Quantized Non-Volatile Nanomagnetic Domain Wall Synapse based Autoencoder for Efficient Unsupervised Network Anomaly Detection

Muhammad Sabbir Alam[1], Walid Al Misba[1], Jayasimha Atulasimha[1,2,3,4*]

[1] Dept. of Mechanical and Nuclear Engineering, Virginia Commonwealth University, Richmond, VA, USA
[2]Dept. of Electrical and Computer Engineering, Virginia Commonwealth University, Richmond, VA, USA
[3]Material Sciences Division, Lawrence Berkeley National Laboratory, Berkeley, CA, USA
[4]Dept. of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA, USA

[*]Email: jatulasimha@vcu.edu

Abstract: Anomaly detection in real-time using autoencoders implemented on edge devices is exceedingly challenging due to limited hardware, energy, and computational resources. We show that these limitations can be addressed by designing an autoencoder with low-resolution non-volatile memory-based synapses and employing an effective quantized neural network learning algorithm. We further propose nanoscale ferromagnetic racetracks with engineered notches hosting magnetic domain walls (DW) as exemplary non-volatile memory based autoencoder synapses, where limited state (5-state) synaptic weights are manipulated by spin orbit torque (SOT) current pulses to write different magnetoresistance states. The performance of anomaly detection of the proposed autoencoder model is evaluated on the NSL-KDD dataset. Limited resolution and DW device stochasticity aware training of the autoencoder is performed, which yields comparable anomaly detection performance to the autoencoder having floating-point precision weights. While the limited number of quantized states and the inherent stochastic nature of DW synaptic weights in nanoscale devices are typically known to negatively impact the performance, our hardware-aware training algorithm is shown to leverage these imperfect device characteristics to generate an improvement in anomaly detection accuracy (90.98%) compared to accuracy obtained with floating-point synaptic weights that are extremely memory intensive. Furthermore, our DW-based approach demonstrates a remarkable reduction of at least three orders of magnitude in weight updates during training compared to the floating-point approach, implying significant reduction in operation energy for our method. This work could stimulate the development of extremely energy efficient non-volatile multi-state synapse-based processors that can perform real-time training and inference on the edge with unsupervised data.

Keywords: Domain wall, NSL-KDD, anomaly detection, autoencoder, unsupervised learning, deep learning, neuromorphic computing, quantized weight.

1. Introduction

In today's interconnected world, the security and integrity of computer networks are of paramount importance. By 2030, it is estimated that 500 billion devices will be connected to the internet [1], with a significant portion comprising Internet of Things (IoT) devices. While the rapid growth of network-based devices, applications and services offer immense convenience, it also has led to an increasing number of cyber threats and attacks [2]. The proliferation of cybercrimes and network intrusions underscores the need for developing robust solutions that can safeguard network security. Anomaly detection plays a vital role in safeguarding these networks by identifying and mitigating abnormal or malicious activities that deviate from expected patterns [3]. Detecting such anomalies in real-time is crucial to prevent potential damage, data breaches, and service disruptions [4]. As we look towards the future, where the Internet of Things (IoT) and edge computing gain prominence, the need for efficient and effective anomaly detection becomes even more critical.

Autoencoders have emerged as a promising approach for anomaly detection in unlabeled network traffic samples [5]-[7]. These neural network architectures are capable of learning meaningful representations of data by training on unlabeled samples. The essence of an autoencoder lies in its ability to encode input data into a lower-dimensional latent space and then decode it back to reconstruct the original input. By leveraging the power of deep learning, autoencoders excel at capturing and representing complex patterns and structures in data, making them well-suited for unsupervised learning inspired anomaly detection tasks. However, implementing autoencoders for real-time anomaly detection on resource-constrained edge devices poses significant challenges. Edge devices, such as internet of things (IoT) devices and embedded systems, typically have limited hardware capabilities, computational resources, and energy constraints [8], [9]. These constraints hinder the deployment of sophisticated deep learning models like autoencoders on the edge, necessitating the development of novel approaches that can meet these challenges.

The autoencoder's synaptic weight precision can be reduced using quantization techniques [10], [11], which offers an effective solution for energy and computational resource-constraint environments, such as edge devices. By employing low-resolution weight representations instead of floating-point weights, quantization reduces memory footprint, energy consumption, and latency, thus enabling efficient implementation of neural networks [10], [11]. However, traditional post training quantization methods [10], which involve applying quantized weights directly to pre-trained models, may lead to a degradation in model accuracy due to information loss caused by weight rounding. Quantization-aware training aims to address this challenge by integrating the quantization process during the training phase itself. This approach ensures that the neural network learns to adapt to the reduced precision and achieve better performance when deployed in low-precision environments [12], [13].

Besides quantization, further reduction in computational resources can be achieved using in-memory computing [14], [15] platform, where the data can be stored and processed directly within the main memory, eliminating the need for frequent data transfers between memory and processor, as commonly observed in the von-Neumann computing paradigm [16]. Therefore, this approach leads to faster and more efficient data processing. Spintronic memory devices offer a promising solution for implementing in-memory neural networks because of their non-volatility, higher energy efficiency, faster speed, dense footprint, and CMOS compatibility [16]-[20]. For instance, spintronic magnetic domain wall (DW) devices [17], [18] possess all these desirable properties. However, due to small on/off ratio of magnetic tunnel junctions (MTJs), which can be at most 7:1 at room temperature [21], DW devices that encode information in the resistance of MTJs have drawbacks including inherent low-resolution as well as stochastic nature [22], [23]. These imperfect device characteristics typically have a negative impact on neural network accuracy. Nevertheless, many research studies have demonstrated that modified training algorithms can retain high accuracy in analog low-resolution device-based neural networks [24]-[26].

This study introduces a novel approach to efficient anomaly detection in edge nodes through the utilization of a low-resolution quantized DW synapse-based autoencoder. The autoencoder's synapses are designed using a ferromagnetic racetrack device hosting a magnetic DW and having engineered notches at a regular interval. The synaptic weights, limited to five states, are controlled by applying spin orbit torque (SOT) current pulses in the heavy metal layer underneath the magnetic racetrack. Five states are chosen to achieve an optimal balance between performance and hardware complexity. Extensive micromagnetic simulations are conducted in the presence of room temperature thermal noise to assess the stochastic variations in each memory state of the DW device. To enhance the autoencoder's performance, a hardware-aware training algorithm is employed, inspired by neural network training with low-precision weights. In this algorithm, weight gradients are accumulated in a separate high-precision memory before being quantized and programmed into the low-resolution devices in the analog domain. Our proposed method is tested on the

NSL-KDD [27] dataset, which is an exemplary dataset that has been widely used for evaluating the performance of intrusion detection methods. This dataset provides a realistic and challenging environment for assessing the efficacy of anomaly detection algorithms, allowing researchers to determine the robustness and generalizability of their proposed methods.

The contributions of our proposed method can be summarized as follows: Firstly, we explore the feasibility of utilizing unsupervised learning-based deep neural network training with quantization-aware training. To the best of our knowledge, quantization-aware training has not been extensively explored in the context of unsupervised learning. The results of our study demonstrate that training with 5-state synaptic weights yields superior performance compared to training with 2-state or 3-state synaptic weights. Secondly, we demonstrate that, by employing the effective hardware-aware training algorithm, our proposed autoencoder model achieves a high level of testing accuracy in anomaly detection, surpassing the testing accuracy obtained when using floating-point trained weights. The combination of quantization noise and device stochasticity acts as an effective regularization process, thereby improving the testing accuracy of the proposed autoencoder model. Furthermore, our DW-based approach showcases a significant reduction of at least three orders of magnitude in weight updates during training in contrast to the floating-point approach, indicating substantial energy conservation benefits inherent to our method. While our study focuses on the use of the DW device as an illustrative example, the insights gained from our research can be extended to other non-volatile multistate memory technologies. This exploration opens up avenues for implementing quantized autoencoders in anomaly detection, leading to improved efficiency and effectiveness.

The subsequent sections of the paper are structured as follows: Section 2 explores related research in both anomaly detection with autoencoders and use of DW devices as non-volatile memory for synaptic weights. Section 3 delves into the data and the data preprocessing steps. Section 4 provides comprehensive information regarding the proposed quantized autoencoder-based anomaly detection, including details on the autoencoder model, quantization-aware training process, anomaly detection workflow, and algorithm. Section 5 presents the design of the DW synapse and the micromagnetic simulations required for the DW synapse-based autoencoder. Section 6 investigates the performance of anomaly detection using the proposed method. Finally, Section 7 concludes the paper by discussing the outcomes of this study and outlining potential future directions.

2. Related work: Anomaly detection with autoencoders and nanoscale magnetic domain wall synapses

In recent years, the increasing adoption of machine learning approaches for anomaly detection has been driven by the limitations and high costs associated with conventional signature-based intrusion detection techniques [28]. These traditional methods prove inadequate in effectively detecting zero-day attacks, which are characterized by their unknown and previously unseen nature [29]. Various supervised learning-based classification algorithms and hybrid models combining multiple algorithms have been explored to identify network anomalies and detect attacks with high accuracy. Notable algorithms include Support Vector Machine (SVM), Decision Tree (DT), Naive Bayes Network (NB), Naive Bayes Tree (NBTree), J48, Fuzzy logic, and Artificial Neural Networks (ANN) [27], [30]-[32]. However, the effectiveness of these algorithms hinges on accurate labels and balanced training data [33]. The availability of such data, particularly in the realm of network intrusion detection, is limited due to factors like privacy concerns and data confidentiality [34]. To overcome this constraint, researchers have turned to unsupervised learning techniques, such as anomaly detection algorithms based on autoencoders [6], [7]. More recently, studies have been conducted on unsupervised deep learning circuits using memristors to enable real-time anomaly detection with autoencoders on low-power devices [35].

