Boosted Spin Channel Networks for Energy-efficient Inference

Ameya D. Patil, *Student Member, IEEE,* Sasikanth Manipatruni, *Member, IEEE,* Dmitri E. Nikonov, *Senior Member, IEEE,* Ian A. Young, *Fellow, IEEE,* and Naresh R. Shanbhag, *Fellow, IEEE*

***Abstract*—Computational scaling beyond silicon electronics based on Moore's law requires the adoption of alternate state variables such as electronic spin. Multiple research efforts are underway exploring both Boolean and non-Boolean design space using spin devices in order to make their energy and delay benefits competitive to CMOS. In this paper, we propose *spin channel networks (SCN)*, where the exponential decay property of spin current along the spin channel is exploited to achieve energy-efficient dot product implementation for inference applications. As the use of exponentially decaying spin current for analog computation enforces severe locality constraints, we employ Adaptive Boosting (AdaBoost) to design an ensemble of tiny spin channel networks that work in unison to solve any binary classification task. Such boosted spin channel networks achieve up to $112\times$ and $14\times$ higher energy-efficiency over conventional ASL-based and 20 nm CMOS designs, respectively.**

I. Introduction

EXPONENTIAL scaling of CMOS-based logic devices in accordance with Moore's law has enabled tremendous improvement in computational efficiency. However, as the channel lengths continue to reduce beyond a few tens of nanometers, the energy and delay benefits achievable via scaling have stagnated. On the other hand, emerging big data applications, such recognition, mining and synthesis, require significant amount of information processing, thus requiring energy and delay improvements in computing systems. This has led to much interest in exploring the use of alternative state variables such as electron spin [1], [2] for computing. Spin torque devices store information in terms of aligned magnetic moments (spins) of unpaired electrons in nanomagnets and rely upon spin diffusion in non-magnetic metallic channel connecting two nanomagnets for information transfer.

One example of spin-based devices proposed for digital logic computation is the all spin logic (ASL) device [3]. ASL devices offer certain unique advantages such as non-volatility, high logic efficiency and ultra-low operating voltages, and are considered a promising beyond-CMOS alternative when combined with material improvements [4]. However, ASL is found out to be non-competitive compared to CMOS in terms of energy consumption and delay of digital logic implementations [5], [6], mainly due to the large energy and delay required for deterministic nanomagnet switching, the exponential decay of spin alignment during propagation along the spin interconnect, and because ASL gates consume static power [6], [7].

There have been few works that exploit the stochastic nature of nanomagnetic switching to achieve energy-efficiency. In [8], the switching behavior of magnets in super-paramagnetic regime was shown to resemble the dynamics of a Boltzmann machine, and thus a nanomagnetic network was trained to implement certain inference tasks. Stochastic magnets were also employed for energy-efficient random number generation [9] in stochastic computing, as well as for spike generation [10] in spiking neural network implementation.

Several research efforts have explored the neuromorphic design space using spin-devices in order to achieve energy-efficiency. While it is clear that the switching of nanomagnet naturally implements thresholding function of a neuron, these approaches use additional devices (resistive memory [11], domain wall magnets [12]–[14]) to achieve synaptic weighing of spin currents feeding into the nanomagnet. In [15], multiple binary weighted CMOS drivers along with a clever nanomagnet configuration were employed to obtain spin-current weighing in cellular neural network implementation.

There exists work at the architectural-level to fully exploit the advantages of emerging spin-device configurations such as racetrack memory [16], [17]. For example, high area-efficiency and serial access of racetrack memory was exploited to achieve reconfigurable precision [18] and efficient logic operations [19]. In [20], a novel data converter design was proposed by exploiting the serial structure of racetrack memory devices.

Recently, [21] took a physics-based approach for examining the power dissipation in spintronic switches. They identified that nanomagnetic switching consumes $10^3\times$-to-$10^4\times$ higher switching charge (Q_{sw}) compared to the CMOS inverter of comparable size. Such large gap in the switching charge requirements underscores the fundamentally expensive nature of the nanomagnetic switching. ASL networks (Fig. 1(a)) use nanomagnetic switching at the output of every gate to implement digital logic. Hence, they require switching of a large number intermediate nanomagnets leading to a high energy consumption.

In this paper, we propose *spin channel networks (SCN)* (Fig. 1(b)), where all intermediate nanomagnets are eliminated and all input nanomagnets contribute to the charge required to switch a single output nanomagnet to represent final decision, thereby amortizing the energy consumed in switching it. It is particularly suited for inference implementations. While elimination of intermediate nanomagnetic switching is expected to enhance the energy-efficiency, it also presents following two

Ameya D. Patil and Naresh R. Shanbhag are with the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, 61801 USA.

Sasikanth Manipatruni, Dmitri E. Nikonov, and Ian A. Young are with Component Research, Intel Corp.

This work was supported in part by Systems on Nanoscale Information fabriCs (SONIC), one of the six SRC STARnet Centers, sponsored by MARCO and DARPA.

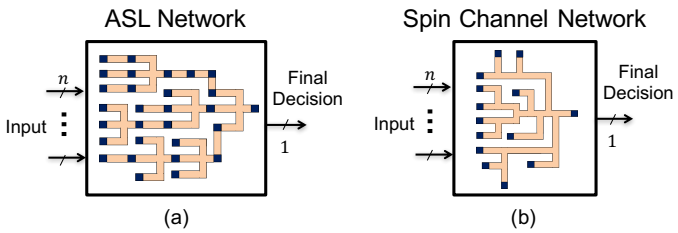

Fig. 1. Illustration of (a) an all spin logic (ASL) network, and (b) a spin channel network (SCN) implementing an inference kernel by mapping a large n-bit input vector to 1-bit decision.

key challenges:

1) how does one realize arbitrary computation while accumulating analog spin currents from multiple nanomagnets?

2) will this approach scale with the input vector dimensionality (complexity) of the inference kernel?

To address challenge 1), we show that the exponential decay property of spin current along the spin channel, a disadvantage in digital ASL networks, can be exploited to achieve energy-efficient analog dot product implementation. To circumvent the challenge 2) we employ Adaptive Boosting (AdaBoost) [22] framework to design multiple isolated tiny spin channel networks (t-SCNs) that work in unison to solve an arbitrary binary classification task. Such boosted t-SCNs achieve $112\times$-to-$22.5\times$ and $14\times$-to-$2.5\times$ higher energy-efficiency over conventional ASL-based and 20 nm CMOS designs, respectively, when realizing 10-to-100 dimensional binary classifiers.

