Orthogonal Spin Current Injected Magnetic Tunnel Junction for Convolutional Neural Networks

Venkatesh Vadde®, Bhaskaran Muralidharan®, Member, IEEE, and Abhishek Sharma®

Abstract—We propose that a spin Hall effect (SHE) driven magnetic tunnel junction (MTJ) device can be engineered to provide a continuous change in the resistance across it when injected with orthogonal spin currents. Using this concept, we develop a hybrid device-circuit simulation platform to design a network that realizes multiple functionalities of a convolutional neural network (CNN). At the atomistic level, we use the Keldysh nonequilibrium Green’s function (NEGF) technique that is coupled self-consistently with the stochastic Landau-Lifshitz-Gilbert–Slonczewski (LLGS) equations, which in turn is coupled with the HSPICE circuit simulator. We demonstrate the simultaneous functionality of the proposed network to evaluate the rectified linear unit (ReLU) and max-pooling functionalities. We present a detailed power and error analysis of the designed network against the thermal stability factor of the free ferromagnets (FMs). Our results show that there exists a nontrivial power-error trade-off in the proposed network, which enables an energy-efficient network design based on unstable free FMs with reliable outputs. The static power for the proposed ReLU circuit is 0.56 µW and whereas the energy cost of a nine-input ReLU-max-pooling network with an unstable free FM (Δ = 15) is 3.4 pJ in the worst case scenario. We also rationalize the magnetization stability of the proposed device by analyzing the vanishing torque gradient points.

Index Terms—Convolutional neural network (CNN), magnetic tunnel junction (MTJ), max-pooling, neuromorphic computing, rectified linear function.

I. INTRODUCTION

NEUROMORPHIC computing takes inspiration from biological brains to perform highly complex problems while consuming remarkably low energy [1], [2]. The brain performs in-memory computation and uses many low-precision calculations in parallel to perform a task. Meanwhile, modern computers are primarily based on the von Neumann architecture, which separates the computation and memory units and uses high-precision calculations [3]. Convolutional neural networks (CNNs) are a class of artificial neural networks (ANNs) [4] which produces excellent performance in machine learning problems dealing with image data [5], computer vision [6], [7], and natural language processing [8]. Binary neural networks (BNNs) and recurrent neural networks (RNNs) also deal with image data, with BNNs [9] being optimized for binary computations and RNNs [10] designed for sequential data processing, while CNNs are well-suited for image and video processing tasks.

A CNN has two stages: the first one is the feature extraction stage, and the second is the fully connected stage. Feature extraction is achieved by employing various layers, such as the convolutional layer, the activation function layer, and the pooling layer. These layers enable the CNN to exploit the spatial features of an image, introduce nonlinearity [11], and suppress noise [12].

The pooling layer is responsible for reducing the feature’s size, which helps reduce the parameters and computation in the network. There are two types of pooling: max pooling and average pooling. Max pooling calculates the maximum value from the portion of the image feature, while the average pooling finds the average value of the part of the image [4], [13]. The max-pooling also performs noise suppression by discarding noisy activations and is more often used in the network compared to the average pooling [12].

The activation function introduces nonlinearity to the network. Nonlinearity enables the network to learn complex structures in the data and differentiates between outputs [11]. Traditionally, sigmoid and tanh activation functions have been widely utilized. But, sigmoid and tanh functions saturate when the input is very high or low and are only sensitive to changes around their mid-points. After saturation, the network won’t be able to learn well [12], [14]. The sigmoid and tanh functions also face the vanishing gradient problem, where the gradient information used to learn networks vanishes for deep networks. Without any helpful gradient information, deep networks won’t be able to learn effectively [12]. The vanishing gradient and saturation problems faced by tanh and sigmoid
functions can be overcome by the rectified linear unit (ReLU) activation function [12].