In the emerging field of spintronic memory devices, researchers have made significant advancements in leveraging their properties for efficient computing. It has been shown that non-volatile nanomagnetic devices can be controlled efficiently using voltage-controlled magnetic anisotropy (VCMA) [36]-[38], voltage-induced strain [39]-[43], current control [44], [45], and a combination of both current and voltage control [23], [46], [47]. Studies have demonstrated that, despite having imperfect device characteristics, nanomagnetic devices can be implemented as multistate synapses for deep neural networks [23], [24], [55]. However, for quantized neural network implementation, weight gradients need to be stored in high-precision memory during the training phase to retain high accuracy [11], [48]. In recent studies, it has been observed that by incorporating an effective quantization-aware training algorithm, stochastic and extremely low-resolution (less than 3-bit) DW devices can achieve comparable accuracy levels to floating-point precision neural networks [24].

While the feasibility and implementation of various deep neural network architectures using emerging spintronic memory devices are actively being studied, the implementation of autoencoders using nanomagnetic devices remains relatively unexplored. This research gap serves as the motivation to explore the connection between the fields of unsupervised autoencoder-based anomaly detection and spintronic nanomagnetic synapse technology.

3. Data and Preprocessing

In this section, we explore the characteristics of the NSL-KDD dataset and outline the preprocessing steps undertaken to prepare the data for further analysis.

3.1 NSL-KDD Data

The NSL-KDD dataset is derived from the KDD Cup 1999 dataset, which represents a comprehensive collection of network traffic data containing both normal and various attack instances [27]. Within the NSL-KDD dataset, two distinct sets of data exist: KDDTrain+ and KDDTest+. The training data (KDDTrain+) consists of 125,973 packets, each categorized into one of the 23 distinct data types. These types include malicious categories such as Ipsweep, Guess_passwd, Warezclient, Neptune, Multihop, Perl, Smurf, Phf, Rootkit, Imap, Loadmodule, Portsweep, Nmap, Back, Pod, Spy, Land, Warezmaster, Satan, Buffer_overflow, Teardrop, Ftp_write, as well as the category labeled Normal [49]. Among these, 58,630 packets in the KDDTrain+ dataset are labeled as malicious, while the remaining 67,343 packets represent the normal packets. As for the KDDTest+ dataset, it comprises a total of 22,544 packets, with 12,833 packets categorized as malicious and the remaining 9,711 packets labeled as normal.

TABLE 1
Features and Data Types of NSL-KDD Dataset

No	Attribute	Type	No	Attribute	Type
1	duration	int64	22	is_guest_login	int64
2	protocol_type	object	23	count	int64
3	service	object	24	srv_count	int64
4	flag	object	25	serror_rate	float64
5	src_bytes	int64	26	srv_serror_rate	float64
6	dst_bytes	int64	27	rerror_rate	float64
7	land	int64	28	srv_rerror_rate	float64
8	wrong_fragment	int64	29	same_srv_rate	float64
9	urgent	int64	30	diff_srv_rate	float64
10	hot	int64	31	srv_diff_host_rate	float64
11	num_failed_logins	int64	32	dst_host_count	int64
12	logged_in	int64	33	dst_host_srv_count	Int64

13	num_compromised	int64	34	dst_host_same_srv_rate	float64
14	root_shell	int64	35	dst_host_diff_srv_rate	float64
15	su_attempted	int64	36	dst_host_same_src_port_rate	float64
16	num_root	int64	37	dst_host_srv_diff_host_rate	float64
17	num_file_creations	int64	38	dst_host_serror_rate	float64
18	num_shells	int64	39	dst_host_srv_serror_rate	float64
19	num_access_files	int64	40	dst_host_rerror_rate	float64
20	num_outbound_cmds	int64	41	dst_host_srv_rerror_rate	float64
21	is_host_login	int64			

Each packet in the NSL-KDD dataset consists of 41 features that are identical to those in the KDD Cup 1999 dataset. Table 1 provides an overview of the 41 features along with their corresponding data types. The 42nd feature represents the data label (normal/attack) [49]. However, for training purposes, the packets do not include the data label. Among the selected features, 38 are numeric (int64/float64) and the remaining 3 are categorical (object) variables.

3.2 Preprocessing Steps

Prior to the training phase, preprocessing is applied to all packets in the dataset. First, the categorical features are converted into numerical representations. Specifically, the second position (protocol/type), third position (service), and fourth position (flag) contain categorical data. The three categorical features are modified to one-hot encoded numerical values. For instance, in the "protocol type" feature, the strings "tcp", "udp", and "icmp" are replaced with their respective one-hot encoded representations: [1 0 0], [0 1 0], and [0 0 1]. As a result, the three categorical features in the NSL-KDD dataset, namely "protocol type," "service," and "flag," which have 3, 70, and 11 distinct strings respectively, are transformed into a total of 84 features. This one-hot encoding process leads to a combined total of 122 features, including the selected original 38 numeric features.

Additionally, the dataset undergoes normalization to ensure consistency. Each feature vector is normalized by scaling them to fit in the range of [0; 1] based on the maximum value within that feature vector. All types of malicious data are labeled as "1," while normal data packets are labeled as "0".

4. Quantized Autoencoder based Anomaly Detection

4.1 Autoencoder

An autoencoder is a neural network architecture that employs unsupervised learning to reconstruct input data. Comprising multiple layers, including one or more hidden layers, the autoencoder maintains the same size for both the input and output layers. At the center of the network lies the bottleneck layer, which represents a compressed latent space representation of the input data. The encoder maps the input to the bottleneck layer representation, while the decoder reconstructs it in the output layer [7]. Fig. 1 illustrates the architecture of a standard autoencoder.

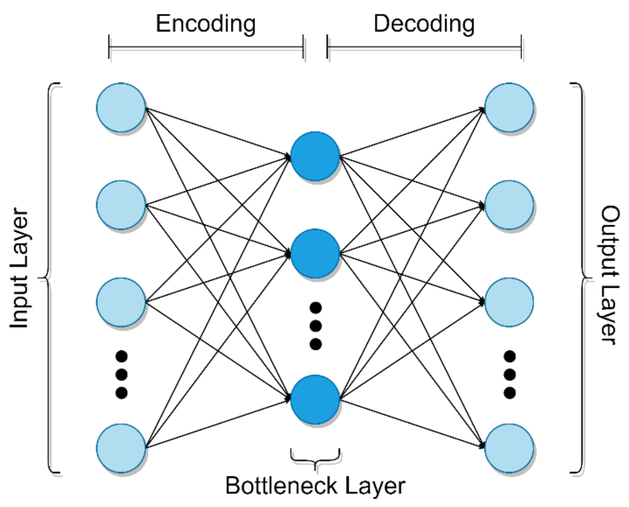

Figure 1: A standard autoencoder architecture.

In the encoding phase, an n dimensional input vector X $[x_1, x_2, x_3, \ldots, x_n]$ is mapped to hidden layer representation H. The encoding operation is expressed as equation (1):

$$H = F_1(W_1X + b_1) \tag{1}$$

where W_1 is the weight matrix, b_1 is the bias vector, and F_1 denotes the encoder activation function.

In the decoding phase, the latent space representation H is mapped to reconstruct the input (X) as output R $[r_1, r_2, r_3, \ldots, r_n]$. The decoding operation is expressed as equation (2):

$$R = F_2(W_2H + b_2) \tag{2}$$

where W_2 is the weight matrix, b_2 is the bias vector, and F_2 denotes the decoder activation function.

As the autoencoder goes through training with backpropagation, the weights and biases are updated to minimize the loss function (L). The loss function is used to minimize the reconstruction error. It can be expressed as equation (3):

$$L(\Theta) = \frac{1}{n}\sum_{i=1}^{n} \|X_i - R_i(\Theta)\|^2 \tag{3}$$

where Θ is the autoencoder parameters (weights and biases). Here, L represents a mean squared error (MSE) loss function.

The reconstruction error is used to determine whether a network traffic sample is normal or malicious. During the testing phase, if a network sample shows a high reconstruction error, it is likely to be considered as a malicious packet. This is because an autoencoder trained on normal network traffic packets generally has low reconstruction error for normal data.

4.2 Autoencoder Model

In this study, we use an autoencoder architecture comprising five layers. The number of nodes in each layer, ranging from the input to the output layer, is [122-32-10-32-122], influenced by the model architectures investigated in [7]. However, it is important to note that our primary focus is not the impact of different autoencoder model architectures; instead, we aim to compare the performance of an autoencoder with low-resolution quantized DW synapses to a similar autoencoder with floating-point synapses. The autoencoder comprises three hidden layers: the first hidden layer consists of 3904 synapses, the second hidden layer consists of 320 synapses, and the third hidden layer also contains 320 synapses. Furthermore, the output layer is composed of 3904 synapses. Thus, the total number of artificial synapses within the autoencoder architecture sums up to 8448.

The input features undergo a sequence of weighted sum operations and activation functions within the autoencoder model until they are reconstructed in the output layer. Specifically, we employ the sigmoid function as the activation function across all hidden and output layers within the autoencoder architecture. This activation choice introduced essential non-linearity into the model, enhancing its ability to learn intricate patterns and representations embedded within the input data. The sigmoid function, which maps input values to the interval [0, 1], aligns well with our autoencoder's requirements, where both input data and reconstructed outputs are confined to this range. By utilizing sigmoid activation in the output layer, our model's reconstructed outputs naturally conform to the [0, 1] input data scale, simplifying direct comparison and reconstruction tasks. In contrast, alternative activation functions might necessitate additional normalization or scaling methods to match the input/output data range. Moreover, the sigmoid function's smoothness and differentiability contribute to stable and efficient training of our autoencoder model, particularly when employing gradient-based optimization techniques. The autoencoder is trained using the backpropagation algorithm in conjunction with stochastic gradient descent. This enables the adjustment of the weights and biases of the model based on the difference between the predicted output and the actual input. The performance of the autoencoder is measured using MSE as the loss function. Fig. 2 illustrates the autoencoder architecture.