Rest of the paper is organized as follows. Section II gives the relevant background about ASL, support vector machine (SVM), and classifier ensemble designs via AdaBoost, while Section III focuses on the design of SCNs. Section IV describes the SCN-based SVM and the boosted t-SCN implementations. Section V presents the simulation results and VI concludes this paper.

II. Background

A. All Spin Logic Device

Figure 2 shows the schematic of an ASL device. It consists of two nanomagnets separated by a conducting channel of length L. The input magnet (M_{in}) polarizes the supply current passing through it. This creates a spin concentration gradient and propagates a spin current in the channel of length L. This spin current, in turn, exerts a torque on the magnetization of the output magnet (M_{out}) forcing it to switch. Since the magnets are non-volatile, they retain the magnetization vector state when the supply current is switched off.

Electrical current in the order of $10\,\mu A$-to-$100\,\mu A$ is required to generate sufficient spin current to switch the output magnet. Since the nanomagnets and the spin channel are metallic, the equivalent electrical resistance across the nanomagnetic stack is small (few Ωs), enabling these devices to operate at ultra-low supply voltages. However, the electrical current through the input magnet flows irrespective of output activity, causing high static energy consumption. Hence, [6], [23] propose to clock these devices via a MOSFET, operating in the linear region, which acts as a switch turning ON the ASL device only when it needs to process information as shown in Fig. 2.

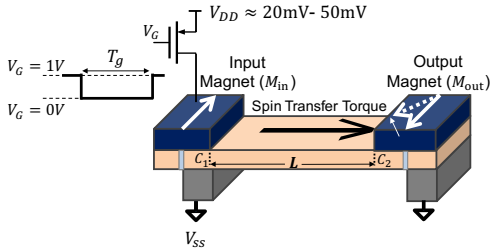

Fig. 2. All spin logic (ASL) device with a power gating transistor [6], [23].

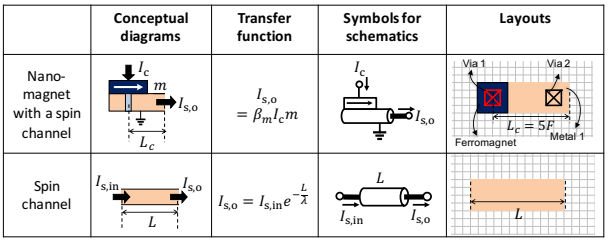

Fig. 3. SCN primitives derived from ASL: symbol, transfer function, and layout. Each layout grid cell is of size $\frac{F}{2} \times \frac{F}{2} = 7.5\,\text{nm} \times 7.5\,\text{nm}$ [25].

Figure 3 defines two SCN primitives (derived from ASL) and their functionalities as will be used to design SCNs in section III. The primitives are nanomagnet with a spin channel and a spin channel. In particular, nanomagnet takes input charge current I_c and injects proportional spin current $I_{\text{s,o}}$ in the channel, where β_m is a proportionality constant that depends upon the device material and geometry, including the channel length L_c. The input spin current $I_{\text{s,in}}$ into a spin channel is reduced by a factor of $e^{\frac{L}{\lambda}}$ to generate an output spin current $I_{\text{s,o}}$, where L denotes channel length, and λ is the spin flip length [7], [24]. The layouts are obtained by following the λ-rules in [5].

B. Support Vector Machine (SVM)

A linear SVM [26] is a simple and popular machine learning algorithm for binary classification. The SVM learns a hyperplane to separate the training feature vectors into two regions as shown below:

$$\mathbf{w}^T\mathbf{x} + b \underset{\hat{y}=-1}{\overset{\hat{y}=1}{\gtrless}} 0 \tag{1}$$

where $\mathbf{w}$ and b denote the trained weight vector and bias representing the separating hyperplane, respectively, $\mathbf{x}$ denotes the N-dimensional input feature vector, and $\hat{y}$ denotes its label predicted by the SVM. If the true label is denoted by y, the accuracy of SVM is given by the probability of classification error $p_e = \Pr\{\hat{y} \neq y\}$, which can be empirically estimated for a given dataset.

C. Classifier Ensemble via Adaptive Boosting (AdaBoost)

A classifier ensemble consists of multiple weak classifiers. Each weak classifier is computationally simple but inaccurate, i.e., with p_e close to 0.5. However, decisions of the weak classifiers can be combined to obtain a highly accurate final decision. Adaptive boosting (AdaBoost) [27] is a technique to train these weak classifiers sequentially. Each weak classifier is specifically trained to correct errors made by the other weak classifiers trained earlier (see [27] for the training algorithm). Let the output label of ith weak classifier be denoted as $\hat{y}_i = f_{\mathbf{w}_i}(\mathbf{x})$, where $f_{\mathbf{w}_i}(.)$ denotes the ith weak classifier

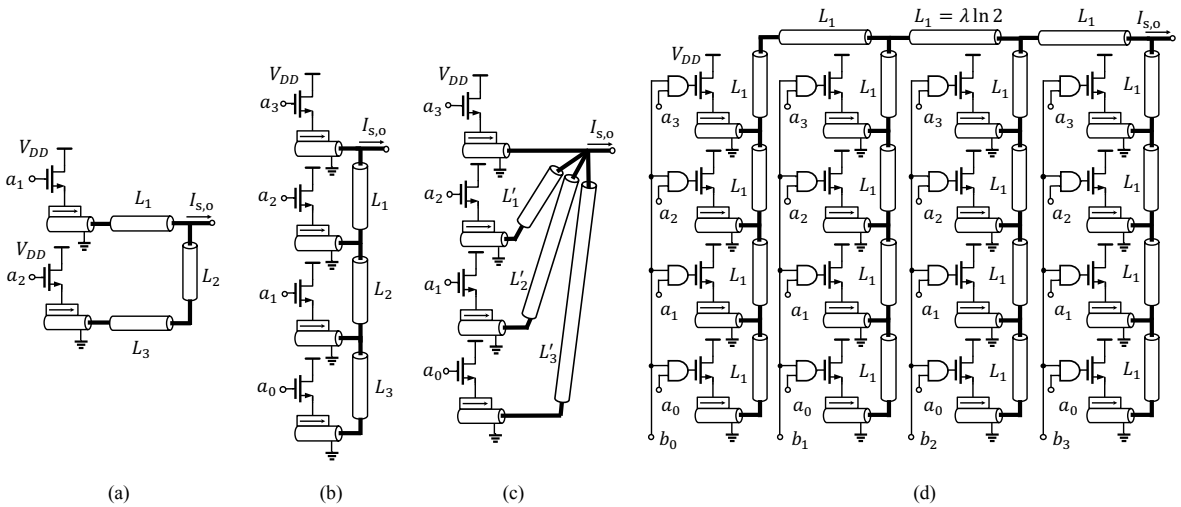

Fig. 4. Conceptual spin channel network topologies: (a) basic, (b) *ladder*, (c) *star*, and (d) a *ladder-of-ladders* topology of a 4×4 bit spin channel network multiplier (SCNM).

function parametrized by weight vector $\mathbf{w}_i$, which is computed during training. The final decision $\hat{y}_f$ is computed by linearly combining the weak classifier decisions $\hat{y}_i$, followed by thresholding as shown below:

$$\sum_{i=1}^{M} \alpha_i \hat{y}_i \underset{\hat{y}_f=-1}{\overset{\hat{y}_f=1}{\gtrless}} 0 \tag{2}$$

where output weights α_is of the linear combiner are also learned during the training phase.