The hardware implementation of various layers of the feature extraction stage of a CNN has been explored by a few studies [15], [16], [17], with the ultimate aim of developing neuromorphic computing [2], [18]. These works are based on CMOS circuit realizations of only the activation function [15], [16], [17] and lack the concatenation ability to perform simultaneous max-pooling. Furthermore, the in-memory computation using the existing CMOS technology is severely limited by area and energy requirements [3], [19]. Spintronics, on the other hand, provides a wide range of devices and physical effects for in-memory computing that suits the hardware realization of neuromorphic computing and enables the paradigm of “let the physics do the computing” [20]. Although the spintronic implementation of activation functions is studied [21], [22], [23], [24] but the implementation of max-pooling is still elusive, and moreover, some of these works require an external magnetic field to obtain the activation function. We propose a technologically relevant spintronic implementation of ReLU and max-pooling network using orthogonal spin current injection in magnetic tunnel junction (MTJ) device and show that unstable ferromagnets (FMs) can be used for energy-efficient design employing an atom-to-circuit approach that weaves quantum transport modeling with CMOS circuit design.

Most of the works centered around hardware realization of neuromorphic computing are implementing a part of the network on hardware [25], [26], [27], [28], [29], [30], [31] (like synapses based on domain wall, MTJ, nano-oscillators) and employ software-implementation (for activation function and pooling layers) to complete the network. The lack of spintronic hardware implementation of the activation function and pooling layers is a ramification of the absence of continuous change in the resistance of the MTJ device using spin current. Our work fills this critical gap by providing hardware for activation and max-pooling functions, thus enabling a fully-hardware implementation of neuromorphic networks.

The article is organized as follows. In Section II, we describe the basics of the ReLU and the max-pooling functionalities. In Section III, we discuss in detail our developed hybrid simulation platform based on Keldysh nonequilibrium Green’s function (NEGF) formalism coupled with the Landau-Lifshitz-Gilbert-Slonczewski (LLGS) equations for quantum transport description at the device level, which is then interfaced with the HSPICE circuit simulator to capture the spin and charge current interplay along with CMOS devices. In Section IV we present the performance of the ReLU and max-pooling circuit in terms of power consumption and noise analysis. We show that there exists a nontrivial power-error trade-off in the proposed ReLU-max-pooling network, which enables an energy-efficient network design based on unstable free FMs with reliable outputs. We conclude in Section V.

II. DESIGN

A. ReLU Circuit

The ReLU function emulation requires a device with linear characteristics. We propose a spin Hall effect (SHE)-driven MTJ device with orthogonal currents, as shown in Fig. 1(a), to generate a linear and continuous rotation in magnetization of the free FM layer. In this work, we have used typical CoFeB-based FMs for the fixed and free layers with the equilibrium magnetizations along the $\hat{y}$-direction and the
linear response in the free FM magnetization. The constant orthogonal current that is required to obtain a linear output of the ReLU circuit is shown in Fig. 1(b). The voltage divider drives a CMOS inverter pair connected to a resistor to form a voltage divider, as shown in Fig. 1(c). The voltage divider drives a CMOS inverter pair that operates in the linear region to produce the ReLU output. These spin currents interact with the free FM layer to rotate the magnetization to the in-plane direction. The continuous rotation in magnetization can be translated to the electrical signal using the tunnel magnetoresistance (TMR) of the MTJ device. In order to achieve the ReLU functionality, the SHE-driven MTJ is connected to a resistor to form a voltage divider, as shown in Fig. 1(b). The voltage divider drives a CMOS inverter pair that operates in the linear region to produce the ReLU output. The current drawn depends on the output of other ReLU circuits. The magnitude of the current drawn depends on the output of other ReLU circuits, which may not give the desired linear rotation (see [34, Appendix]).

### B. Max-Pooling

Max-pooling forms a crucial layer for the CNN, and we discuss in this section that the proposed ReLU circuit can be appropriately concatenated to perform the simultaneous local max-pooling function. The max-pooling function calculates the maximum of the inputs presented [4]. We introduce a competition among the ReLU circuits to achieve max-pooling with only one winner. We present a strategy that makes the current input to all the ReLU circuits (except one) less than zero to have one winner. We will achieve this by drawing current from the input of the ReLU circuits. The magnitude of the current drawn depends on the output of the ReLU circuits, thus enabling competition. At the end of the operation, there will be only one ReLU circuit with nonzero output. This ReLU output injects a negative current to all other ReLU devices so that their effective input remains less than zero. We have used an MTJ with a moderate TMR to showcase that a high TMR is not necessary for the design of the ReLU-Max pooling network. Any variations in the TMR value can be mitigated by adjusting the value of $R_1$ in conjunction with the CMOS inverter to ensure that the ReLU-Max pooling functions remain reliable without significantly impacting the network’s performance.