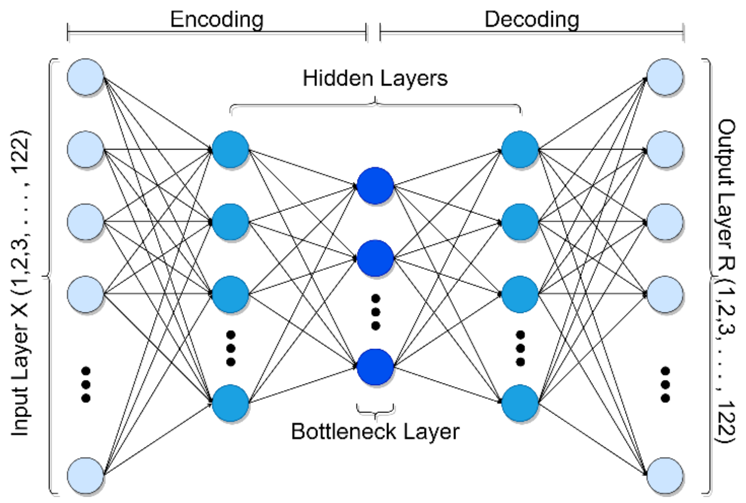
Encoding
Decoding
Hidden Layers
Input Layer X (1,2,3, . . . , 122)
Output Layer R (1,2,3, . . . , 122)
Bottleneck Layer

Figure 2: Proposed autoencoder model with five layers.

4.3 Autoencoder Quantization

Since the proposed nanomagnetic synapses have a limited number of states, the autoencoder parameters are quantized in the training and the inference stages. The quantization process can be mathematically expressed with following functions [50]:

$$\mathrm{clip}(N_{fp}; l, h) = \max(l, \min(h, N_{fp}))$$

$$N_q = \mathrm{round}\left(\frac{\mathrm{clip}(N_{fp}; l, h) - l}{h - l}(n-1)\right) \times \frac{h-l}{n-1} + l \tag{4}$$

where N_q is the quantized value, N_{fp} is the full precision value, [l; h] is the quantization range, and n is the number of quantization levels.

4.4 Autoencoder Training Process

While KDDTrain+ and KDDTest+ carry both normal and malicious traffic samples, only the normal traffic samples from the KDDTrain+ are used for training the quantized autoencoder. Fig. 3 illustrates the workflow for quantized autoencoder based anomaly detection. In the training phase, the processed normal traffic samples are sent to the autoencoder, where the original features are encoded to a latent space representation. Next, output features are reconstructed from this latent space. The reconstruction error is assessed from the difference between the reconstructed traffic sample and the original sample. The reconstruction error is measured for all the traffic samples and the standard deviation is estimated, which acts as the threshold for detecting anomalies. The details of threshold calculation can be found in the Algorithm section. In the inference phase, network traffic samples (carrying both normal and malicious samples) are sent to the trained autoencoder, and reconstruction error is calculated for each sample. The difference between the reconstruction error of a sample and the mean error of samples is referred to as the anomaly score (AS). The anomaly score is then compared to the threshold. If the anomaly score is higher than or equal to the threshold, then the traffic sample is inferred as malicious. Otherwise, the sample is inferred as benign.

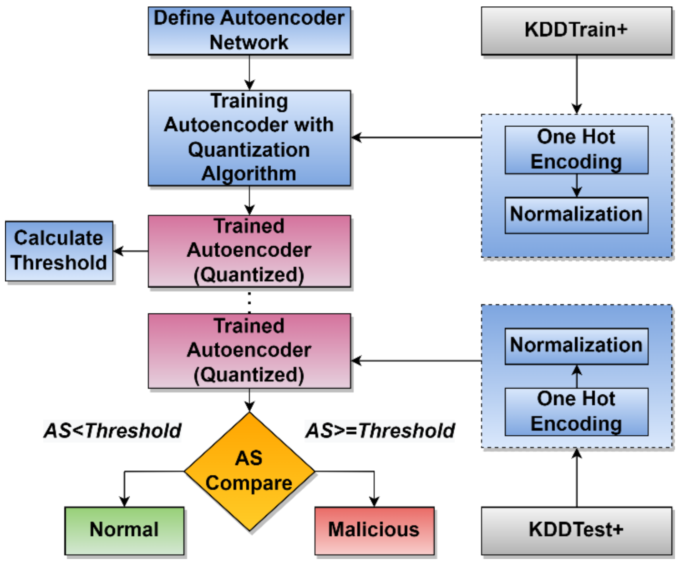

Figure 3: Schematic diagram of quantized autoencoder based anomaly detection.

Quantization-aware training of the autoencoder is performed by quantizing the weights according to (4) in the feed-forward phase and using the backpropagation algorithm based on low-precision neural network training [11]. In this process, while all weights are quantized in the forward pass, weight gradients are stored in separate high precision memory units during the backward pass to retain accuracy. However, the weight gradients need to propagate through non-differentiable quantization blocks in the backward pass. To tackle this issue, the straight through estimator (STE) is used, which provides a workaround by treating the quantization operation as a simple identity function during backpropagation [11]. This allows the gradients to be backpropagated as if the quantization were not applied.

Algorithm: Quantization-aware training for DW synapse-based autoencoder for network anomaly detection:

```
Input: Training dataset $\mathbf{X_0} = [X_1, X_2, X_3, \ldots, X_N]$
       Testing dataset $\mathbf{T_0} = [T_1, T_2, T_3, \ldots, T_M]$

// X and T are both n dimensional vectors
// Encoder: E, Decoder: D, Cost function: C
// Output: Anomaly = "1", Normal = "0"
// Number of Layers: L, Learning rate: η, Noise Tolerance Margin: α

begin
      //Training Phase
      Initialize Weights: W ← Gaussian Dist. [-1, 1]
    for number of training iterations do
    //Step 1: Feed-Forward
    for k = 1 to L do
    $\mathbf{X_k} = \mathbf{X_{k-1}W_k}$
    end
    $\mathbf{Y} = \mathbf{X_L} = \mathbf{D(E(X))}$
    //Step 2: Compute Gradients
    Compute gradient $\mathbf{G_L} = \frac{\partial C}{\partial X_L}$ from $\mathbf{X_L}$ and $\mathbf{X_0}$
    for k = L to 1 do
    $\mathbf{G_{k-1}} = \mathbf{G_k W_k}$
    $\Delta\mathbf{W_k} = \eta\mathbf{G_k X_{k-1}}$
    end
    //Step 3: Weight Update
    for k = 1 to L do
    $\mathbf{W_{k,fp}}(t+1) \leftarrow$ Update ($\mathbf{W_{k,fp}}(t)$ , $\Delta\mathbf{W_k}$)
    $\mathbf{W_{k,quant}}(t+1) \leftarrow$ Quantize (Clip ($\mathbf{W_{k,fp}}(t+1), -1, 1$), QL)
    //QL = Number of levels/states for Quantization
      if $|\mathbf{W_k}(t) - \mathbf{W_{k,quant}}(t+1)| > \alpha$ then
       $\mathbf{W_k}(t+1) \leftarrow$ Program ($\mathbf{W_k}(t)$, $\mathbf{W_{k,quant}}(t+1)$)
      end
    end
    end
    //Threshold Calculation from Training dataset
    for j = 1 to N do
    $D_j = \sqrt{\sum_{i=1}^{n} \|X_{ji} - Y_{ji}\|^2}$
    end
    $D_m = \frac{1}{N}\sum_{j=1}^{N} D_j$
    Threshold, $D_{SD} = \sqrt{\frac{\sum_{j=1}^{N}\|D_j - D_m\|^2}{N}}$
    // Inference Phase
```

```
        $\mathbf{Z} = \mathbf{T_L} = \mathbf{D_{quant}}(\mathbf{E_{quant}}(\mathbf{T}))$
        for j = 1 to M do
        $F_j = \sqrt{\sum_{i=1}^{n} \|T_{ji} - Z_{ji}\|^2}$
        end
        for j = 1 to M do
        $\Delta = |F_j - D_m|$
            if $\Delta >= D_{SD}$ then
                $T_j$ is an anomaly
                insert $T_j$ to Anomaly Set
            else
                $T_j$ is not an anomaly
                insert $T_j$ to Normal Set
            end
        end
end
```

5. Implementing Quantized Synapses with Non-Volatile Memory

5.1 DW Synapse Design

In this section, we discuss the proposed low-resolution (quantized) DW synapse design for the autoencoder neurons. Being non-volatile, the DW memory retains data for a long time even in absence of power. We design the device by simulating a thin ferromagnetic racetrack with five engineered notches where DW positions can be controlled with current pulse. The racetrack has a dimension of 560 nm × 60 nm × 1 nm. Along the racetrack, engineered notches are incorporated at a regular interval of 100 nm. However, the first and the fifth notches are positioned at 80 nm and 480 nm respectively. Fig. 4 illustrates the design of the nanomagnetic DW-based synaptic device.