III. Spin Channel Networks

A. Basic Concept

Spin channel networks exploit the exponential decay of spin current along spin channels for efficient computation. They compute via weighted analog accumulation of spin currents by careful choice of spin channel lengths. SCNs are composed of the two primitives defined in Fig. 3. The most basic SCN consists of two nanomagnets connected using spin channels having different lengths is shown in Fig. 4(a). The resulting output spin current $I_{\text{s,o}}$, is approximately given by

$$I_{\text{s,o}} = \beta_m I_c \left(a_1 m_1 e^{\frac{-L_1}{\lambda}} + a_2 m_2 e^{\frac{-(L_2+L_3)}{\lambda}}\right) \tag{3}$$

where $a_1, a_2 \in \{0, 1\}$ are the digital Boolean inputs, $m_1, m_2 \in \{-1, 1\}$ denote the directions of magnetization vectors of two nanomagnets, λ denotes spin flip length, I_c denotes the ON current of the NMOS transistors. Each bit a_i controls the charge current through one nanomagnet, and the corresponding spin current in weighed by a factor exponential in its channel length. More complex SCNs can be designed to achieve weighted accumulation of spin currents from M nanomagnets placed at lengths L_is, where $i \in \{0, \ldots, M-1\}$.

B. SCN Topologies

For $M > 2$, multiple circuit topologies can achieve the same input-to-output transfer function (up to a scaling constant) depending upon how the nanomagnets and spin channels are interconnected. For example, conceptual diagrams of two extreme topologies, namely the *ladder* and the *star* topology, are shown for $M = 4$ in Fig. 4(b) and 4(c), respectively. In the ladder topology, all nanomagnets share a single spin channel that connects them to the output node. The star topology, on the other hand, consists of a unique channel connecting each nanomagnet to the output node.

Weighted accumulation of spin currents in SCNs can be used to efficiently implement multiplication in analog. The $M \times N$ bit SCN multiplier (SCNM) in Fig. 4(d) takes two charge-domain digital operands A and B having bitwidths M and N, respectively, and generates an output spin current proportional to their product $A \times B$. The SCNM consists of $M \times N$ input nanomagnets, each contributing spin current corresponding to a partial product. Individual partial products are computed by the AND gates with bits a_is and b_js of operands A and B, respectively. The AND gates drive the gate of the NMOS, thereby controlling the input charge current through the SCN nanomagnets. Fig. 4(d) shows a *ladder-of-ladders* topology of a 4×4 bit SCNM, where four vertical ladders are connected horizontally in a ladder topology. It is to be noted at all the spin channel lengths are multiples of $\lambda \ln 2$ in order to achieve appropriate weighing (in the powers of two) of spin currents corresponding to individual partial products. The output spin current of this 4×4 bit SCNM is given by:

$$I_{\text{s,o}} = I_{\text{s,lsb}} \sum_{i,j=0}^{3} a_i b_j 2^{(i+j)} \tag{4}$$

where the unit spin current $I_{\text{s,lsb}}$ corresponds to the least significant partial product for a given NMOS ON current I_c. For the SCNM in Fig. 4(d), $I_{\text{s,lsb}} = \frac{\beta_m I_c}{2^7}$. The signs of these operands can be accounted for by changing the magnetization vector directions of the corresponding magnets (for A), and by using a differential supply [15] (for B). The energy consumption of such an SCNM is given by:

$$E_{\text{mult}}(A, B) = \left[I_c^2 T_g (R_{\text{spin}} + R_{\text{mos}}) + C_g V_g^2 + E_{\text{and}}\right] \sum_{i,j=0}^{3} a_i b_j \tag{5}$$

where R_{spin} denotes the series resistance of the nanomagnet and channel, E_{and} denotes the energy consumed in switching

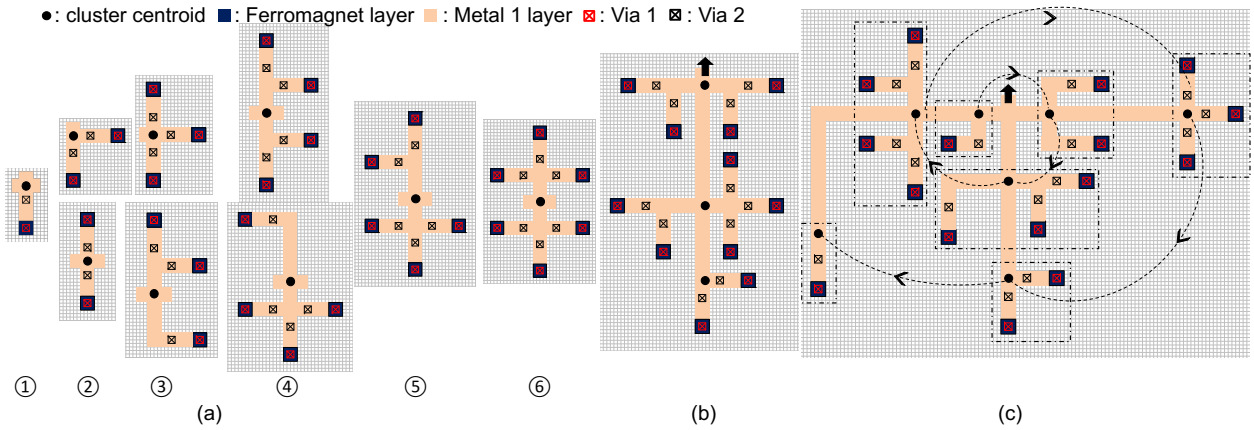

Fig. 5. Layouts of spin channel networks: (a) a set of clusters along with their centroids, (b) a ladder topology of three clusters, (c) a star-of-ladders layout topology of a 4×4 bit spin channel network multiplier. The layout grid cell is of size $7.5\,\text{nm} \times 7.5\,\text{nm}$ [25].

of the AND gate, while R_{mos} and C_g denote the ON resistance and gate capacitance of the transistor, respectively. The gate voltage V_g is applied to switch ON the NMOS for T_g duration.