Our design enables the competition through an nMOS transistor connected to a resistor. The gate terminal of the transistor connects to the output of a different ReLU circuit. Thus, the current drawn from one ReLU circuit depends on the output voltage of another ReLU circuit. The gate terminal of the transistor connects to the output of a different ReLU circuit. This circuit design calculates both the ReLU and max-pooling functions simultaneously. This strategy makes all outputs zero other than the one corresponding to the maximum input. We can sum up all the outputs and get the correct result without wondering which device has the maximum input.

Crossbar arrays [40], [41] can be used to calculate the convolution function, and the output of the convolution layer is given as input to the proposed simultaneous ReLU-max-pooling circuit to complete the feature extraction stage in the CNN as shown in [34, Appendix]. Nonvolatile memory devices such as domain-wall [3], [42] and magnetic skyrmion [3], [43] can be used as synapses in the crossbar arrays. The input resistance of the proposed network can be tuned (by changing the dimensions of the SHE layer) as per the loading requirement of the crossbar array. The proposed ReLU circuit can be directly connected to the crossbar array without the max-pooling portion to complete the McCulloch-Pitts neuron [44] that can be used in hidden layers of feed-forward networks [45], [46] and reservoir computing [47].

### III. Simulation Methods

We show in Fig. 2 the schematic overview of the developed hybrid NEGF-CMOS simulation framework. The MTJ parameters, such as the effective mass of electrons in the insulator, the barrier height of the insulator, thickness, area, the resistance value of $R_1$, and the resistance of the MTJ, are presented to the NEGF simulator as shown in [34, Appendix]. This effect is substantial, and using a simple conductance equation to describe the

Fig. 2. Schematic overview of the simulation setup. (a) Various parameters of the MTJ and the resistor $R_1$ are given as input to the NEGF simulator. (b) MTJ resistance is calculated using the NEGF framework, and the voltage across the MTJ is evaluated from the voltage divider self-consistently. (c) Schematic of the dependence of the resistance of the MTJ with the FM polarization angle is shown. (d) NEGF results are incorporated into the HSPICE using VerilogA, and the entire circuit is simulated, including the LLGS equation.
MTJ ignores this effect. NEGF, on the other hand, includes these effects via the atomistic level simulation whose results agree with the experimental data [48], [50], [51]. In circuits employing MTJ, there is usually a change in MTJ voltage. The combination of these effects stresses the necessity for using NEGF self-coupled with the circuit simulation to capture the resistance dependence on the voltage. While simulating CMOS devices such as inverters, HSPICE is very useful, and integrating NEGF with HSPICE provides a pathway to design circuits with spintronic and CMOS devices. This integration also allows access to HSPICE’s analysis tools, such as noise and frequency analysis. This coupling of NEGF and HSPICE is further helpful in simulating more extensive networks. The NEGF simulator calculates the resistance of the MTJ for a given MTJ orientation angle and the voltage across it. The voltage across the MTJ depends on the voltage divider formed by the MTJ and resistor $R_1$, so the NEGF and voltage divider are simulated self-consistently as shown in Fig. 3.

This result is coupled to the HSPICE simulator using VerilogA. The HSPICE simulates the entire circuit, including the LLGS equation [52], [53], and the HSPICE also simulates the approximation of the CMOS inverter pair based on a 16 nm node of the predictive technology model (PTM) [54]. The circuit parameters used in the simulation are given in Table I. We have employed The Keldysh NEGF formalism [55], [56] coupled with LLGS equations [57], [58], [59] to describe the magnetization dynamics of the free FM as shown in Appendix A (see [34, Appendix] for more details on the simulation methods).

