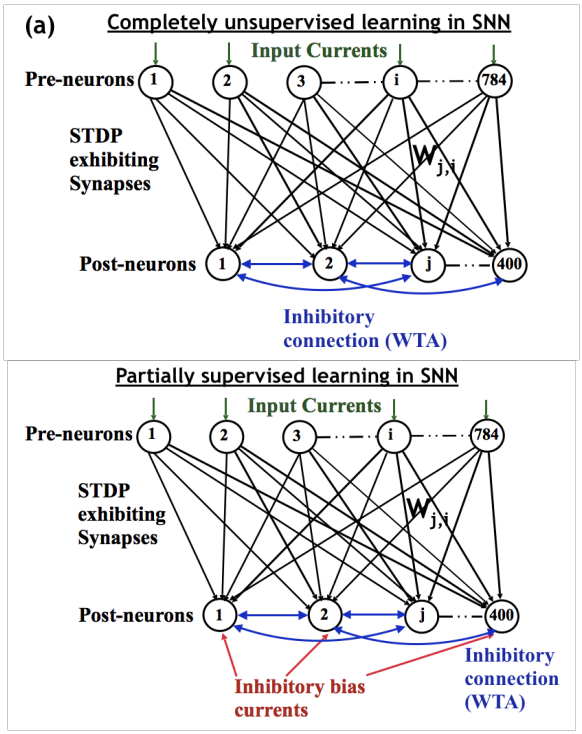

Fig. 1. (a) Schematic showing design of SNN with completely unsupervised mode of learning for the MNIST dataset. (b) Schematic showing design of SNN with partially supervised mode of learning for the MNIST dataset. In the partially supervised mode, inhibitory bias currents are applied on selected post-neurons to prevent them from spiking.

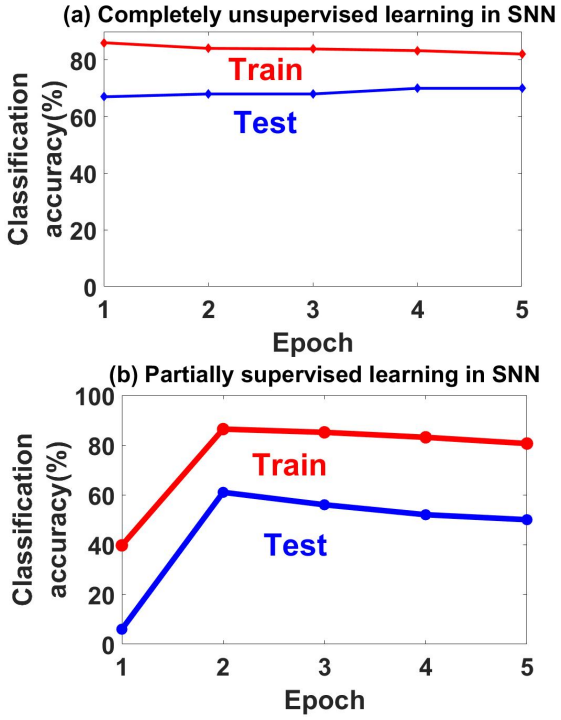

Fig. 2. Classification accuracy as a function of epoch for 1000 train images and 100 test images from the MNIST dataset using two modes of learning in our designed SNN- (a) completely unsupervised learning and (b) partially supervised learning.

amount ΔV_{th} = 7 mV and then decays with a homoestasis time constant τ_{homeo} = 15 μs as follows:

$$V_{th}(t) = V_{th}(t_{spike}) + \Delta V_{th} e^{-(\frac{t-t_{spike}}{\tau_{homeo}})} \quad (2)$$

for time (t) greater than t_{spike} until the occurrence of the next spike in the neuron.

Each pre-neuron is connected to each post-neuron in our designed SNN through a synapse, as shown in Fig. 1(a),(b). When the pre-neuron, numbered i, fires at time $t_{spike,i}$, the post-neuron, numbered j receives an input current from the pre-neuron i, as given by:

$$I_{j,i}(t) = I_0 w_{j,i} (e^{-\frac{t-t_{spike,i}}{\tau_M}} - e^{-\frac{t-t_{spike,i}}{\tau_S}}) \quad (3)$$

where τ_M = 10 μs and τ_S = 2.5 μs are two time constants [41], [42]. $w_{j,i}$ is the weight stored in the synapse, which is updated following the biologically plausible Spike Time Dependent Plasticity (STDP) rule as follows

$$\Delta w_{j,i} = \Gamma_1 (1 - \frac{w}{w_{max}})^{\mu}) e^{-(\frac{t_{spike,j} - t_{spike,i}}{\tau_1})} \quad if \;\; t_{spike,j} > t_{spike,i};$$
$$\Delta w = -\Gamma_2 (\frac{w}{w_{max}})^{\mu} e^{-(\frac{t_{spike,i} - t_{spike,j}}{\tau_2})} \quad if \;\; t_{spike,j} < t_{spike,i} \quad (4)$$

where Γ_1 = 9, Γ_2 = 15, τ_1 = 10 μs, τ_2 = 20 μs and $\mu = 1.7$ are constants related to STDP [41], [42].