The other topologies such as *star-of-ladders*, *star-of-stars* and *ladder-of-stars* are also possible, and this topological degree of freedom will be explored while identifying the energy-efficient layout in the following subsection III-C.

C. Hierarchical Layout Construction

Topological schematic diagrams in Fig. 4 are idealized and convey the SCN functionality at a very high level. They neither account for spin current branching at the spin channel junction, nor the physical constraints of component placements and maintaining spin channel lengths. We address both of these issues by developing precise layouts for SCN circuits, and obtain input-to-output transfer function from SCN layouts.

For layouts, we choose $F = 15\,\text{nm}$, where F denotes the DRAM half pitch [5], [25]. All SCN layouts need to satisfy λ-rules described (in terms of F) in [25]. For example, the layout pitch between any two contacts needs to be at least $4F$. The value of channel length L_c turns out to be $5F$ (in Fig. 3) as a direct consequence of λ-rule constraints. Similarly, λ-rules impose constraints on minimum distance between nanomagnet and a spin channel, and two parallel spin channels.

The layouts of more complex SCNs are particularly challenging, since there is a trade-off between satisfying λ-rule constraints and the magnitude of the output spin current, and hence the energy consumption. We propose a hierarchical construction of SCN layouts. We define nine primitive topologies referred to as *clusters* (see Fig. 5(a)). Each nanomagnet in a cluster contributes identical spin currents to the output node, referred to as the *cluster centroid*. The layouts of the clusters are fixed per λ-rules. These clusters can be connected in various topologies, such as a ladder, star or a ladder-of-stars, to generate layouts of more complex SCNs in a hierarchical manner. An illustrative ladder topology of three clusters is shown in Fig. 5(b). Once the clusters are connected, only the lengths between their centroids need to be adjusted to achieve appropriate weighing of the corresponding spin currents and simultaneously satisfy λ-rules.

Figure 5(c) shows a star-of-ladders layout topology of a 4×4 bit SCNM. Each cluster centroid J_k generates spin current

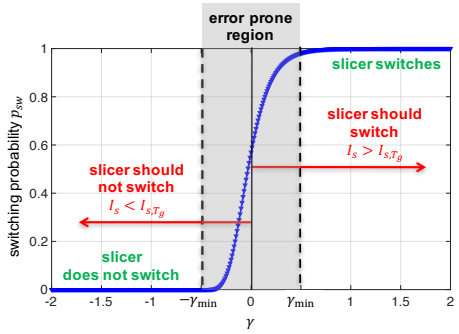

Fig. 6. The switching probability p_{sw} of stochastic slicer as a function of γ (given approximately by (8)) with $E_b = 35kT$ and T =300 K. The dotted outlines of the gray region mark the minimum energy operating point (MEOP) at $\gamma_{\min} = 0.5$ corresponding to slicer switching accuracy of 99%.

corresponding to p_k, where p_k is defined as the sum of partial products having identical binary weight k as follows:

$$p_k = \left(\sum_{\substack{i,j=0 \\ i+j=k}}^{3} a_i b_j \right) \tag{6}$$

The final output spin current in (4) can be computed as:

$$I_{\text{s,o}} = I_{\text{s,lsb}} \sum_{k=0}^{6} 2^k p_k \tag{7}$$

where the binary weighing of 2^k among the spin current contributions is achieved by adjusting spin channel lengths between them. The clusters are sequentially placed in a spiral order along three ladders as shown in Fig. 5(c). Thus, along a single ladder, the minimum channel length between any two consecutive clusters corresponds to the spin current weighing of 2^{-3}, thus allowing sufficient spacing to satisfy λ-rules. Hence, this layout topology effectively spreads out the nanomagnets radially. The actual channel lengths in the layout are chosen via extensive simulations using SPICE-based circuit models of spin current injection and propagation in spin devices [24] in order to account for spin current branching at the spin channel junctions.

D. Stochastic Slicer

The output of SCN circuits is an analog spin current. We use a nanomagnet as the final decision device. It acts as a sink for the spin current and thresholds it to produce the

final decision represented by its magnetization vector. The nanomagnetic switching is stochastic due dominant thermal noise in the nanomagnet [28]. Hence, we refer to such decision generating nanomagnet as a *stochastic slicer*. The stochastic slicer switches when the magnetization direction of the corresponding nanomagnets flips due to the input spin current.

In this work, we operate the stochastic slicer for the fixed duration of T_g. If the slicer switches during this duration, it corresponds to final decision $\hat{y} = 1$, otherwise, $\hat{y} = -1$. The slicer is reset after every decision. For a given duration T_g, the probability that slicer switches p_{sw} can be approximated as a function of its input spin current I_s as follows:

$$p_{\text{sw}}(I_s) \approx \left(\frac{1}{2}\right)^{\left[\left(\frac{\beta_1}{\ln 2}\right)^{-\gamma}\right]} \tag{8}$$

where $\gamma = \left(\frac{I_s - I_{s,T_g}}{I_{s,T_g}}\right)$, I_{s,T_g} denotes the spin current for which $p_{\text{sw}}(I_{s,T_g}) \approx 0.5$, and $\beta_1 = \frac{\pi^2 E_b}{4kT}$. In particular, E_b denotes the energy barrier of the nanomagnet, while k and T denote Boltzmann constant and absolute temperature, respectively. The spin current value I_{s,T_g} is a device dependent constant for a given switching duration T_g. The (8) is a good approximation of the p_{sw} expression in [28], when $|I_s|, I_{s,T_g} \gg I_{\text{crit}}$, where I_{crit} denotes the minimum spin current required for nanomagnet to switch with probability 1 as $T_g \to \infty$. The stochastic slicer strives realize the thresholding operation:

$$I_s \underset{\hat{y}=-1}{\overset{\hat{y}=1}{\gtrless}} I_{s,T_g} \tag{9}$$

However, due its stochastic nature, stochastic slicer probabilistically makes switching errors, i.e., it switches with certain non-zero probability even when $I_s < I_{s,T_g}$, and vice versa as shown in Fig. 6. Thus, there exists a trade off between input spin current magnitude I_s (proportional to energy consumption) and switching probability p_{sw} (see (8)). There exists a minimum energy operating point (MEOP) for a target switching probability. For example, as shown in Fig. 6, $|I_s| > 1.5 I_{s,T_g}$ to achieve slicer switching accuracy of 99%. This MEOP of slicer dictates the minimum charge current $I_{c,\text{min}}$ through each nanomagnet required for SCN-based binary classifier to achieve certain classification error probability p_e. For a fixed decision delay and error probability, MEOP for SCN-based binary classifiers is uniquely defined by the value of $I_{c,\text{min}}$ as shown in section V.