IV. RESULTS

We show in Fig. 4(a) the continuous rotation of the magnetization of the free FM layer with the input current $I_{in}$, the stability of the magnetization is discussed in Appendix C. The continuous rotation of the magnetization with $I_{in}$ is achieved when the orthogonal current $I_{cy}$ magnitude is larger than a critical value (see [34, Appendix]). The TMR effect translates the continuous change in the magnetization to a continuous resistance change of the MTJ device, as shown in Fig. 4(a). The linear region of the MTJ-resistance around the zero $I_{in}$ has been utilized to obtain the ReLU output as shown in Fig. 4(b) by injecting an appropriate bias current $I_b$. Various parameters of the ReLU circuit design are given in Table I. The proposed circuit shown in Fig. 1(b) emulates the ReLU function closely for normalized inputs smaller than unity.

We evaluate the performance of the ReLU circuit against the thermal stability factor ($\Delta = [(H_1 M_s V)/(2k_B T)]$ of the free FM layer of the MTJ device. The $\Delta$ factor of the free-FM not only captures the stability of the magnetization direction against thermal noise but also the extent to which the spin current changes the magnetization direction. The critical spin current ($I_{sc} = [4eek_BT^2]/\hbar$) for magnetization switching is proportional to $\Delta$ [32]. We vary the $\Delta$ of the free FM layer by changing the anisotropy field, but the same can also be changed by the MTJ device dimensions. The critical orthogonal current ($I_{cy}$), bias current ($I_b$), and the input current ($I_{in}$) required for a continuous rotation of magnetization decreases with a reduction in $\Delta$. Thus, the mean static power consumption of the ReLU circuit also decreases with a reduction in $\Delta$ as shown in Fig. 5(a). We also observe that the $\Delta$ reduction does not affect the output settling time ($\approx4$ ns) as various input currents decrease proportionally with $\Delta$.

We show in Fig. 5(a) the reduction in the output error of the ReLU circuit with a corresponding increase in $\Delta$. The thermal noise ($\langle H_k^2 \rangle = [2eek_BT]/\gamma M_s V$) stays constant as $\Delta$ is increased and the weight of the thermal noise decreases with $H_{eff}$ as $H_k$ is increased since $H_{eff}$ also includes the
TABLE I
SIMULATION PARAMETERS

| Symbol | Quantity                        | Value                              |
|--------|--------------------------------|-----------------------------------|
| $M_s$  | saturation magnetization       | 1150 emu/cm$^3$ [32], [37]        |
| $H_k$  | anisotropy field               | 330 - 3300 Oe [38]                |
| $V$    | volume of ferromagnet          | 1000 nm$^3$                        |
| $\Delta$ | thermal stability factor     | 4.58 - 45                           |
| $\alpha$ | Gilbert damping                | 0.01 [32], [35]                    |
| $L$    | length of ferromagnet          | 40 nm                              |
| $W$    | width of ferromagnet           | 25 nm                              |
| $C_{MTJ}$ | MTT capacitance               | 26.56 $\mu$F                        |
| TMR    | Tunnel magnetoresistance       | 89.16% at 5 mV                     |
| $I_{in}$ | input current to SHE layer    | 35 - 340 $\mu$A                    |
| $\delta$ | spin-hall angle                 | 0.3 rad [39]                       |
| $d_{HM}$ | thickness of heavy metal      | 5 nm [39]                          |
| $R_1$  | reference resistor             | 11.12 M$\Omega$                    |
| $R_2$  | resistance in NMOS current source | 1 - 7 $k\Omega$                  |
| $I_b$  | biasing current                | 17 - 160 $\mu$A                    |
| $V_{DD}, V_{SS}$ | voltage sources       | 0.5 V, -0.5 V                      |
| $C_{G}$ | CMOS inverter input capacitance | 0.175 $\mu$F                       |
| $C_{O}$ | CMOS inverter output capacitance | 0.305 $\mu$F                      |
| $\Delta t$ | simulation time step          | 9.5 ps                              |
| $\hbar$ | reduced Planck's constant      | $1.055 \times 10^{-34}$ J s        |
| $k_B$  | Boltzmann constant            | $1.38 \times 10^{-16}$ erg K$^{-1}$ |
| $T$    | temperature                    | 300 K                               |