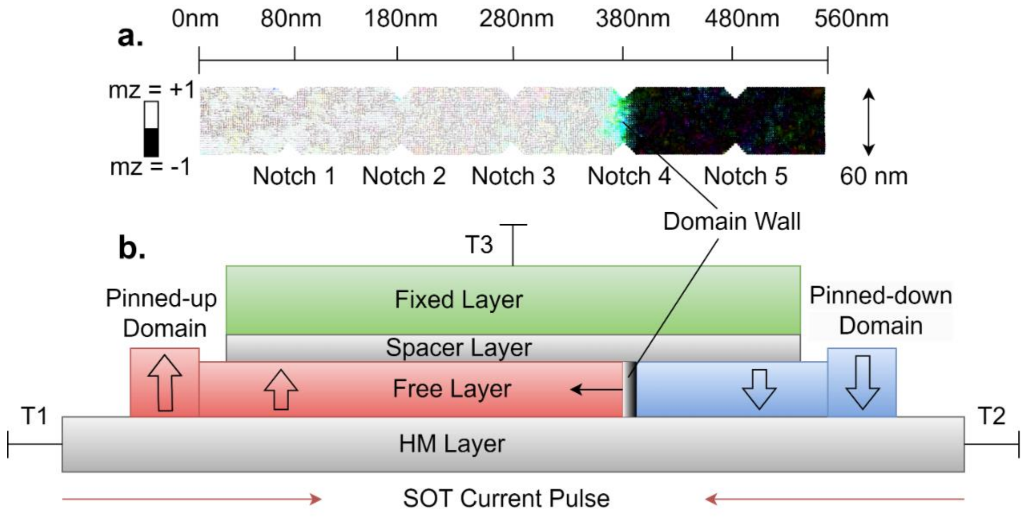

Figure 4: Schematic diagram of magnetic domain wall non-volatile synapses driven by SOT current pulses: (a) A sample of micromagnetic simulations showing pinned position of the domain wall, (b) Configuration of DW device with 5 notches.

In the designed DW synapse configuration, fixed magnetization zones are strategically positioned at the ends of the DW track to ensure stability and control over the DW movement. Specifically, at the start of the DW track, a pinned up-state domain is established on the left, and conversely, a pinned down-state domain is established on the right at the end of the DW track. Due to this configuration, the left side of the DW track, adjacent to notch 1, remains consistently in the up-state, while the right side, near notch 5, remains in the down-state. This fixed zone arrangement effectively confines the DW within the racetrack structure.

To control the DW position, current pulses with specific amplitude and fixed pulse duration are applied across the heavy metal layer. The current pulses generate SOT, which acts on the magnetic racetrack above it. By varying the number and the direction of current pulses, DW can be moved to different positions.

An MTJ is formed by combining the racetrack (free layer), an insulator (MgO tunneling layer), and a ferromagnetic material (reference layer) as seen in Fig. 4b. The MTJ is used to read the state of the racetrack's magnetization. DW positions are encoded as the conductance of MTJ, thereby creating a synapse that can be programmed with current pulses. For the DW-based MTJ device, resistance values of 16.75 kΩ and 23.45 kΩ represent the low and high resistance states respectively, with a difference of 6.7 kΩ between these states. This configuration results in a tunneling magnetoresistance (TMR) value of 40% during read operations [51].

5.2 Micromagnetic Simulations

Extensive micromagnetic simulations are performed using mumax3 [52], which simulate the magnetization dynamics of the DW in the magnetic racetrack while considering thermal noise at room temperature (300 K). The simulations provide insights into the evolution of the DW synapse. Table 2 presents the parameters employed for the micromagnetic simulation.

TABLE 2
Parameters for the Micromagnetic Simulations [23]

Symbol	Description	Value
α	Damping Parameter	0.015
D	DMI Constant	0.0006 $\mathrm{Jm^{-2}}$
A_{ex}	Exchange Constant	2×10^{-11} $\mathrm{Jm^{-1}}$
Ψ	Gilbert Damping	0.03
K_u	Perpendicular Magnetic Anisotropy (PMA)	7.5×10^{5} $\mathrm{Jm^{-3}}$
M_s	Saturation Magnetization	10^{6} $\mathrm{Am^{-1}}$
λ_s	Saturation Magnetostriction	250 ppm
T	Temperature	300 K

Fig. 4a illustrates the micromagnetic configuration of the racetrack's free layer. A fixed amplitude current pulse of 85×10^{9} $\mathrm{A/m^2}$ with a fixed duration of 0.5 ns is applied through the heavy metal layer to initiate the DW depinning from its initial pinned position and move it towards the intended adjacent notch. However, due to the DW tilting caused by the presence of Dzyaloshinskii-Moriya interaction (DMI) [53] and thermal noise, the DW exhibits significant stochastic motion when driven by the SOT current pulses. As a result, the DW might get pinned at a different notch position rather than getting pinned at the intended specific notch position after the SOT current pulses are applied. Fig. 5 illustrates the probabilistic distribution of the DW positions due to stochastic variation in the DW motion. The equilibrium pinned positions of the DWs are used to calculate the conductance of the MTJ using the subsequent equation (5) [24]:

$$G_{Synapse} = \frac{G_{max} + G_{min}}{2} + \frac{G_{max} - G_{min}}{2} < m_z > \tag{5}$$

where $< m_z >$ denotes the average magnetization moment of the ferromagnetic racetrack along the z-direction. The magnetization of the reference ferromagnetic layer is assumed to point upward in the +z-direction. G_{max} and G_{min} respectively denote the maximum and minimum conductance of the synaptic device.

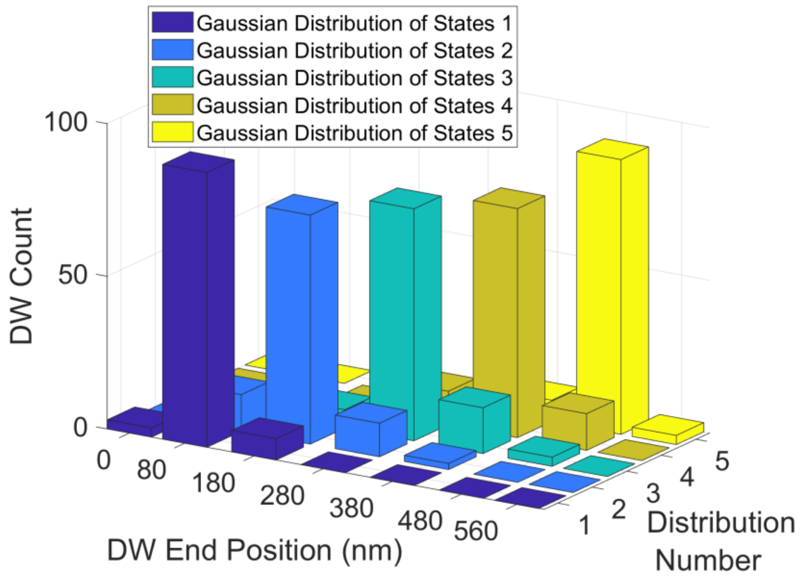

Figure 5: Probabilistic distribution of DW positions and DW counts after SOT current pulses are driven to move the DW to target notches from adjacent notch positions.

The conductance of the DW device ($G_{synapse}$) which corresponds to the position of the DW, represents the weights. However, this means that only positive weights can be attained using the DW device since the conductance values are inherently positive quantity. Consequently, synaptic weight updates are confined to positive values spanning from G_{min} to G_{max}. To address the need for weight updates in both positive and negative directions, a circuit model illustrated in Fig. 6 is designed [24]. This model utilizes two separate rows connected to a single column (bit line or BL) in the crossbar. These rows are supplied with opposite polarity voltages and are responsible for connecting the synaptic devices to additional conductance ($G_{parallel}$) in parallel. A negative input voltage is applied to $G_{parallel}$ to accommodate both positive and negative linear weight updates. The synapse conductance can be calculated using Kirchhoff's current law for a single column [57]:

$$I_{eq} = -V_1 G_{parallel} + V_1 W_{1,1} \dots \dots \dots \dots \dots - V_m G_{parallel} + V_m W_{m,1} \tag{6}$$

where $G_{parallel} = \frac{G_{max}+G_{min}}{2}$ (average of the maximum and minimum conductance values in DW device).

Solution of equation (6) for single synapse:

$$I_{eq} = -V_1 G_{parallel} + V_1 W_{1,1} \tag{7}$$

$$\frac{I_{eq}}{V_1} = -G_{parallel} + W_{1,1} = G_{1eq,synapse} \tag{8}$$

where $W_{1,1}$ is $G_{synapse}$ with respect to the position of the DW. Equivalent conductance and resistance can be formulated as:

$$G_{1eq,synapse} = -\frac{G_{max}+G_{min}}{2} + G_{synapse} \tag{9}$$

$$R_{1eq,synapse} = \frac{2R_{min}R_{max}R_{synapse}}{2R_{min}R_{max}-R_{min}R_{synapse}-R_{max}R_{synapse}} \tag{10}$$

where $R_{synapse}$ is the DW device resistance corresponding to the position of DW. R_{min} and R_{max} are resistances corresponding to G_{min} and G_{max}, respectively. Consequently, this approach enables the realization of linear weights in both positive and negative directions.

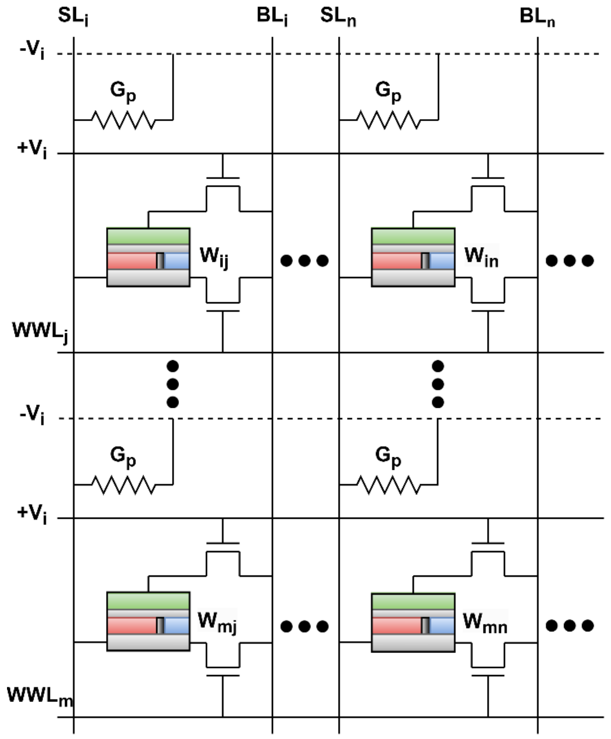

Figure 6: Implementation of autoencoder in crossbar architecture with nonvolatile DW memory-based synapses.