The increase in weight update that happens when post-neuron j spikes after the pre-neuron i ($t_{spike,j} > t_{spike,i}$) is known as "synaptic potentiation", while the decrease in weight update that happens when post-neuron j spikes before the pre-neuron i ($t_{spike,j} < t_{spike,i}$) is known as "synaptic depression". Since the weight update and hence learning happens in the synapse based on the spiking pattern of the pre-neuron i and the post-neuron j, the learning is local in nature. It also is largely unsupervised, though some amount of supervision may be provided by controlling the spiking of the post-neurons [41], [42]. Thus as mentioned in Section I, we use two modes of learning/ training for our designed SNN in this paper- completely unsupervised learning, and partially supervised learning. In either case, each post-neuron exhibits homoeostasis property as described above. Also in either case, the post-neurons are connected inhibitorily with each other to implement the "Winner Take All" (WTA) mechanism, as shown in Fig. 1(a),(b) [6], [41], [42].

However, in the partially supervised learning mode, during the training phase, when the input belongs to a particular digit/ class, no inhibitory current is applied on 40 post-neurons allotted for that digit but inhibitory current is applied on the other post-neurons so that only some specific neurons of those 40 post-neurons fire (Fig. 1(b)) [24], [41], [42]. Thus knowledge of the digit/ class each input belongs to is used in

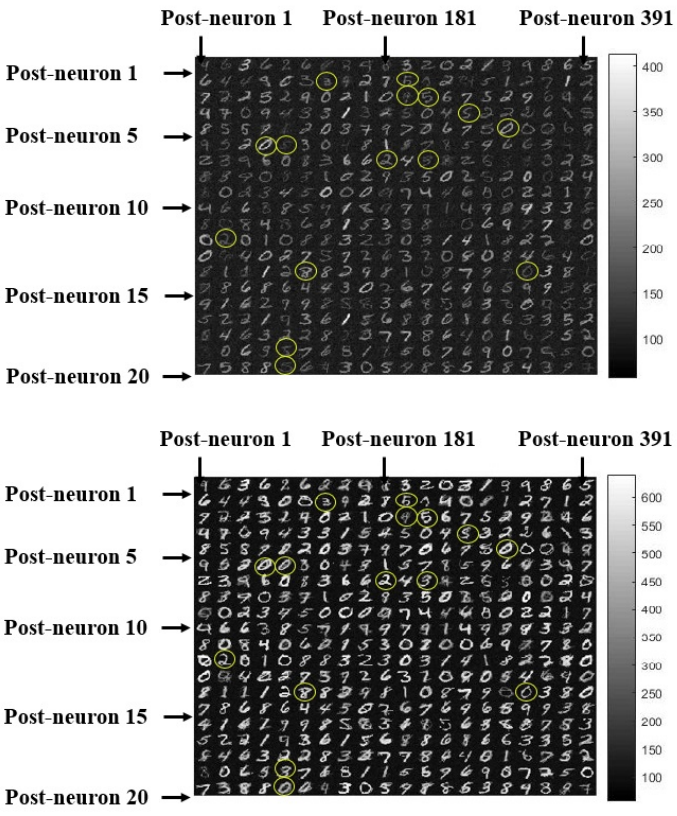

Fig. 3. (a)After 1st epoch of completely unsupervised learning on the MNIST dataset, weights of all the synapses connecting the 784 pre-neurons to a post-neuron are plotted as a 28x28 pixel grayscale image. Intensity of each pixel is proportional to the value of each weight. Images corresponding to all such 400 post-neurons are then plotted together. (b) Same plot of synaptic weights after 5th epoch of completely supervised learning.