contribution from the anisotropy field $H_k$. The error in the output is estimated by performing fifty cycles of Monte Carlo simulations [60] with a normalized input current ($I_{in}/I_0$) of value 0.5. The decrease in power consumption and the increase in the error due to the reduction in $\Delta$ presents an opportunity to optimize the circuit to consume less power while keeping the error in an acceptable range. It can be inferred from Fig. 5(a) that the ReLU circuit has an optimal performance of power (0.5–1.2 mW) and error (<2.2%) with $\Delta$ in the range of 15 – 25. Fig. 5(b) shows the boxplot of the error (%) with the $\Delta$. The boxplot [61] gives profound insights into the error statistics and uses the following five parameters to summarize the data error. The medians (the red lines) converging to zero indicate that the average thermal noise is zero. The reduction in its spread is associated with the variance of the thermal noise since its effect on free FM depends on $\Delta$.

We show in Fig. 6(a) the transient response of the ReLU-max-pooling circuit with nine inputs corresponding to the typically used $3 \times 3$ pooling layer for the worst case scenario. In the worst case scenario, all the inputs are close to the maximum possible value and the observed settling time of the output is 9 ns. Initially, all nine outputs rise to reach the output dictated by the corresponding inputs. Then quickly, the competition starts, and all the ReLU circuits inject negative currents into each other through the nMOS transistors causing the outputs to decrease. As the competition progresses, one of the outputs (that has the maximum input) becomes the winner and reaches its steady-state value corresponding to its input while keeping all the other outputs at zero. It can also be inferred from Fig. 6(a) that the sum of all the outputs is equal to the maximum output in the steady-state, and it performs the max-pooling without any knowledge about which of the nine-ReLU devices is producing it.

Fig. 6(b) shows the ReLU-max-pooling network results for 500 Monte Carlo simulations. The value of the input currents is varied using the Monte Carlo method. The inputs are selected such that the maximum input is varied over the entire input range. The results from Fig. 6(b) closely resemble the ReLU function while simultaneously calculating the max pooling.

We show in Fig. 7(a) the ReLU-max-pooling network’s energy consumption to reach the steady state in the worst case scenario. The worst case energy consumption decreases as $\Delta$ decreases. This decrease in energy is due to the reduction of energy requirement for each ReLU circuit and the reduction in the competition circuit energy that is caused by a decrease in the nMOS device dimensions and resistance $R_2$ due to the decrease in $I_{in}$ required.

Fig. 7(b) shows the static power consumption and percentage error of the network. The static power consumption decreases, and the percentage error increases with a reduction in $\Delta$. This behavior shares the same rationale as that of a single ReLU device discussed earlier. As $\Delta$ increases, there is a linear increase in power consumption and a sharp fall in error (%). It opens up a possibility for the optimization of the
The above discussion suggests that a room temperature unstable free FM (Δ < 40) based MTJ can be utilized to design an energy-efficient ReLU-max-pooling network. We show in Fig. 7(c) the boxplot of the error percentage of the nine-input network. Here the medians converge to zero, indicating that the average thermal noise is zero. As Δ increases, the effect of thermal noise on the free FM decreases, causing a decline in the spread of the boxplot.

V. CONCLUSION

In this article, we proposed a circuit design for calculating crucial neuromorphic functions (ReLU and max-pooling) based on a continuous rotation of magnetization of the free FM of an MTJ through orthogonal spin current injection. Using our developed simulation platform based on NEGF and LLGS equations coupled with the HSPICE circuit simulator, we showed the optimal range of performance (power and error) against the thermal stability factor of the free FM of the MTJ device. We demonstrated that the designed circuit is robust against thermal noise, while consuming 0.56 μW of power for ReLU functionality and requires 3.4 pJ of energy for nine-input ReLU-max-pooling computation in the worst case scenario, and consumes 120 μW of static power in a typical scenario with a percentage error of 2.3% at Δ = 15. Our work also opens the possibility of using an unstable FM with reliable output.