To enable read and write operations, additional components such as the write and read word lines (WWL/RWL) and source line (SL) are introduced (as shown in Fig. 6). It is worth noting that for simplicity, the WWL for the parallel conductance is not shown in the figure. To perform column sum read/write operations, the RWL or WWL is activated accordingly. When programming a device, the WWL is activated, and the SL and BL are adjusted to high or low levels based on the direction and number of current pulses. The WWL/RWL, SL, and BL are controlled and operated by external transistors, which function as switches regulating the current flow to these lines based on the required operations. These transistors play a pivotal role in overseeing the read and write processes of synaptic devices, particularly in managing current pulses during programming and inference phases.

6. Results

6.1 Performance Measures

We employ accuracy, precision, true positive rate (TPR), and F1 score to assess the performance of the autoencoder model. These performance measures can be expressed using four quantities: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). TP represents the number of correctly classified malicious samples, TN represents the number of correctly classified normal samples, FP represents the number of normal samples incorrectly classified as malicious samples, and FN represents the number of malicious samples incorrectly classified as normal samples.

Accuracy quantifies the overall correctness of the classification model and represents the ratio of correctly classified packets to the total number of packets. It is represented by the following equation (11):

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{11}$$

Precision quantifies the proportion of correctly classified malicious samples out of all samples predicted as malicious. It is expressed as equation (12):

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{12}$$

TPR measures the proportion of correctly classified malicious samples out of all malicious packets. It is expressed as equation (13):

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{13}$$

The F1 score is a harmonic mean of precision and TPR, providing a balanced measure of both metrics. It is expressed as equation (14):

$$\text{F1 score} = \frac{2 \times \text{Precision} \times \text{TPR}}{\text{Precision} + \text{TPR}} \tag{14}$$

6.2 Results for Quantized Autoencoder

In this section, we compare the anomaly detection performance of the proposed quantized autoencoder (without DW Synapse) with 2, 3, and 5-level weight quantization to a structurally identical autoencoder with floating-point precision (32-bit) weights. The performance evaluation is based on testing accuracy, precision, TPR, and F1 score. Fig. 7 illustrates these performance metrics for the autoencoder with 2, 3, 5-level quantized weights, and floating-point precision weights.

Fig. 7a demonstrates that employing only 2 quantization levels in the autoencoder leads to random fluctuations in the resulting accuracy as the training progresses through successive epochs. Thus, training with 2 quantization levels shows occasional high accuracy with extended training cycles; however, predicting the number of epochs required to achieve high accuracy remains challenging since the accuracy does not converge over time. Similarly, training the autoencoder with 3 quantization levels yields a comparable outcome, showing random fluctuations in accuracy with extended training cycles. However, the fluctuation pattern is less random compared to the previous case. On the other hand, training the autoencoder with 5 quantization levels demonstrates a more deterministic accuracy progression. Notably, across all epochs, the accuracy for training with 5-level quantized weights is competitive compared to the accuracy for training with floating-point weights. Similar conclusions can be drawn for the remaining performance metrics illustrated in Fig. 7b, Fig. 7c, and Fig. 7d.

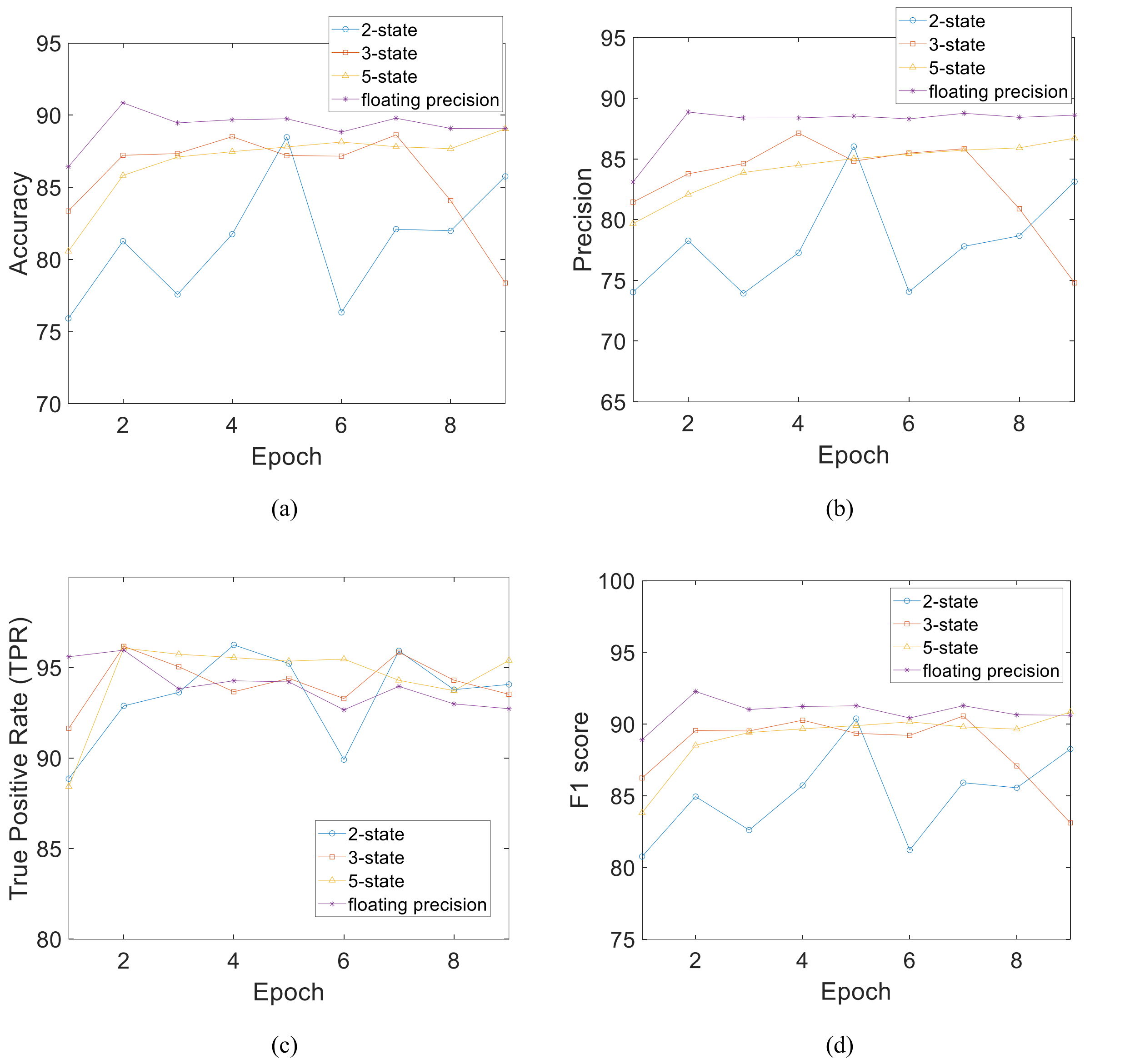

Figure 7: Anomaly detection testing (a) Accuracy (b) Precision (c) TPR, and (d) F1 score for autoencoder with different quantization levels (2, 3, and 5-level weight quantization) and floating-point weights.

Based on the results illustrated in Fig. 7, it is evident that training with 5 quantization levels yields competitive performance metrics compared to training with floating-point weights, despite the significantly reduced number of bits and weight precision required by 5-level quantization. Quantization-aware training enables the autoencoder model to adapt to lower-precision weight representation. While the use of a limited number of quantization levels is typically considered detrimental to training and testing performance, quantization-aware training can leverage quantization noise to achieve a significant improvement in performance metrics. In this case, quantization acts as a regularization operation that limits overfitting of the trained weights. By reducing the precision of numerical values in network parameters, quantization reduces model complexity and prevents the network from memorizing outliers or noise in the training data. Therefore, this reduction in precision introduces a controlled level of noise or approximation error, which helps smooth out decision boundaries and makes the network less sensitive to small variations in the input data. Additionally, the computational efficiency gained from quantization, such as faster inference and reduced memory requirements, indirectly contributes to regularization by reducing the risk of overfitting that can arise from longer training times or limited training data. Furthermore, the results demonstrate that, for this specific anomaly detection problem, 5-level quantization is optimal as it represents the minimum level of quantization where performance metrics closely match those obtained with floating-point weights. For example, a reduced number of quantization levels (2 or 3-level) could be more hardware efficient; however, the performance is not consistent. This motivates our study of a multistate non-volatile synaptic memory that can maintain at least five different non-volatile resistance states.

6.3 Results for Quantized DW-based Autoencoder

In this section, we evaluate and compare the effectiveness of anomaly detection in three different configurations of autoencoders: one with quantized synapses (without DW device), another with quantized DW-based stochastic synapses, and the third with floating-point precision synapses.