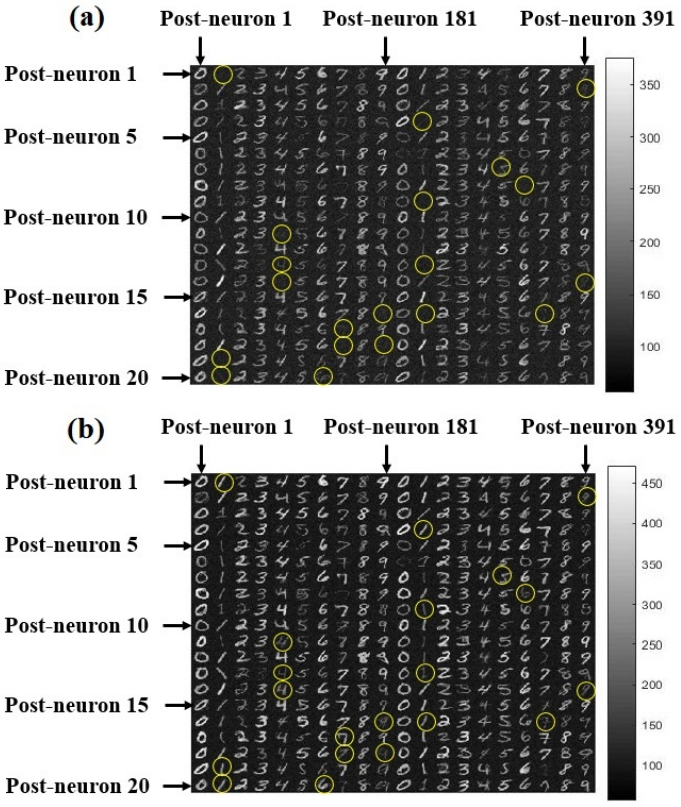

Fig. 4. (a) After 1st epoch of partially supervised learning on the MNIST dataset, weights of all the synapses in the SNN are plotted as a grayscale image, following the same method as in Fig. 3. (b) Plot of synaptic weights after 2nd epoch of partially supervised learning.

the learning phase for mode (b) of learning.

In the completely unsupervised learning mode, no such inhibitory current is applied on the post-neurons at any stage (Fig. 1(a)). Knowledge of what digit/ class each input belongs to is not needed at the training/ learning stage, but only while determining the classification accuracy after training. To determine classification accuracy, all the inputs corresponding to the different digits are applied on the SNN. For each post-neuron, it is noted how many times the neuron fires corresponding to the different inputs corresponding to each digit/ class. The neuron is assigned the label of the digit/ class for which it fires the most. Then for each input, if the label of the post-neuron that fires the most matches with the digit/ class the input belongs to, it is considered a success and increases the classification accuracy [6], [24].

B. Accuracy results and related discussion

Using 1000 images from the MNIST dataset for training and 100 images for testing, we obtain a fairly high train and test dataset classification accuracy both for completely unsupervised and partially supervised learning, as shown in the accuracy vs epoch plots of Fig. 2(a) and (b) respectively. Reasonably high accuracy is obtained after 1 or 2 epochs only, and the accuracy metric does not improve after that. This is because unlike non-spiking NN, no global loss function is optimized here to train the weights [36]. Instead a very local spiking based feedback method is used. Hence gradual increase in accuracy with epoch is not expected here as opposed to the case with loss function based non-spiking NN algorithms [36].

Rather training accuracy decreases with the number of epochs in the case of completely unsupervised learning (Fig. 2(a)). We can explain the result by studying the evolution of the synaptic weights in the SNN with epochs. Fig. 3(a) shows the weights for completely unsupervised learning after first epoch (train accuracy = 86 %) and Fig. 3(b) shows the weights for the same learning after the 5th epoch (train accuracy = 82 %). In each case, weights of all the synapses connecting the 784 pre-neurons to a post-neuron are plotted as a 28x28 pixel grayscale image. Images corresponding to all such 400 post-neurons are then plotted together. Intensity of each pixel is proportional to the value of each weight (see vertical bar next to the image). Minimum value of weight allowed is 0 and maximum value is 900.

In order to identify the reason for drop in accuracy, in Fig. 3(a) and (b), we have put yellow circles around the images corresponding to weights of synapses attached to those

post-neurons, which have correct results after 1st epoch but not after 5th epoch. After 1st epoch, these post-neurons fire the most for input images, belonging to the class/ label that match with the label assigned to those post-neurons. But after 5th epoch, the two things don't match leading to drop in accuracy (Fig. 3(a). This happens because the network learns as the synaptic weights form impressions of the different images. As the number of epochs increase, impression of image belonging to a particular digit/class/ label can superpose on image belonging to another digit/class/ label as can be observed for post-neurons corresponding to the yellow circles in Fig. 3(a) and (b). This leads to the inaccuracy.

Fig. 4(a) and (b) show the weights of the SNN in the case of partially supervised learning after the 1st and 2nd epoch respectively, plotted in the same way as before. In the case of partially supervised learning, since inhibitory bias currents are applied on all post-neurons other than post-neuron 1-20 and post-neuron 201-220 during training for input images belonging to class/ digit "0", post-neurons 1-20 and 201-220 capture different variations of digit "0". Same is true for the other digits. On the other hand, in the case of unsupervised learning, since no supervision is provided to any specific post-neuron in the form of bias current, the different post-neurons randomly capture impressions of different digits from "0" to "9", as expected [6] (Fig. 3(a),(b)).

The sharp increase in accuracy from 1st epoch to 2nd epoch in the case of partially supervised learning (Fig. 2(b)) can also be explained from the images of synaptic weights in Fig. 4(a) and (b). As shown through yellow circles around images of synaptic weights corresponding to specific post-neurons, those post-neurons do not fire correctly after 1st epoch because their corresponding synaptic weights do not yet form impression of image belonging to any class/digit/label. After 2nd epoch, those impressions form, the neurons fire correctly and accuracy increases.