E. CMOS Driver

Figure 7(a) and (b) show the abstract model and transistor-level schematic of the CMOS input driver in a 14 nm technology, respectively. The nanomagnet is controlled by an NMOS, which should switch ON only when $a_i = 1$ and $b_j = 1$. This is achieved via a CMOS NOR gate driving the gate of the NMOS $N1$ as shown in Fig. 7(b). The inputs to the NOR gate are driven by identical inverters, who receive ideal step inputs. The NMOS $N1$ is sized to provide a charge current of $I_{c,\text{min}}$, while satisfying $V_{DS} < 10\,\text{mV}$. The gate voltage of $N1$ gets raised to 600 mV, turning it ON in the linear region with overdrive voltage $\geq 350\,\text{mV}$. The NOR gate is sized so that

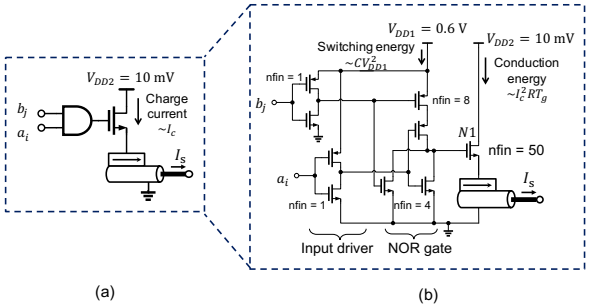

Fig. 7. Detailed schematic of the CMOS input driver designed and simulated using 14 nm HP FinFET ASU predictive technology models [29], where nfin denotes number of fins in the FinFET [30].

the CMOS driver switches within 50 ps while driving $N1$ and the inverters are minimum sized. We simulate this schematic using 14 nm HP FinFET ASU predictive technology models [29] to estimate its switching delay and energy consumption.

IV. Design of SCN-based Classifiers

A. Linear Support Vector Machine (SVM) Classifier

A linear SVM classifier can be realized using the proposed SCNs by connecting multiple SCNMs in parallel (Fig. 8(c)) and one stochastic slicer to generate final classification decision. The SCNM and slicer symbols are defined in Fig. 8(a) and (b), respectively. Each multiplier generates spin current corresponding to the product $x_i w_i$, where x_i and w_i denote ith dimension of input feature vector $\mathbf{x}$ and the weight vector $\mathbf{w}$, respectively. Both x_i and w_i are fixed-point binary numbers. The spin currents at the output of SCNMs accumulate in a common channel (generating $I_{\text{s,o}}$) and feed into the stochastic slicer.

The inputs are kept ON for the duration T_g since the stochastic slicer requires that much time to produce its decision. Noting that the stochastic slicer thresholding involves a comparison of its input spin current with I_{s,T_g}, (1) of SVM can be realized as follows:

$$I_{\text{s,o}} = (\mathbf{w}^T\mathbf{x} + b) I_{\text{s,lsb}} + I_{\text{s,bias}} \underset{\hat{y}=-1}{\overset{\hat{y}=1}{\gtrless}} I_{s,T_g} \tag{10}$$

where $I_{\text{s,bias}}$ is additional bias current.

When $I_{\text{s,o}} = I_{s,T_g}$, the slicer switches with probability 0.5. In SVM, this operating point would occur when the input feature vector $\mathbf{x}$ lies on the classifier hyperplane, i.e. when $\mathbf{w}^T\mathbf{x} + b = 0$, resulting in

$$(I_{\text{s,o}})|_{\mathbf{w}^T\mathbf{x}+b=0} = I_{s,T_g} = I_{\text{s,bias}} \tag{11}$$

To avoid large bias currents, we modify the feature vector $\mathbf{x}$ to $(\mathbf{x} + d\mathbf{1})$, where $\mathbf{1}$ denotes all-one vector and d is a constant, thereby transforming (10) into

$$\left[\mathbf{w}^T(\mathbf{x} + d\mathbf{1})\right] I_{\text{s,lsb}} + I_{\text{s,bias}} \underset{\hat{y}=-1}{\overset{\hat{y}=1}{\gtrless}} I_{s,T_g} \tag{12}$$

where $I_{\text{s,bias}}$ is now given by:

$$I_{\text{s,bias}} = I_{s,T_g} + b I_{\text{s,lsb}} - \left[d\left(\sum_i w_i\right)\right] I_{\text{s,lsb}} \tag{13}$$

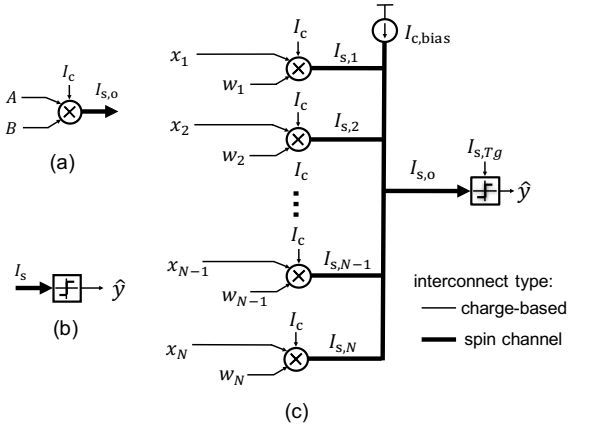

Fig. 8. SCN-based linear SVM classifier: (a) SCNM symbol, (b) stochastic slicer symbol, and (c) the SCN-based N dimensional linear SVM classifier architecture.

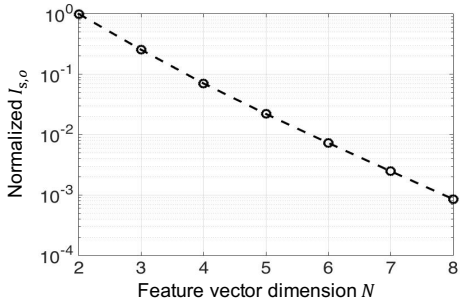

Fig. 9. Normalized magnitude of $I_{s,o}$ in Fig. 8(c) as a function of classifier dimensionality N, when $x_1 = c_1 \neq 0$, $w_1 = c_2 \neq 0$, and $x_i = w_i = 0$ for all $i \in \{2, \ldots, N\}$, where c_1 and c_2 are some constants.