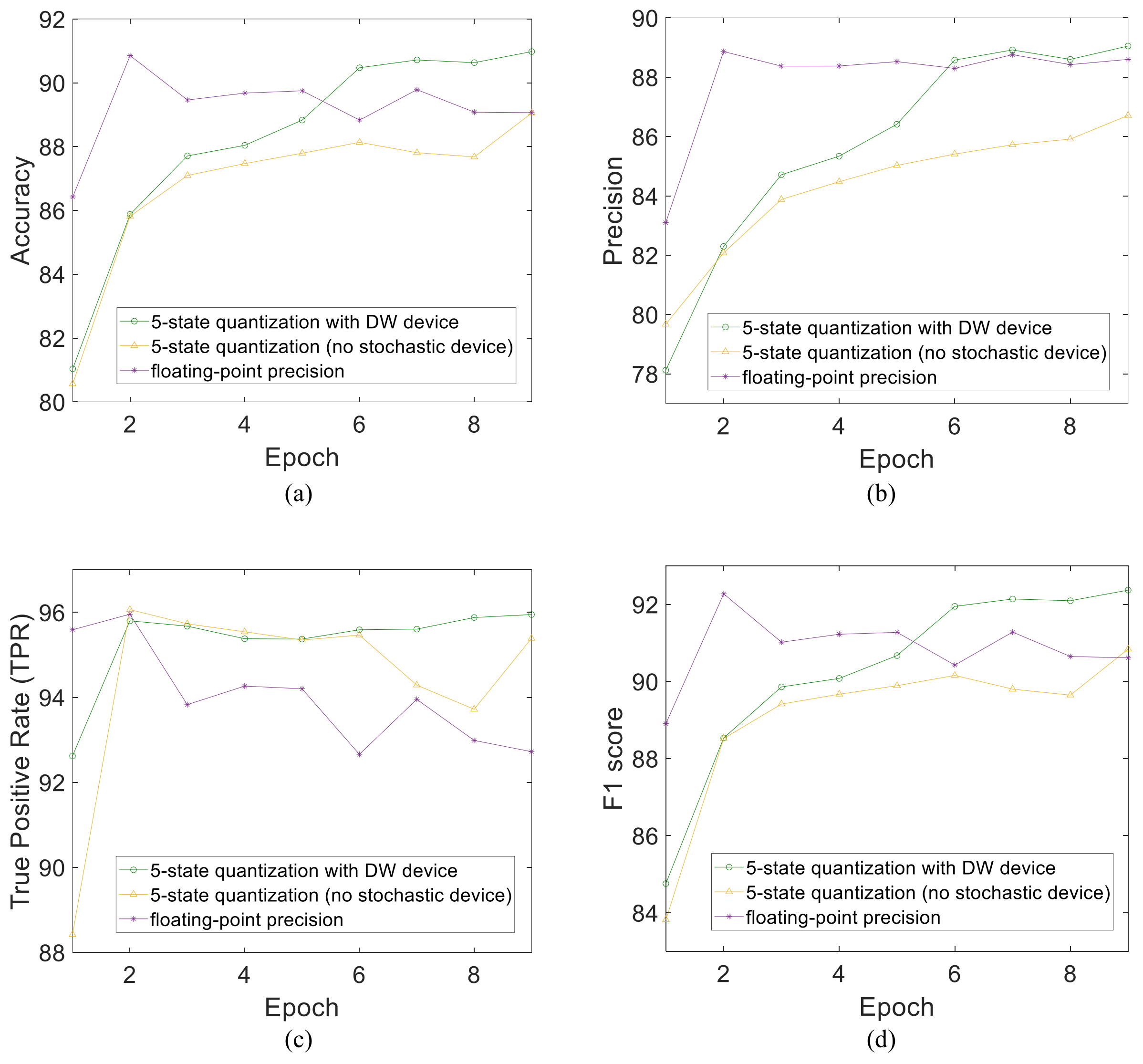

Figure 8: Anomaly detection testing (a) Accuracy, (b) Precision, (c) TPR, and (d) F1 score plotted against epoch for autoencoder with 5-state quantized DW synapses, 5-state synapses (without stochastic device), and floating-point weight synapses.

Fig. 8 illustrates anomaly detection testing accuracy, precision, TPR, and F1 score for different configurations of the autoencoder. From Fig. 8a it can be inferred that when using only five quantization levels (without DW device), the quantized autoencoder achieves competitive anomaly detection accuracy compared to the autoencoder with floating-point precision weights. This performance can be attributed to quantization acting as a regularization operation, as explained in the previous section. However, the autoencoder with quantized DW-based stochastic synapses shows higher accuracy than both the autoencoder with quantized synapses (without DW device) and the autoencoder with floating-point precision synapses. For the autoencoder with DW-based synapses, non-volatile synapses are designed using a specific hardware technology called racetrack MTJ. The synapses can encode multiple non-volatile states and the training process considers the characteristics of the device, such as noise and stochasticity. By introducing randomness during the training process, stochasticity serves as a regularization technique. It adds noise to the model and encourages exploration of different solutions, thus reducing the risk of overfitting to specific patterns in the training data [56]. The results obtained with the DW synapses illustrate a higher anomaly detection accuracy (90.98%) surpassing even the floating-point accuracy (90.85%). Similar conclusions can be drawn for the remaining performance metrics illustrated in Fig. 8b, Fig. 8c, and Fig. 8d. The findings presented in Fig. 8 indicate that combining stochasticity with quantization further improves the performance of anomaly detection. This combination acts as a better regularization process. Moreover, the fact that stochasticity arises from the inherent properties of the DW devices, rather than being added separately makes it energy efficient as generating random numbers in software can be energy inefficient. Thus, stochasticity inherent to nanoscale devices, which is decremental to Boolean logic, is beneficial to hardware AI applications at no additional energy cost.

6.4 Total Number of Programmed Weights

In this section, we conduct a comparison of the total number of programmed weights (weight updates) across different autoencoder synapse schemes.

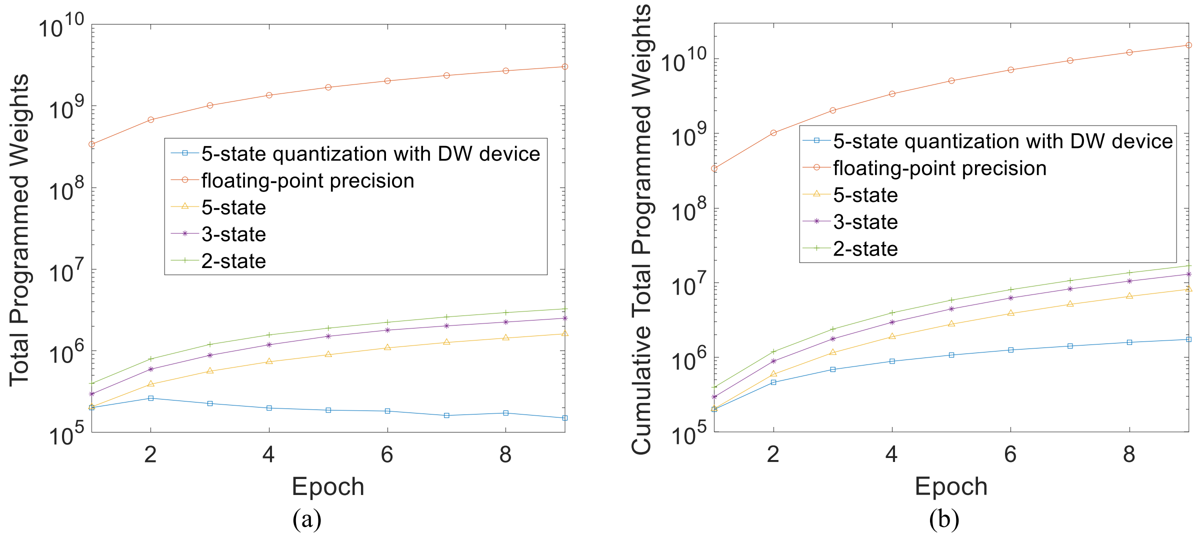

Figure 9: (a) Comparison of the total programmed weights at each epoch instance vs. epoch, and (b) Comparison of the cumulative total programmed weights vs. epoch for different autoencoder configurations. The proposed 5-state quantized DW synapse-based autoencoder shows significantly fewer weight updates during training compared to other autoencoder configurations.

Fig. 9a depicts the graphical representation of the total programmed weights at each epoch instance against the number of training epochs for the 5-state quantized DW synapse-based autoencoder, as well as the 2, 3, 5-level quantized weight autoencoders (without DW device), and the floating-point weight-based autoencoder. Similarly, Fig. 9b illustrates the cumulative total programmed weights against the number of training epochs for different autoencoder configurations. The data from Fig. 9 shows a significant distinction: the proposed DW-based approach exhibits a remarkable reduction of at least three orders of magnitude in weight updates when compared to the floating-point approach. The 2, 3, and 5-level quantized weight autoencoders also demonstrate notably fewer weight updates compared to their floating-point weight-based counterparts. Among these three quantization approaches, the 5-level quantized weight autoencoder requires the least number of weight updates. However, weight updates were further reduced with 5-state DW-based autoencoder. Moreover, it can be inferred from Fig. 9a that the inclusion of stochasticity with the 5-sate DW-based autoencoder results in diminishing number of weight updates at each epoch instance as the training epochs progress, in contrast to the pattern observed in other autoencoder configurations. Consequently, the proposed DW device-based autoencoder demonstrates a higher degree of computational resource efficiency.

6.5 Energy

The energy consumption in domain wall synapses stems from the I^2R losses induced by SOT current pulses in the heavy metal layer. Assuming the heavy metal layer is composed of Platinum (Pt) with a specific resistance of 100 Ω-nm, the resistance of a synapse is determined. Given the heavy metal layer's dimensions of $560 \times 60 \times 5\ \mathrm{nm}^3$, the calculated resistance is approximately 186.67 Ω. Subsequently, the energy dissipated for a current pulse of $85 \times 10^9\ \mathrm{A/m}^2$ applied for 0.5 ns in the heavy metal layer is computed to be 0.06 fJ. Consequently, the energy expenditure for programming a synapse with a current pulse is estimated to be around 0.06 fJ. The highest testing accuracy is obtained from the autoencoder with 5-state DW synapses with relatively low number of weight updates, considering high noise tolerance margin during training. By incorporating a noise tolerance margin of $\alpha = 0.25$, the overall count of weight updates reaches roughly 1.74 million after conducting training for 9 epochs. Consequently, the energy cost for programming domain wall synapses is approximately 1.56 fJ for a single inference event following the weight updates for each packet of 67,434 normal packets of KDDTrain+.