III. Implementation of our SNN with spintronic devices

As mentioned in Section I, spintronic devices- more specifically spin orbit torque driven domain wall devices- are ideal for implementation of analog hardware SNN because such devices can act both as synapses and neurons [18]–[24]. We discuss the characteristics of the domain wall devices and the associated transistor based circuits, that enable functionality of synapses and neurons in them, next.

A. Domain wall synapse

Current driven domain wall motion in heavy metal/ ferromagnetic metal heterostructure based spintronic device has been utilized to propose and experimentally demonstrate synaptic behavior in such a device [34], [35], [44], [45]. In-plane current, also known as "write" current, flowing through the device moves the ferromagnetic domain wall in the ferromagnetic layer even in the absence of magnetic field as observed in experiments and simulations [46]–[50] (Fig. 5(a)). For a fixed pulse duration (3 ns in our case-

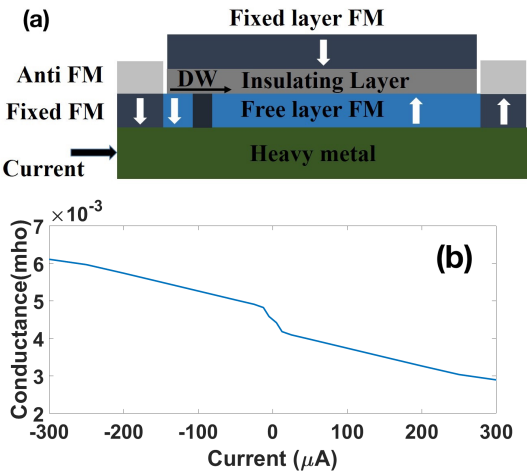

Fig. 5. (a) Schematic of Domain Wall (DW) motion based synaptic device. Arrows represent direction of the magnetic moments. The domain wall moves in the free layer FerroMagnet (FM) of the vertical tunnel junction structure. Anti FerroMagnet (FM) layer prevents destruction of domain wall at the edges. (b) Conductance of vertical Magnetic Tunnel Junction (MTJ) structure (Fixed FM layer/ Insulating oxide layer/ Free FM layer) in the device as function of the programming current which moves the domain wall.

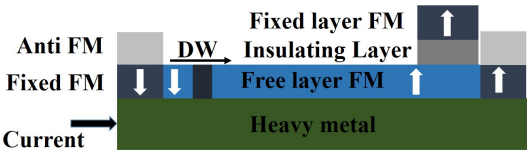

Fig. 6. Schematic of Domain Wall (DW) motion based neuron device.

Fig. 5(b)), the wall moves by different lengths for different magnitudes of the current pulse. Different positions of the domain wall correspond to different non-volatile conductance states of the vertical Magnetic Tunnel Junction (MTJ) structure (fixed ferromagnetic layer/ insulating oxide layer/ free ferronagnetic layer). This is because according to the physics of the Tunneling Magneto Resistance (TMR) effect [51], [52], conductance of the MTJ is proportional to the average perpendicular magnetization (in the vertically down direction) in the free ferromagnetic layer of the MTJ, which changes as the domain wall moves. The different conductance values correspond to the different values of weight stored in the synapse (Fig. 5(b)).

We carry out micromagnetic simulations on the numerical package "mumax3" [53] to model the domain wall motion and obtain the variation of conductance of the device as a function of the magnitude of the programming current pulse. We choose paramaters with respect to Pt (heavy metal)/ CoFe (ferromagnetic layer)/ MgO (insulating oxide) system and calibrate our model against experimental data [46]–[50]. More details of our simulation method can be found in [24], [35], [40]. Transistor based peripheral circuit has also been designed (Fig. 7) to enable STDP based weight update characteristic in the domain wall device, as given by equation (4). Transistor T3 drives current into the domain wall synapse when the spike

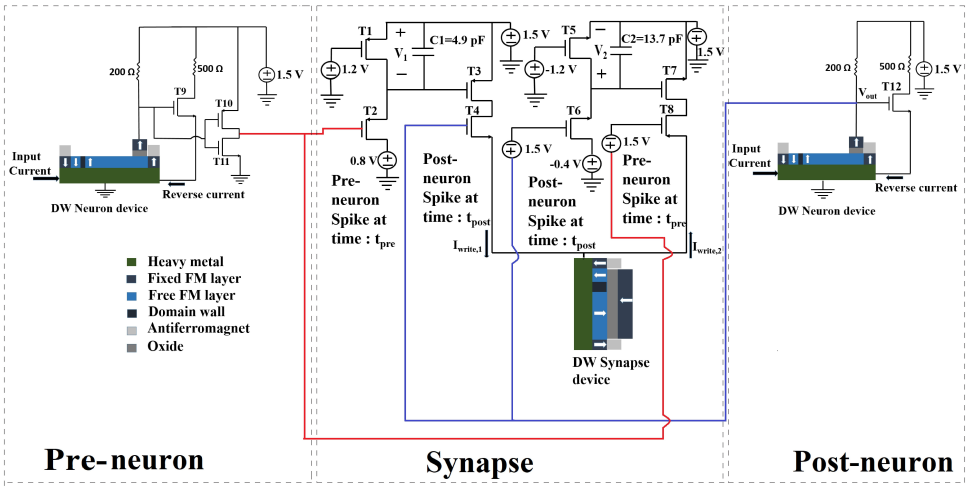

Fig. 7. Transistor based circuit connected to the domain wall devices to enable STDP functionality for synapse and LIF property for neuron. Write current ($I^{write} = I_{write,1} - I_{write,2}$) flows into the domain wall device of the synapse.