Appendix

A. Simulation Method (Magnetization Dynamics)

We utilize the LLGS equation [57], [58] to describe the magnetization dynamics of the free FM. The LLGS equation is given by

\[
\left(1 + \alpha^2 \right) \frac{d\hat{m}}{dt} = -\hat{m} \times \vec{H}_{\text{eff}} - \alpha \hat{m} \times \hat{m} \times \vec{H}_{\text{eff}}
\]

where \( \hat{m} \) is the unit vector along the direction of magnetization of the free magnet, \( \gamma \) is the gyromagnetic ratio, \( \alpha \) is the Gilbert damping parameter, \( \vec{H}_{\text{eff}} = (\vec{H}_{\text{eff}}/H_k) \) is the reduced effective field and \( \vec{i}_s = (\vec{H}_{\text{eff}}/(2qM_sVH_k)) \) is the normalized spin current. The term \( \vec{H}_{\text{eff}} \) includes the contribution of the anisotropy field (\( H_k \)), the applied magnetic field (\( H_{\text{app}} \)) and the thermal noise (\( H_{\text{th}} \)). The thermal noise is given by \( (H_{\text{th}}^2) = (2k_B T)/(\gamma M_s V) \) and \( \langle \rangle \) represents the ensemble average [62]. The spin current from the SHE layer is given by Hirsch [63] and Takahashi and Maekawa [64]

\[
I_s = \theta \frac{I_{\text{FM}}}{t_{\text{HM}}} I_c \times \sigma
\]

where \( I_s \) is the magnitude of the spin current, \( \theta \) is the spin Hall angle, \( I_{\text{FM}} \) is the length of the FM, \( t_{\text{HM}} \) is the thickness of the SHE layer, \( I_c \) is the charge current, and \( \sigma \) is the polarization of the spin current.

B. Micromagnetic Simulation

Fig. 8 shows the micromagnetic simulation of the free-FM of the MTJ using the LLGS equation with an applied spin current of 510 μA with x-polarization and 400 μA with y-polarization. The settling magnetization of the FM in (a) \( m_x \) magnetization, (b) \( m_y \) magnetization, and (c) \( m_z \) magnetization. This shows that the FM is mono-domain FM.

C. Stability of Magnetization

In this section, we explore different ways to validate the stability of magnetization.

1) LLGS Simulation: The stability of the magnetization in the presence of the spin current can be analyzed via evaluation of the magnetization from the different initial configurations.
We show in Fig. 9(a) and (b) the magnetization dynamics of the FM in the presence of orthogonal spin currents 510, 400 μA with x- and y-polarizations, respectively, producing a stable magnetization orientation of $m_x = 0.7869$, $m_y = 0.6171$, and $m_z = 0.00$. It can be observed [Fig. 9(a) and (b)] the different initial magnetization of the FM stabilizes to the same final value for given spin current inputs.

2) Net Torque on the FM: The stability of magnetization can be analyzed from the plots of torque versus polar angle ($\theta$) and azimuthal angle ($\phi$) of the magnetization.

The net torque on the free FM is given by

$$\vec{\tau} = -\hat{m} \times \vec{H}_{\text{eff}} - \alpha \hat{m} \times \hat{m} \times \vec{H}_{\text{eff}} - \beta \hat{m} \times \hat{m} \times \vec{I}_t + \alpha \beta \hat{m} \times \vec{I}_s.$$

Here, $\beta = (\hbar/(2qM_sV))$. The net torque ($\tau$) can be decomposed into in-plane torque ($\tau_{\text{ip}}$) and out-of-plane torque ($\tau_{\text{op}}$). The $\tau_{\text{ip}}$ can be damping (positive) or anti-damping (negative). The $\tau_{\text{ip}}$ is given by

$$\tau_{\text{ip}} = \hat{i}_p \cdot \vec{\tau},$$

where $\hat{i}_p$ is given by

$$\hat{i}_p = \frac{-\hat{m} \times \hat{m} \times \hat{m}_v}{| -\hat{m} \times \hat{m} \times \hat{m}_v |}.$$

Here, $\hat{m}_v$ is the vanishing torque point ($\vec{\tau} = 0$) [66], which can be extracted from plots of torque versus $\theta$ and $\phi$ of the magnetization. If the $\tau_{\text{ip}}$ is damping (positive), that means the $\hat{m}_v$ is a stable point, and if it is anti-damping (negative), then $\hat{m}_v$ is not a stable point in the presence of the applied spin current.