Next, we study the energy requirements within both the analog and digital domains for the training and inference processes of the proposed network. In the analog domain, energy utilization is contingent on executing matrix-vector multiplication during both forward and backward propagation, as well as updating the weights of the DW devices. On the digital front, energy is expended on tasks such as computing neuron activations, determining error gradients, and accumulating gradients for subsequent weight updates. The analog computation phase incurs energy consumption across various stages. This includes the sequential input and output data transfer to and from the crossbar rows and columns, the conversion of input data to voltage pulses using Pulse-width modulation (PWM), regulation of column voltage to a specified value for a corresponding voltage drop across DW devices, reading the analog weighted sum in the crossbar arrays, and the analog-to-digital conversion (ADC) of the weighted sum before transmitting it to the crossbar for the implementation of subsequent layers in the autoencoder. We adhere to an 8-bit resolution for both PWM and ADC, as detailed in [26]. Given the parameters specified in [24], [26], the energy consumption during the analog computation phase is approximated at ~ 0.32 nJ for forward propagation and ~ 0.18 nJ for backward propagation per training packet. The energy calculation guidelines are outlined in Ref. [26]. For DW device weight updates, an average of ~25 devices undergo updates during each training instance, resulting in an energy expenditure of ~1.56 fJ per training instance, with a write energy of ~ 0.06 fJ per update. Consequently, the cumulative energy in the analog computation phase is estimated to be ~ 0.5 nJ per training instance.

For updating DW device weights, gradients are initially accumulated in a digital unit using 32-bit precision memory. However, when quantizing neuron activations (during forward propagation) and error gradients (during backward propagation) to 3 bits, a significantly reduced number of 32-bit memory accesses becomes feasible, owing to the diminished non-zero entries resulting from quantization, without compromising accuracy [26], [54]. Moreover, if memory access occurs close to 1% of the total synapses (approximately 85 synapses, constituting 1% of 8448 synapses), the energy expended for weight updates in the digital unit can be approximated at ~ 2.64 nJ based on Ref. [26]. Considering that an analog read operation is preceded by a 4-bit PWM input signal (as a 4-bit PWM is sufficient, given that PWM output voltage resolution does not impact the reading of device conductance), followed by read voltage regulation and subsequent ADC operation, the total read energy is estimated to be ~ 0.21 nJ per training instance for reading device conductances and parallel conductances. Assuming comparator energy equals ADC energy at ~ 330 fJ, the combined energies for quantizer, read (including comparator), and other operations are estimated at ~ 3.51 nJ per training instance.

Furthermore, the energy dissipated in a digital unit for computing neuron activations (during forward propagation), error gradients (during backward propagation), and accurately addressing analog DW devices for sending write pulses can be extrapolated from the application-specific integrated circuit (ASIC) design implemented in Ref. [26] using on-chip static random access memory (SRAM). In their design, energy consumption for forward and backward passes in a digital unit is demonstrated to be ~ 9 nJ and ~ 3 nJ per training instance. Given that the number of synapses in Ref. [1] is approximately 23.5 times that of our network, we can estimate energy consumption for our architecture at ~ 0.38 nJ and ~ 0.13 nJ. Consequently, the total energy consumption in the digital unit is projected to be ~ 6.66 nJ per training instance. Combining the energies of the analog and digital units, the overall energy consumption is estimated to be ~ 7.16 nJ per training instance and ~ 0.7 nJ per inference instance (considering energies solely for forward propagations). Therefore, energy consumption is comparable to the state-of-the-art non-volatile technologies [24], [26] and would be significantly more efficient than using traditional von-Neumann schemes with purely CMOS devices [26]. As discussed, our algorithm also guarantees a substantially reduced frequency of weight programming, resulting in a minimal energy cost for training cost.

7. Conclusion

The state-of-the-art autoencoder based unsupervised anomaly detection methods have shown promising results in detecting network anomalies. However, implementing these methods on edge devices with limited hardware, computational resources, and energy has been a challenge. In this paper, we proposed a solution to this challenge by designing a quantized autoencoder with low-resolution non-volatile DW-based synaptic weights to detect anomalies efficiently on edge devices. We designed the synapses using racetrack MTJ in which the synapses can encode multiple non-volatile states. The hardware-aware training performed on the 5-state quantized DW-based autoencoder yields higher anomaly detection performance compared to the floating-point weight autoencoder. Therefore, our proposed solution offers a promising avenue for implementing efficient anomaly detection methods on edge devices with limited hardware resources. This technology is particularly well-suited for devices with size and power constraints, supporting applications in smart sensors, wearables, and IoT, where local processing and privacy are key considerations.

In the future, we would like to explore the compatibility of the proposed quantized DW-based autoencoder with diverse datasets and anomaly scenarios. Additionally, we are interested in integrating more advanced in-memory computing technologies into the autoencoder synapse design. Furthermore, we plan to investigate more complex networks, such as a transformer model designed with quantized DW-based stochastic synapses to perform anomaly detection.

Acknowledgements

This work was supported in part by the National Science Foundation (NSF) grant ECCS: #1954589, Commonwealth of Cyber Initiative (CCI) grants (award #VV-1Q23-008, #VV-1Q24-009), and Virginia Commonwealth University (VCU) Breakthrough grant.

References

[1] Zikria, Y.B., Ali, R., Afzal, M.K. and Kim, S.W., 2021. Next-generation internet of things (iot): Opportunities, challenges, and solutions. *Sensors*, *21*(4), p.1174.

[2] Jang-Jaccard, J. and Nepal, S., 2014. A survey of emerging threats in cybersecurity. *Journal of Computer and System Sciences*, *80*(5), pp.973-993.

[3] Chandola, V., Banerjee, A. and Kumar, V., 2009. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, *41*(3), pp.1-58.

[4] Ahmad, S., Lavin, A., Purdy, S. and Agha, Z., 2017. Unsupervised real-time anomaly detection for streaming data. *Neurocomputing*, *262*, pp.134-147.

[5] Chalapathy, R. and Chawla, S., 2019. Deep learning for anomaly detection: A survey. *arXiv preprint arXiv:1901.03407*.

[6] Zhou, C. and Paffenroth, R.C., 2017, August. Anomaly detection with robust deep autoencoders. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 665-674).

[7] Xu, W., Jang-Jaccard, J., Singh, A., Wei, Y. and Sabrina, F., 2021. Improving performance of autoencoder-based network anomaly detection on nsl-kdd dataset. *IEEE Access*, *9*, pp.140136-140146.

[8] Varghese, B., Wang, N., Barbhuiya, S., Kilpatrick, P. and Nikolopoulos, D.S., 2016, November. Challenges and opportunities in edge computing. In *2016 IEEE international conference on smart cloud (SmartCloud)* (pp. 20-26). IEEE.

[9] Shi, W., Cao, J., Zhang, Q., Li, Y. and Xu, L., 2016. Edge computing: Vision and challenges. *IEEE internet of things journal*, *3*(5), pp.637-646.

[10] Gholami, A., Kim, S., Dong, Z., Yao, Z., Mahoney, M.W. and Keutzer, K., 2021. A survey of quantization methods for efficient neural network inference. *arXiv preprint arXiv:2103.13630*.

[11] Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R. and Bengio, Y., 2017. Quantized neural networks: Training neural networks with low precision weights and activations. *The Journal of Machine Learning Research*, *18*(1), pp.6869-6898.

[12] Park, E., Yoo, S. and Vajda, P., 2018. Value-aware quantization for training and inference of neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)* (pp. 580-595).

[13] Fan, A., Stock, P., Graham, B., Grave, E., Gribonval, R., Jegou, H. and Joulin, A., 2020. Training with quantization noise for extreme model compression. *arXiv preprint arXiv:2004.07320*.

[14] Sebastian, A., Le Gallo, M., Khaddam-Aljameh, R. and Eleftheriou, E., 2020. Memory devices and applications for in-memory computing. *Nature nanotechnology*, *15*(7), pp.529-544.

[15] Ielmini, D. and Wong, H.S.P., 2018. In-memory computing with resistive switching devices. *Nature electronics*, *1*(6), pp.333-343.

[16] Wong, H.S.P. and Salahuddin, S., 2015. Memory leads the way to better computing. *Nature nanotechnology*, *10*(3), pp.191-194.

[17] Sengupta, A., Shim, Y. and Roy, K., 2016. Proposal for an all-spin artificial neural network: Emulating neural and synaptic functionalities through domain wall motion in ferromagnets. *IEEE transactions on biomedical circuits and systems*, *10*(6), pp.1152-1160.

[18] Bhowmik, D., Saxena, U., Dankar, A., Verma, A., Kaushik, D., Chatterjee, S. and Singh, U., 2019. On-chip learning for domain wall synapse based fully connected neural network. *Journal of Magnetism and Magnetic Materials*, *489*, p.165434.

[19] Vincent, A.F., Larroque, J., Locatelli, N., Romdhane, N.B., Bichler, O., Gamrat, C., Zhao, W.S., Klein, J.O., Galdin-Retailleau, S. and Querlioz, D., 2015. Spin-transfer torque magnetic memory as a stochastic memristive synapse for neuromorphic systems. *IEEE transactions on biomedical circuits and systems*, *9*(2), pp.166-174.

[20] Alamdar, M., Leonard, T., Cui, C., Rimal, B.P., Xue, L., Akinola, O.G., Patrick Xiao, T., Friedman, J.S., Bennett, C.H., Marinella, M.J. and Incorvia, J.A.C., 2021. Domain wall-magnetic tunnel junction spin–orbit torque devices and circuits for in-memory computing. *Applied Physics Letters*, *118*(11).