at the gate of transistor T4 (post-neuron spike) occurs after the spike at the gate of transistor T2 (pre-neuron spike) (Fig. 7). Current flows inside the domain wall synapse in a direction such that conductance of the synapse increases ("synaptic potentiation") following the conductance versus current plot of Fig. 5(b). At the time of pre-neuron spike, T2 turns on and voltage across capacitor C1 becomes equal to V_1 = 1.5-0.8 = 0.7 V (Fig. 7). Then it is discharged steadily with time through transistor T1 which approximately acts as a constant current. Thus voltage across C1 decreases linearly with time, leading to linear increase in gate voltage of T3 with time. T7 operates in the sub-threshold regime. Hence the current driven by T3 ($I_{write,1}$), through T4 when T4 turns on, to the domain wall device is an exponential function of the negative of time difference between spiking time of post-neuron and pre-neuron [54]. Thus the positive weight update ("synaptic potentiation") given by equation (4) is achieved in the domain wall synapse.

Similarly, T7 drives current $I_{write,2}$ into the domain wall synapse when the spike at the gate of transistor T8 (pre-neuron spike) occurs after the spike at the gate of transistor T6 (post-neuron spike) (Fig. 7). In this case, current flows inside the domain wall synapse in a direction to that in the previous case. Hence, net in-plane "write" current flowing through the domain wall device $I^{write} = I_{write,1} - I_{write,2}$ (Fig. 7). As a result, the conductance of the synapse decreases as desired ("synaptic depression"). The exponential dependence on negative of time difference between pre-neuron spike and post-neuron spike in equation (4) is achieved in this case through the steady discharge of capacitor C2 with time [11], [21], [24].

We simulate this transistor based electronic circuit in Fig. 7, which enables STDP property in the domain wall synapse as described above, on Cadence Virtuso SPICE circuit simulator meant for analog design. From the SPICE simulation, conductance change of the domain wall synapse device is obtained as a function of timing difference between a spike at a post-neuron connected to it and a spike at pre-neuron connected to it (Fig. 8). The desired STDP characteristic from equation (4) is obtained for both positive conductance/ weight update ("synaptic potentiation"- red plot) and negative conductance/ weight update ("synaptic depression"- blue plot). All transistors are designed in the 65 nm technology node. Transistor parameters and capacitance values of capacitors C1 and C2 are chosen (C1 = 4.9 pF, C2= 13.7 pF)) such that the STDP time constant $\tau_1 = 10\ \mu s$ and $\tau_2 = 20\ \mu s$, as desired in our SNN algorithm of Section II.

The design of the transistor circuit makes sure that the current driven into the domain wall synapse depends exponentially on the difference between time of occurrence of a spike at a pre-neuron and that at a post-neuron. However it is to noted that the weight update, caused by that current flow, also follows the same exponential relationship owing to the fact that the conductance of the domain wall synapse varies linearly and symmetrically (between positive and negative current) with in-plane current flowing through the device (Fig. 5(b)). This linear and symmetric characteristic is absent in other Non Volatile Memory (NVM) devices proposed for similar application, e.g., RRAM and PCM synapses [25], [37]–[40]. Hence we have chosen spintronic synapse (domain wall synapse) to implement the hardware SNN in our paper, as we have already mentioned in Section I.

B. Domain wall neuron

The same physics of spin orbit torque driven domain wall motion in a ferromagnetic metal/ heavy metal heterostructure

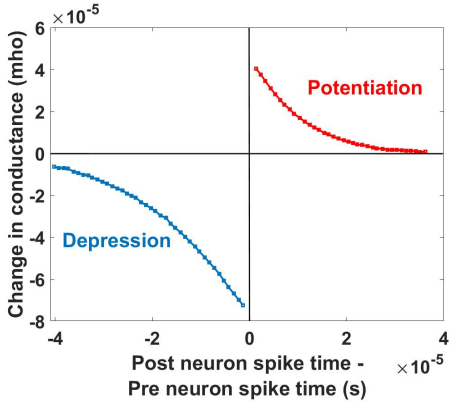

Fig. 8. Change in conductance, proportional to weight update, of the domain wall synapse is obtained as a function of difference in time of occurrence of spike at a post-neuron and at a pre-neuron connected to it, from SPICE simulation of the circuit in Fig. 7.

has been utilized to propose the domain wall based neuron device [18], [24]. The difference in design with respect to the domain wall synapse is that the MTJ structure is present only at one end of the ferromagnetic track, in which the domain wall moves (Fig. 6). Thus conductance of the MTJ remains unchanged for most of the domain wall motion in the track. Then it drops sharply when the domain wall reaches underneath the MTJ structure (Fig. 6) because then the magnetic moment of the free layer is then switched to the vertically down direction. This results in an anti-parallel alignment of the moments in the fixed and free layer, leading to a drop in conductance following the Tunneling Magneto Resistance (TMR) effect [51], [52].