The stability of the magnetization considered in this article can be explored using the above method. The spin currents of 510 and 600 μA with x- and y-polarization, respectively, are injected into the PMA free-FM, and the expected stable point for this condition is $m = [0.6477 0.7619 0]$. We show in Fig. 10(a) the net torque on the free-FM. It can be seen that net torque vanishes at two points ($m_1 = [0.6476 0.7619 0]$ and $m_2 = [-0.6476 -0.7619 0]$), and both the torque vanishing points appear at $\theta = \pi/2$, these points can be clearly seen from Fig. 10(b).

We show the in-plane torque in Fig. 10(c) to inspect the stability of these torque vanishing points. The $\tau_{\text{ip}}$ is always positive (damping) for $m_1$ implying the stable point, and always negative (anti-damping) for $m_2$, implying not a stable point in the presence of the orthogonal spin currents. We believe that the thermal stability of the magnetization may be correlated with the area under the in-plane torque curve, which will be addressed in our future works.

ACKNOWLEDGMENT

The author Venkatesh Vadde would like to thank Saumya Gupta for her help in micromagnetic simulations.

REFERENCES

[1] D. Marković, A. Mizrahi, D. Queloz, and J. Grollier, “Physics for neuromorphic computing,” Nature Rev. Phys., vol. 2, no. 9, pp. 499–510, Jul. 2020.
[2] C. D. Schuman et al., “A survey of neuromorphic computing and neural networks in hardware,” 2017, arXiv:1705.06963.
[3] I. Grollier, D. Queloz, K. Y. Camsari, K. Everschor-Sitte, S. Fukami, and M. D. Stiles, “Neuromorphic spintronics,” Nature Electron., vol. 3, no. 7, pp. 360–370, Mar. 2020.
[4] K. O’Shea and R. Nash, “An introduction to convolutional neural networks,” 2015, arXiv:1511.08458.
[5] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” 2014, arXiv:1409.1556.
[6] S. Khan, H. Rahmani, S. A. Shah, and M. Bennamoun, “A guide to convolutional neural networks for computer vision,” Synth. Lect. Comput. Vis., vol. 8, no. 1, pp. 1–207, Feb. 2018.
[7] L. Xie and A. Yuille, “Genetic CNN,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 1379–1388.
[8] S. Albawi, T. A. Mohammed, and S. Al-Zawi, “Understanding of a convolutional neural network,” in Proc. Int. Conf. Eng. Technol. (ICET), Aug. 2017, pp. 1–6.
[9] S. Shreya, G. Verma, S. N. Piramanayagam, and B. K. Kaushik, “Energy-efficient all-spin BNN using voltage-controlled spin-orbit torque device for digit recognition,” IEEE Trans. Electron Devices, vol. 68, no. 1, pp. 385–392, Jan. 2021.
[10] S. Sheikhhaa, S. Angizi, and R. F. DeMara, “Energy-efficient recurrent neural network with MRAM-based probabilistic activation functions,” IEEE Trans. Emerg. Topics Comput. Vis., early access, Sep. 9, 2022, doi: 10.1109/TETC.2022.3202112.
[11] S. Sharma, S. Sharma, and A. Athaiya, “Activation functions in neural networks,” Towards Data Sci., vol. 6, no. 12, pp. 310–316, 2017.
[12] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. Cambridge, MA, USA: MIT Press, 2016.
[13] Y.-L. Bourne, J. Ponce, and Y. LeCun, “A theoretical analysis of feature pooling in visual recognition,” in Proc. 27th Int. Conf. Mach. Learn. (ICML), 2010, pp. 111–118.
[14] X. Glorot, A. Bordes, and Y. Bengio, “Deep sparse rectifier neural networks,” in Proc. 14th Int. Conf. Artif. Intell. Statist., 2011, pp. 315–323.
[15] C.-H. Chang, H.-Y. Kao, and S.-H. Huang, “Hardware implementation for multiple activation functions,” in Proc. IEEE Int. Conf. Consum. Electron.-Taiwan (ICCE-TW), May 2019, pp. 1–2.
[16] C. Geng, Q. Sun, and S. Nakatake, “An analog CMOS implementation for multi-layer perceptron with ReLU activation,” in Proc. 9th Int. Conf. Mod. Circuits Syst. Technol. (MOCAST), Sep. 2020, pp. 1–6.