The bias current $I_{s,bias}$ is generated by having an additional magnet with a supply current $I_{c,bias}$ as shown in Fig. 8. If the signed precision of x_i is M bits, we choose d to be 2^{M-1}. This makes $(x_i + d)$ an unsigned number, removing the need for differential supply. The sign of $\mathbf{w}$ is accounted for by changing the magnetization vector directions of the corresponding spin-current injecting nanomagnets appropriately.

B. Classifier Dimensionality Scaling via Boosted Tiny SCNs (t-SCNs)

The exponential decay of spin current exploited in SCNM makes it very hard to route the output spin current to another block as doing so inevitably incurs a significant loss in the spin current magnitude. This severely limits the ability to scale the classifier dimensionality (Fig. 8(c)), which requires the N multiplier output spin currents to be routed to a single stochastic slicer. Assuming a circular layout of N SCNMs with the slicer at its center, we estimate the loss in the spin current $I_{s,o}$ magnitude as a function of classifier dimensionality N as shown in Fig. 9, when $x_1 = c_1 \neq 0$, $w_1 = c_2 \neq 0$, and $x_i = w_i = 0$ for all $i \in \{2, \ldots, N\}$, where c_1 and c_2 are some constants. Recall from Fig. 6 that the stochastic slicer requires minimum magnitude of the input spin current in order to operate accurately for a given switching delay. Thus, the exponential loss in the output spin current magnitude results in exponentially increasing classifier energy consumption in order to maintain classifier accuracy.

In order to address the problem, we limit the dimensionality of the SCN-based linear SVM classifier to only two dimensions, and refer to the resulting design as *tiny SCN* (t-SCN). We then employ AdaBoost to design an ensemble of multiple such t-SCNs to implement an arbitrary N-dimensional binary classification task. Figure 10 shows the boosted t-SCNs architecture. In particular, given an N-dimensional input feature vector, each t-SCN observes only two unique feature dimensions and computes its local decision $\hat{y}_i$. These local decisions could be inaccurate with higher probability, and hence are referred to as *weak decisions*. The final weighted sum block combines these weak decisions to obtain the final decision $\hat{y}_f$ as per (2). We restrict the number of weak classifiers to $\frac{N}{2}$ so that computational complexity of the boosted t-SCNs architecture is similar to the standard N dimensional linear SVM implementation (Fig. 8(c)).

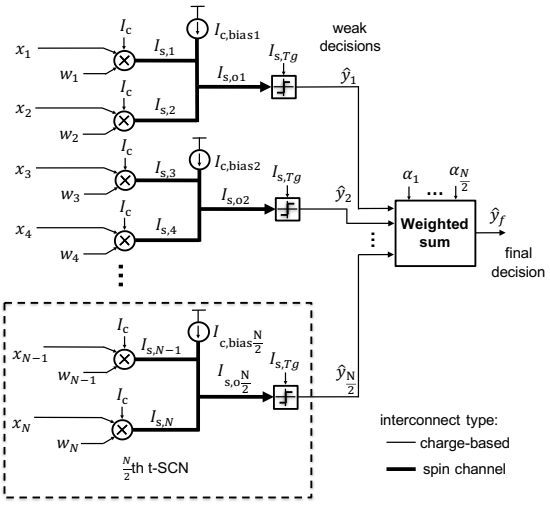

Fig. 10. Boosted tiny SCNs architecture: adaptively boosted ensemble of $\frac{N}{2}$ tiny SCNs (t-SCNs), where each t-SCN consists of 2 SCNMs in parallel and one stochastic slicer to implement a 2-dimensional linear SVM classifier.

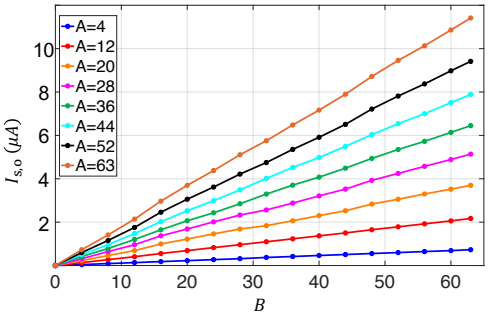

Fig. 11. Simulated transfer function of a 6×6 bit SCNM realizing $A \times B$.

It is important to note that, in the boosted architecture, the output spin current of the channel network gets processed locally, and only the binary weak decisions are routed to the final weighted sum block, thus requiring much shorter spin interconnect routing within each t-SCN. It is straightforward to convert the binary slicer decisions $\hat{y}_i$ $i \in \{1, \ldots, \frac{N}{2}\}$ to equivalent voltage [8] and then route it using charge interconnects. We designed the linear combiner in (2) in conventional digital 14 nm CMOS. Its complexity in terms of the full adder count is less than 5% of the total complexity of the $\frac{N}{2}$ t-SCNs. One can also employ other schemes, such as Boolean logic, Winner-Take-All, to efficiently combine binary t-SCN outputs, achieving similar energy benefits.

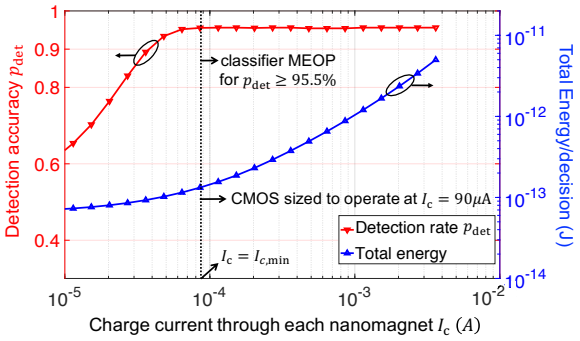

Fig. 12. Accuracy p_{det} and total energy vs. I_c tradeoff for 10-dimensional boosted t-SCN classifier operating at a final decision delay of 3 ns. The minimum energy operating point (MEOP) is achieved at $I_{c,\text{min}} = 90\,\mu\text{A}$.

V. Simulation Results

We design and characterize a 6×6 bit SCNM using the SPICE-based spin device models [24] for the material and device parameters provided in supplementary information section II. Fig. 11 shows the interpolated SCNM transfer function after carrying out detailed SPICE simulations for 289 different A and B values. The observed $(\frac{\sigma}{\mu})$ of the deviations from ideal output spin current is 2%. We employ this SCNM transfer function in our system-level simulations to estimate the accuracy of SCNM classifiers, and use benchmarking methodology [5] to estimate energy and delay of all classifier implementations. The simulation methodology is described in supplementary information section I.

We demonstrate the effectiveness of proposed approach for two classification tasks: 1) 10-dimensional (10D) breast cancer detection (UCI repository dataset [31], [32]), and 2) 100-dimensional (100D) face detection (MIT CBCL dataset [33]). We quantify classification accuracy in terms of detection rate $p_{\text{det}} = 1 - p_e$, where p_e is classification error probability.