Because of the potential divider circuit in Fig. 7, when the conductance in the MTJ drops, the voltage V_{out} in the circuit associated with the neuron device (Fig. 6) increases sharply. The operating principle of the STDP synapse circuit in Fig. 7 is such that the post-neuron needs to generate positive spike, i.e. the gate voltage of T4 and T6 need to increase sharply with time for succesful weight update at the domain wall synapse [24]. This is accomplished by a sharp increase in V_{out}. The pre-neuron needs to generate negative spike, i.e. the gate voltage of T2 and T6 need to drop sharply with time [24]. Hence, an extra inverter circuit, comprising transistors T10 and T11, is present at the output stage of the pre-neuron circuit which is not present for the post-neuron circuit. Whenever the domain wall reaches the end of the track in the pre-neuron or post-neuron circuit, transistor T9 or T12 turns on. This applies reverse current that moves the domain wall back to its initial position. This is equivalent to the potential dropping to its reset value after every spike in the LIF model [8], [24].

Just like the transistor circuit to enable synaptic property in Fig. 7, we simulate the transistor circuit that enables pre-neuron and post-neuron functionalities in Fig. 7 on Cadence Virtuoso SPICE circuit simulator.

IV. Energy consumption and speed metrics for training the spintronic hardware SNN

In this section, we compute the time taken and the energy consumed in implementing the SNN training algorithm discussed in Section II on the spintronic hardware discussed in Section III. We only calculate the energy consumed in the synaptic circuit of Fig. 7 corresponding to all the synapses in the SNN (Fig. 1), and not the neuron circuits. However we account for both the energy dissipated in the domain wall devices and the rest of the transistor based circuit with them that enable STDP based synaptic functionality (Fig. 7).

C1 is charged instantaneously to V_1 = 0.7 V (Fig. 7) and then discharged steadily every time a pre-neuron spikes, as explained in Section III. Hence $\frac{1}{2}C_1V_1^2$ is dissipated for charging and $\frac{1}{2}C_1V_1^2$ is dissipated for discharging in each of the $N_2 = 400$ synapse circuits connected to each pre-neuron. Similarly, C2 is charged instantaneously to V_2 = 1.1 V (Fig. 7) and discharged steadily through T5 every time a post-neuron spikes. Hence $\frac{1}{2}C_2V_2^2$ is dissipated for charging and $\frac{1}{2}C_2V_2^2$ is dissipated for discharging in each of the $N_1 = 784$ synapse circuits connected to each post-neuron. In addition, for the synapse connecting post-neuron j with pre-neuron i, the energy dissipated in the domain wall device due to current flow ($I_{j,i}^{write}$) that causes the weight update, along with energy dissipated in transistors through which the same current flows (T3 and T4 for positive weight update, T7 and T8 for negative weight update) can together be written as $V_{DD}|I_{j,i}^{write}|\Delta t_{pulse}$ where V_{DD} =1.5 V and t_{pulse} is the duration of each current pulse (3 ns, as used in our micromagnetic simulations- Fig. 5(b)). $I_{j,i}^{write}$ for each synapse can be obtained for the weight update ($\Delta w_{j,i}$) in the synapse given by equation (4), using the relationship between weight and conductance, and then conductance and current (Fig. 5(b)).

Thus for each epoch, the total energy dissipated in the STDP circuit (E_{epoch}) is given by:

$$\begin{aligned} E_{epoch} &= \Sigma_{i=1}^{i=N_1}\Sigma_{all\ spikes\ of\ pre-neuron\ i}C_1V_1^2N_2 + \\ &\quad \Sigma_{j=1}^{j=N_2}\Sigma_{all\ spikes\ of\ post-neuron\ j}C_2V_2^2N_1 \\ &\quad + \Sigma_{i=1}^{i=N_1}\Sigma_{j=1}^{j=N_2}\Sigma_{all\ spikes\ of\ pre-neuron\ i} \\ &\quad \Sigma_{all\ spikes\ of\ post-neuron\ j}|I_{j,i}^{write}|V_{DD}\Delta t_{pulse} \\ &= \Sigma_{i=1}^{i=N_1}C_1V_1^2N_2n_i^{pre} + \Sigma_{j=1}^{j=N_2}C_2V_2^2N_1n_j^{post} \\ &\quad + \Sigma_{i=1}^{i=N_1}\Sigma_{j=1}^{j=N_2}\Sigma_{all\ spikes\ of\ pre-neuron\ i} \\ &\quad \Sigma_{all\ spikes\ of\ post-neuron\ j}|I_{j,i}^{write}|V_{DD}\Delta t_{pulse} \end{aligned} \tag{5}$$

where n_i^{pre} is the total number of times the pre-neuron i spikes per epoch and n_j^{post} is the total number of times the post-neuron j spikes per epoch.