[21] Ikeda, S., Hayakawa, J., Ashizawa, Y., Lee, Y.M., Miura, K., Hasegawa, H., Tsunoda, M., Matsukura, F. and Ohno, H., 2008. Tunnel magnetoresistance of 604% at 300K by suppression of Ta diffusion in CoFeB∕ MgO∕ CoFeB pseudo-spin-valves annealed at high temperature. *Applied Physics Letters*, *93*(8).

[22] Jiang, X., Thomas, L., Moriya, R., Hayashi, M., Bergman, B., Rettner, C. and Parkin, S.S., 2010. Enhanced stochasticity of domain wall motion in magnetic racetracks due to dynamic pinning. *Nature communications*, *1*(1), p.25.

[23] Al Misba, W., Kaisar, T., Bhattacharya, D. and Atulasimha, J., 2021. Voltage-controlled energy-efficient domain wall synapses with stochastic distribution of quantized weights in the presence of thermal noise and edge roughness. *IEEE Transactions on Electron Devices*, *69*(4), pp.1658-1666.

[24] Al Misba, W., Lozano, M., Querlioz, D. and Atulasimha, J., 2022. Energy efficient learning with low resolution stochastic domain wall synapse for deep neural networks. *IEEE Access*, *10*, pp.84946-84959.

[25] Le Gallo, M., Sebastian, A., Mathis, R., Manica, M., Giefers, H., Tuma, T., Bekas, C., Curioni, A. and Eleftheriou, E., 2018. Mixed-precision in-memory computing. *Nature Electronics*, *1*(4), pp.246-253.

[26] Nandakumar, S.R., Le Gallo, M., Piveteau, C., Joshi, V., Mariani, G., Boybat, I., Karunaratne, G., Khaddam-Aljameh, R., Egger, U., Petropoulos, A. and Antonakopoulos, T., 2020. Mixed-precision deep learning based on computational memory. *Frontiers in neuroscience*, *14*, p.406.

[27] Tavallaee, M., Bagheri, E., Lu, W. and Ghorbani, A.A., 2009, July. A detailed analysis of the KDD CUP 99 data set. In *2009 IEEE symposium on computational intelligence for security and defense applications* (pp. 1-6). Ieee.

[28] Liu, H. and Lang, B., 2019. Machine learning and deep learning methods for intrusion detection systems: A survey. *applied sciences*, *9*(20), p.4396.

[29] Rhode, M., Burnap, P. and Jones, K., 2018. Early-stage malware prediction using recurrent neural networks. *computers & security*, *77*, pp.578-594.

[30] Agrawal, S. and Agrawal, J., 2015. Survey on anomaly detection using data mining techniques. *Procedia Computer Science*, *60*, pp.708-713.

[31] Ashfaq, R.A.R., Wang, X.Z., Huang, J.Z., Abbas, H. and He, Y.L., 2017. Fuzziness based semi-supervised learning approach for intrusion detection system. *Information sciences*, *378*, pp.484-497.

[32] Ingre, B. and Yadav, A., 2015, January. Performance analysis of NSL-KDD dataset using ANN. In *2015 international conference on signal processing and communication engineering systems* (pp. 92-96). IEEE.

[33] Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J. and Ahmad, F., 2021. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies*, *32*(1), p.e4150.

[34] Sommer, R. and Paxson, V., 2010, May. Outside the closed world: On using machine learning for network intrusion detection. In *2010 IEEE symposium on security and privacy* (pp. 305-316). IEEE.

[35] Alam, M.S., Fernando, B.R., Jaoudi, Y., Yakopcic, C., Hasan, R., Taha, T.M. and Subramanyam, G., 2019, July. Memristor based autoencoder for unsupervised real-time network intrusion and anomaly detection. In *Proceedings of the International Conference on Neuromorphic Systems* (pp. 1-8).

[36] Grezes, C., Ebrahimi, F., Alzate, J.G., Cai, X., Katine, J.A., Langer, J., Ocker, B., Khalili Amiri, P. and Wang, K.L., 2016. Ultra-low switching energy and scaling in electric-field-controlled nanoscale magnetic tunnel junctions with high resistance-area product. *Applied Physics Letters*, *108*(1).

[37] Bhattacharya, D., Al-Rashid, M.M. and Atulasimha, J., 2016. Voltage controlled core reversal of fixed magnetic skyrmions without a magnetic field. *Scientific reports*, *6*(1), p.31272.

[38] Bhattacharya, D., Razavi, S.A., Wu, H., Dai, B., Wang, K.L. and Atulasimha, J., 2020. Creation and annihilation of non-volatile fixed magnetic skyrmions using voltage control of magnetic anisotropy. *Nature Electronics*, *3*(9), pp.539-545.

[39] Li, X., Carka, D., Liang, C.Y., Sepulveda, A.E., Keller, S.M., Amiri, P.K., Carman, G.P. and Lynch, C.S., 2015. Strain-mediated 180 perpendicular magnetization switching of a single domain multiferroic structure. *Journal of Applied Physics*, *118*(1).

[40] Roy, K., Bandyopadhyay, S. and Atulasimha, J., 2011. Hybrid spintronics and straintronics: A magnetic technology for ultra low energy computing and signal processing. *Applied Physics Letters*, *99*(6).

[41] Lei, N., Devolder, T., Agnus, G., Aubert, P., Daniel, L., Kim, J.V., Zhao, W., Trypiniotis, T., Cowburn, R.P., Chappert, C. and Ravelosona, D., 2013. Strain-controlled magnetic domain wall propagation in hybrid piezoelectric/ferromagnetic structures. *Nature communications*, *4*(1), p.1378.

[42] Biswas, A.K., Bandyopadhyay, S. and Atulasimha, J., 2014. Complete magnetization reversal in a magnetostrictive nanomagnet with voltage-generated stress: A reliable energy-efficient non-volatile magneto-elastic memory. *Applied Physics Letters*, *105*(7).

[43] Sampath, V., D'Souza, N., Bhattacharya, D., Atkinson, G.M., Bandyopadhyay, S. and Atulasimha, J., 2016. Acoustic-wave-induced magnetization switching of magnetostrictive nanomagnets from single-domain to nonvolatile vortex states. *Nano Letters*, *16*(9), pp.5681-5687.

[44] Slonczewski, J.C., 1996. Current-driven excitation of magnetic multilayers. *Journal of Magnetism and Magnetic Materials*, *159*(1-2), pp.L1-L7.

[45] Ryu, K.S., Yang, S.H., Thomas, L. and Parkin, S.S., 2014. Chiral spin torque arising from proximity-induced magnetization. *Nature communications*, *5*(1), p.3910.

[46] Al Misba, W., Rajib, M.M., Bhattacharya, D. and Atulasimha, J., 2020. Acoustic-wave-induced ferromagnetic-resonance-assisted spin-torque switching of perpendicular magnetic tunnel junctions with anisotropy variation. *Physical Review Applied*, *14*(1), p.014088.

[47] Azam, M.A., Bhattacharya, D., Querlioz, D., Ross, C.A. and Atulasimha, J., 2020. Voltage control of domain walls in magnetic nanowires for energy-efficient neuromorphic devices. *Nanotechnology*, *31*(14), p.145201.

[48] Courbariaux, M., Bengio, Y. and David, J.P., 2015. Binaryconnect: Training deep neural networks with binary weights during propagations. *Advances in neural information processing systems*, *28*.

[49] Duque, S. and bin Omar, M.N., 2015. Using data mining algorithms for developing a model for intrusion detection system (IDS). *Procedia Computer Science*, *61*, pp.46-51.

[50] Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H. and Kalenichenko, D., 2018. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2704-2713).

[51] Slaughter, J.M., Dave, R.W., Durlam, M., Kerszykowski, G., Smith, K., Nagel, K., Feil, B., Calder, J., De Herrera, M., Garni, B. and Tehrani, S., 2005, December. High speed toggle MRAM with MgO-based tunnel junctions. In *IEEE InternationalElectron Devices Meeting, 2005. IEDM Technical Digest.* (pp. 873-876). IEEE.

[52] Vansteenkiste, A., Leliaert, J., Dvornik, M., Helsen, M., Garcia-Sanchez, F. and Van Waeyenberge, B., 2014. The design and verification of MuMax3. *AIP advances*, *4*(10).

[53] Liu, S., Xiao, T.P., Cui, C., Incorvia, J.A.C., Bennett, C.H. and Marinella, M.J., 2021. A domain wall-magnetic tunnel junction artificial synapse with notched geometry for accurate and efficient training of deep neural networks. *Applied Physics Letters*, *118*(20).

[54] Marinella, M.J., Agarwal, S., Hsia, A., Richter, I., Jacobs-Gedrim, R., Niroula, J., Plimpton, S.J., Ipek, E. and James, C.D., 2018. Multiscale co-design analysis of energy, latency, area, and accuracy of a ReRAM analog neural training accelerator. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, *8*(1), pp.86-101.

[55] Leonard, T., Liu, S., Alamdar, M., Jin, H., Cui, C., Akinola, O.G., Xue, L., Xiao, T.P., Friedman, J.S., Marinella, M.J. and Bennett, C.H., 2022. Shape-dependent multi-weight magnetic artificial synapses for neuromorphic computing. *Advanced Electronic Materials*, *8*(12), p.2200563.

[56] Laborieux, A., Ernoult, M., Hirtzlin, T. and Querlioz, D., 2021. Synaptic metaplasticity in binarized neural networks. *Nature communications*, *12*(1), p.2549.

[57] Dhull, S., Al Misba, W., Nisar, A., Atulasimha, J. and Kaushik, B.K., 2024. Quantized magnetic domain wall synapse for efficient deep neural networks. *IEEE Transactions on Neural Networks and Learning Systems*.