For each t-SCN, there exists a tradeoff between NMOS current I_c and weak decision delay T_g for fixed p_{sw}. We choose $T_g = 2.5\,\text{ns}$ throughout this paper to make sure that CMOS driver switching energy $\leq \approx 33\%$ of the total energy. We compare the energy consumption and accuracy of 10D and 100D classifier implementations at a fixed final decision delay of 3 ns and 4 ns, respectively. The remaining duration accounts for the delay of CMOS driver switching, slicer reset, and weighted logic block operation. In particular, CMOS driver can be switched within 50 ps. For 10D classifier, weighted logic block operation can be approximated as a majority operation. We choose identical I_c for all weak classifiers. Given I_c and T_g, $I_{\text{c,bias}}$ is chosen according to (13) for each weak classifier.

For a fixed final decision delay (of 3 ns), the tradeoff between the accuracy and total energy consumption of the 10D boosted t-SCN classifier is shown in Fig. 12 as a function of I_c. As expected, both accuracy and total energy decrease with I_c. The accuracy degradation occurs due to reductions in p_{sw}s of the stochastic slicers. For accuracy of 95.5%, the classifier MEOP (defined in Sec. III-D) is achieved at $I_{c,\text{min}} = 90\,\mu\text{A}$. Hence, we size the NMOS $N1$ to provide I_c of $90\,\mu\text{A}$ at $V_{DD2} = 10\,\text{mV}$ (see Fig. 7).

Fig. 13 shows the accuracy vs energy tradeoff for different 10D classifier implementations. Boosted t-SCN classifier

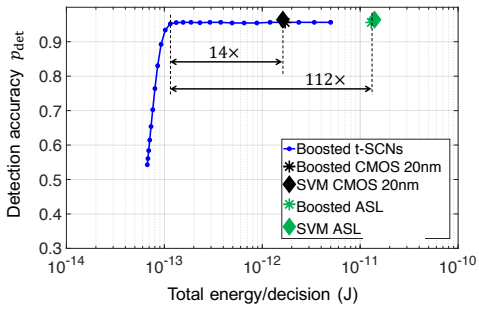

Fig. 13. Classification accuracy p_{det} vs. energy tradeoff for different 10-dimensional classifier implementations operating at a final decision delay of 3 ns.

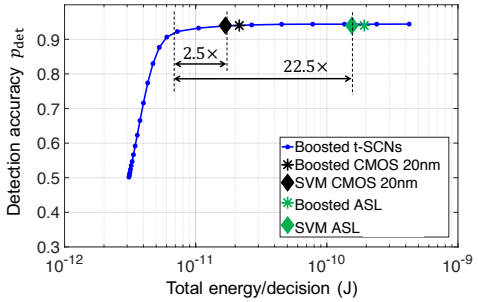

Fig. 14. Classification accuracy p_{det} vs. energy tradeoff for different 100-dimensional classifier implementations operating at a final decision delay of 4 ns.

achieves at least $112\times$ lower energy per decision compared to that of the conventional boosted ASL implementation, while maintaining accuracy. Such large energy savings can be attributed to the elimination of all intermediate switching nanomagnets in the spin channel network implementation. It also achieves $14\times$ lower energy compared to boosted 20 nm LV CMOS digital implementation, while operating at the identical final decision delay. We also observe that both boosted CMOS and boosted ASL implementations achieve energy consumption similar to the corresponding N-dimensional linear SVM implementations. For 100D classifier (Fig. 14), the energy benefits of boosted t-SCN implementation reduce to $2.5\times$ and $22.5\times$ over CMOS and ASL SVM implementations, respectively. This is primarily because of higher I_c requirements for its weak SVM classifiers, resulting from lower class separability of the dataset. Thus, the energy benefits are a function of the input data statistics as well. We only compare dynamic energy here, but leakage energy will be the least for SCN implementations due to having fewer transistors compared to both CMOS and ASL implementations. For all ASL implementations, we assume that the clocking transistors are shared across multiple nanomagnets [6], significantly amortizing their energy consumption.

In Fig. 15, we observe that the CMOS driver conduction energy and switching energy are comparable, and together dominate the energy consumption of the 10D boosted SCN classifier. The CMOS driver is expensive to switch due to large size of NMOS $N1$, which is necessary due to large charge current requirements ($I_{c,\text{min}} \approx 100\,\mu\text{A}$) of SCN classifiers. Conduction energies of CMOS driver and nanomagnet add up to a constant $V_{DD2} I_{c,\text{min}} T_g$. These trends are similar to ASL implementations.

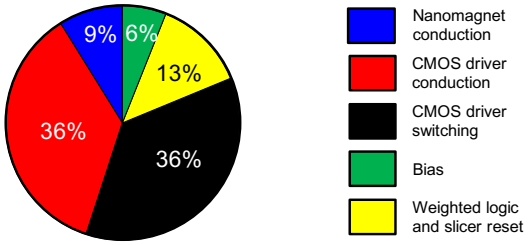

Fig. 15. Category-wise energy breakdown for 10-dimensional boosted t-SCN implementation operating at a decision delay of 3 ns and with $I_{c,\min} = 90\,\mu$A.

VI. Conclusion

In this paper, we proposed spin channel networks where multiple input nanomagnets contribute to the spin current required to switch a single decision nanomagnet. These networks exploit exponential spin current decay for efficient local computation to achieve very high energy-efficiency and ensemble of such isolated networks can solve any given classification task. Moving forward, one needs to evaluate the impact of process variations and temperature on the final decision accuracy of SCN-based classifiers. While inherent robustness of ML classifiers will help in mitigating this impact, one can also employ system-level techniques, such as retraining [34], Shannon-inspired error compensation [35], to achieve further robustness. We plan to explore this direction in future.

This work demonstrates how algorithmic techniques can be employed to take advantages of some device characteristics, such as exponential decay of spin current in ASL, that appear disadvantageous in conventional implementations.

References

[1] D. E. Nikonov, G. I. Bourianoff, and T. Ghani, "Proposal of a spin torque majority gate logic," *IEEE Electron Device Letters*, vol. 32, no. 8, pp. 1128–1130, 2011.

[2] S. Manipatruni, D. E. Nikonov, R. Ramesh, H. Li, and I. A. Young, "Spin-orbit logic with magnetoelectric nodes: A scalable charge mediated nonvolatile spintronic logic," *arXiv:1512.05428*, 2015.