Table I lists the training accuracy, test accuracy, total number of pre-neuron spikes, total number of post-neuron spikes, time taken for training, energy consumed in training (given by equation (5) above) and power consumed in training both for the case of completely unsupervised learning and partially supervised learning on the MNIST dataset (1000

Mode of learning	Number of epochs	Train accuracy (%)	Test accuracy (%)	Total number of pre-neuron spikes	Total number of post-neuron spikes	Total energy for learning	Total time for learning	Total power for learning
a) Completely unsupervised	1	86	67	≈ 320500	≈ 12200	≈ 0.5 mJ	≈ 0.1 s	≈ 5 mW
b) Partially supervised	2	86	61	≈ 641000	≈ 27000	≈ 1 mJ	≈ 0.2 s	≈ 5 mW

TABLE I
ENERGY CONSUMPTION AND SPEED METRICS FOR TRAINING THE SPINTRONIC HARDWARE SNN

train samples, 100 test samples). Only 1 epoch is used in the case of unsupervised learning and 2 epochs are used for partially supervised learning because reasonable train and test accuracies are achieved by then (Fig. 2).

Energy consumed per epoch is almost the same for completely unsupervised and partially supervised learning. It is to be noted that earlier reports [24], [43] only consider energy dissipated in the spintronic devices of the synapse circuits during the training. So the value of total energy for learning reported in [24], [43] is much lower than what we report in Table I.

V. CONCLUSION

Thus in this paper we have implemented training of analog hardware SNN, based on STDP enabled spintronic synapses and neurons, on the MNIST dataset of handwritten digits. We have used two modes of learning and obtained high classification accuracies. We have also reported the net energy consumed for learning in the spintronic devices and associated transistor based circuits that enable synaptic functionality.

ACKNOWLEDGMENT

Debanjan Bhowmik would like to thank Department of Science and Technology (DST), India, and Science and Engineering Research Board (SERB), India, for funding this research through INSPIRE Faculty Award and Early Career Research (ECR) Award respectively.

REFERENCES

[1] M. Davies, Nature Machine Intelligence, vol. 1, 386-388, 2019.
[2] M. Bouvier, A. Valentian, T. Mesquida, F. Rummens, M. Reyboz *et al.*, ACM Journal on Emerging Technologies in Computing Systems 15, 2 (2019).
[3] C. S. Thakur, J. L. Molin, G. Cauewenberghs, G. Indiveri, K. Kumar *et al.*, Frontiers in Neuroscience 12, 891 (2018).
[4] W. Maass, Neural Networks 10, 9 (1997).
[5] W. Maass, Proceedings of the IEEE 103, 12 (2015).
[6] P. U. Diehl and M. Cook, Front. Comput. Neurosci. 9, 99 (2015).
[7] G. Q. Bi and M. M. Poo, J. Neurosci. 18, 10464–10472 (1998).
[8] P. Dayan and L. F. Abbott, Chapter 5, The MIT Press (2005).
[9] W. Zhang and D. J. Linden, Nature Reviews Neuroscience 4, 885–900 (2003).
[10] K. Roy, A. Jaiswal and P. Panda, Nature, vol. 475, 607-617, 2019.
[11] A. Sengupta, and K. Roy, Applied Physics Reviews 4, 041105 (2017).
[12] A. Sengupta, A. Ankit and K. Roy, Proceedings in 2017 International Joint Conference on Neural Networks (IJCNN), 2017
[13] A. Sengupta, Y. Ye, R. Wang, C. Liu and K. Roy, Frontiers in Neuroscience 13, 95 (2019).
[14] P. Diehl, D. Neil, J. Binas, M. Cook, S. Liu et al. Proceedings in IEEE International Joint Conference on Neural Networks (IJCNN) (2015).
[15] P. A. Merolla *et al.*, Science vol. 345, no. 6197, pp. 668-697, 2014.
[16] S.B. Furber *et al.*, Proceedings of the IEEE, vol. 102, no. 5, pp. 652-665, 2014.
[17] M. Davies *et al.*, IEEE Micro., pp. 82-89, 2018.
[18] N. Hassan, X. Hu, L. Jiang-Wei, W. H. Brigner, O. G. Akinola *et al.* Journal of Applied Physics 124, 152127 (2018).
[19] O. Akinola, X. Hu, C. H Bennett, M. Marinella, J. S Friedman *et al.* Journal of Physics D: Applied Physics 52, 49LT01 (2019).
[20] C. H. Bennett, N. Hassan, X. Hu, J. A. C. Incornvia, J. S. Friedman *et al.* Proc. SPIE 11090, Spintronics XII, 110903I (2019).
[21] A. Sengupta, A. Banerjee, and K. Roy, Phys. Rev. Appl. 6, 064003 (2016).
[22] T. Bhattacharya, S. Li, H. Yangki *et al.* IEEE Access 7, 5034 (2019).
[23] K. Yue, Y. Liu, R. K. Lake and A. C. Parker, Science Advances 5, eaau8170 (2019).
[24] U. Sahu, A. Pandey, K. Goyal and D. Bhowmik, AIP Advances 9, 125339 (2019).
[25] H. Tsai *et al.*, J. Phys. D: Appl. Phys. vol. 51, no. 283001, 2018.
[26] G.W.Burr *et al.* in Proc. IEEE IEDM, pp. 29.5.1–29.5.4, 2014
[27] A. Sebastian, M. L. Gallo, G. W. Burr, S. Kim, M. BrightSky and E. Eleftheriou, Journal of Applied Physics vol. 124 no. 111101, 2018.
[28] A. Sengupta and K. Roy, *IEEE Transactions on Circuits and Systems-I*, vol. 63, no. 12, 2016.
[29] I. Boybat *et al.*, Nature Communications, vol. 9, no. 2514, 2018.
[30] C. Li *et al.*, Nature Comm., vol. 9, no. 2385, 2018.
[31] F. Cai *et al.*, Nature Electronics, vol. 2, pp. 290-299, 2019.
[32] M. Suri, O. Bichler, D. Querlioz, O. Cueto, L. Perniola et al., IEEE International Electron Devices Meeting, 4.4.1 (2011).
[33] M. Suri ed., Cognitive Systems Monographs, Springer, 2017.
[34] U. Saxena, D. Kaushik, M. Bansal, U. Sahu and D. Bhowmik, IEEE Transactions on Magnetics 54, 11 (2018).
[35] D. Bhowmik *et al.*, J Magn Magn Mater., vol. 489, no. 165434 (2019).
[36] Y. LeCun, Y. Bengio and G. Hinton, "Deep Learning",*Nature*, vol. 521, pp 436-444 (2015).
[37] G.W. Burr, R.M. Shelby, S. Sidler, C.D. Nolfo, J. Jang et al., IEEE Transactions on Electron Devices 62, 3498 (2015).
[38] P.Y. Chen, X. Peng, and S. Yu, IEEE International Electron Devices Meeting, 6.1.1 (2017).
[39] T. Gokmen and Y. Vlasov, Frontiers in Neuroscience 10, 333 (2016).
[40] D. Kaushik, U. Singh, U. Sahu, I. Sreedevi and D. Bhowmik, arXiv:1910.12919 (2019).
[41] A. Biswas, S. Prasad, S. Lashkare and U. Ganguly, arXiv:1612.02233 (2016).
[42] S. Prasad, A. Biswas, A. Shukla and U Ganguly. International Conference on Artificial Neural Networks (ICANN) (2017).
[43] U. Sahu, K. Goyal, U. Saxena, T. Chavan, U. Ganguly and D. Bhowmik. IEEE International Conference on Emerging Electronics, 1-6 (2018).
[44] A. Sengupta, Y. Shim and K. Roy, IEEE Transactions on Biomedical Circuits and Systems 10, 1152 (2016).
[45] S. Zhang, S. Luo, N. Xu, Q. Zou, M. Song et al., Advanced Electronic Materials 5, 1800782 (2019).
[46] S. Emori, U. Bauer, S.M. Ahn, E. Martinez and G.S.D. Beach, Nature Materials 12, 611 (2013).
[47] K.S. Ryu, L. Thomas, S.H. Yang and S. Parkin, Nature Nanotechnology 8, 527 (2013).
[48] D. Bhowmik, M.E. Nowakowski, L. You, O. Lee, D. Keating et al., Scientific Reports 5, 11823, (2015).
[49] I.M. Miron, T. Moore, H. Szambolics, L.D. Buda-Prejbeanu, S. Auffret et al., Nature materials 10, 419 (2011).
[50] S. Emori, E. Martinez, K.J. Lee, H.W. Lee, U. Bauer et al., Physical Review B 90, 184427 (2014).
[51] E. Fullerton and J. Childress. Proceedings of the IEEE 104, 10, 1787-1795 (2016).
[52] D. Apalkov, B. Dieny and J. M. Slaughter. Proceedings of the IEEE 104, 10, 1796-1830 (2016).
[53] A. Vansteenkiste, J. Leliaert, M. Dvornik, M. Helsen, F. Garcia-Sanchez et al., AIP Advances 4, 107133 (2014).
[54] C. Hu, "Modern Semiconductor Devices", Pearson (2009).