[3] B. Behin-Aein, D. Datta, S. Salahuddin, and S. Datta, "Proposal for an all-spin logic device with built-in memory," *Nature nanotechnology*, vol. 5, no. 4, p. 266, 2010.

[4] S. Manipatruni, D. E. Nikonov, and I. A. Young, "Material targets for scaling all-spin logic," *Physical Review Applied*, vol. 5, no. 1, 2016.

[5] D. Nikonov and I. Young, "Benchmarking of beyond-CMOS exploratory devices for logic integrated circuits," *Exploratory Solid-State Computational Devices and Circuits, IEEE Journal on*, 2015.

[6] Z. Pajouhi, S. Venkataramani, K. Yogendra, A. Raghunathan, and K. Roy, "Exploring spin-transfer-torque devices for logic applications," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 9, pp. 1441–1454, 2015.

[7] J. Kim, A. Paul, P. A. Crowell, S. J. Koester, S. S. Sapatnekar, J.-P. Wang, and C. H. Kim, "Spin-based computing: Device concepts, current status, and a case study on a high-performance microprocessor," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 106–130, 2015.

[8] K. Y. Camsari, R. Faria, B. M. Sutton, and S. Datta, "Stochastic p-bits for invertible logic," *Physical Review X*, vol. 7, no. 3, p. 031014, 2017.

[9] R. Venkatesan, S. Venkataramani, X. Fong, K. Roy, and A. Raghunathan, "Spintastic: Spin-based stochastic logic for energy-efficient computing," in *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition*. EDA Consortium, 2015, pp. 1575–1578.

[10] A. Sengupta, M. Parsa, B. Han, and K. Roy, "Probabilistic deep spiking neural systems enabled by magnetic tunnel junction," *Electron Devices, IEEE Tran. on*, 2016.

[11] M. Sharad, D. Fan, K. Aitken, and K. Roy, "Energy-efficient non-boolean computing with spin neurons and resistive memory," *IEEE Tran. on Nanotechnology*, 2014.

[12] M. Sharad, C. Augustine, G. Panagopoulos, and K. Roy, "Spin-based neuron model with domain-wall magnets as synapse," *IEEE Transactions on Nanotechnology*, vol. 11, no. 4, pp. 843–853, 2012.

[13] S. G. Ramasubramanian, R. Venkatesan, M. Sharad, K. Roy, and A. Raghunathan, "SPINDLE: Spintronic deep learning engine for large-scale neuromorphic computing," in *Proceedings of the 2014 international symposium on Low power electronics and design*. ACM, 2014.

[14] A. Sengupta, Y. Shim, and K. Roy, "Proposal for an all-spin artificial neural network: Emulating neural and synaptic functionalities through domain wall motion in ferromagnets," *Bio. Cir. and Sys., IEEE Tran. on*, 2016.

[15] C. Pan and A. Naeemi, "A proposal for energy-efficient cellular neural network based on spintronic devices," *IEEE Transactions on Nanotechnology*, vol. 15, no. 5, pp. 820–827, 2016.

[16] S. Parkin and S.-H. Yang, "Memory on the racetrack," *Nature nano.*, vol. 10, no. 3, pp. 195–198, 2015.

[17] Z. Sun, X. Bi, A. K. Jones, and H. Li, "Design exploration of racetrack lower-level caches," in *Low Power Electronics and Design (ISLPED), 2014 IEEE/ACM International Symposium on*. IEEE, 2014.

[18] J. Chung, J. Park, and S. Ghosh, "Domain wall memory based convolutional neural networks for bit-width extendability and energy-efficiency," in *Proceedings of the 2016 International Symposium on Low Power Electronics and Design*. ACM, 2016, pp. 332–337.

[19] Y. Wang, H. Yu, L. Ni, G.-B. Huang, M. Yan, C. Weng, W. Yang, and J. Zhao, "An energy-efficient nonvolatile in-memory computing architecture for extreme learning machine by domain-wall nanowire devices," *IEEE Transactions on Nanotechnology*, vol. 14, no. 6, 2015.

[20] Q. Dong, K. Yang, L. Fick, D. Fick, D. Blaauw, and D. Sylvester, "Low-power and compact analog-to-digital converter using spintronic racetrack memory devices," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 3, pp. 907–918, 2017.

[21] S. Ganguly, K. Y. Camsari, and S. Datta, "Evaluating spintronic devices using the modular approach," *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, vol. 2, pp. 51–60, 2016.

[22] Y. Freund, R. E. Schapire *et al.*, "Experiments with a new boosting algorithm," 1996.

[23] V. Calayir, D. E. Nikonov, S. Manipatruni, and I. A. Young, "Static and clocked spintronic circuit design and simulation with performance analysis relative to cmos," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 2, pp. 393–406, 2014.

[24] P. Bonhomme, S. Manipatruni, R. M. Iraei, S. Rakheja, S.-C. Chang, D. E. Nikonov, I. A. Young, and A. Naeemi, "Circuit simulation of magnetization dynamics and spin transport," *IEEE Transactions on Electron Devices*, vol. 61, no. 5, pp. 1553–1560, 2014.

[25] D. E. Nikonov and I. A. Young, "Overview of beyond-cmos devices and a uniform methodology for their benchmarking," *Proceedings of the IEEE*, vol. 101, no. 12, pp. 2498–2533, 2013.

[26] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[27] R. E. Schapire, "Explaining adaboost," in *Empirical inference*. Springer, 2013, pp. 37–52.

[28] W. H. Butler, T. Mewes, C. K. Mewes, P. Visscher, W. H. Rippard, S. E. Russek, and R. Heindl, "Switching distributions for perpendicular spin-torque devices within the macrospin approximation," *IEEE Transactions on Magnetics*, vol. 48, no. 12, pp. 4684–4700, 2012.

[29] S. Sinha, G. Yeric, V. Chandra, B. Cline, and Y. Cao, "Exploring sub-20nm FinFET design with predictive technology models," in *Proceedings of the 49th Annual Design Automation Conference*. ACM, 2012.

[30] D. Hisamoto, W.-C. Lee, J. Kedzierski, H. Takeuchi, K. Asano, C. Kuo, E. Anderson, T.-J. King, J. Bokor, and C. Hu, "FinFET-a self-aligned double-gate mosfet scalable to 20 nm," *IEEE Transactions on Electron Devices*, vol. 47, no. 12, pp. 2320–2325, 2000.

[31] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[32] W. H. Wolberg and O. L. Mangasarian, "Multisurface method of pattern separation for medical diagnosis applied to breast cytology." *Proceedings of the national academy of sciences*, 1990.

[33] B. Heisele, T. Poggio, and M. Pontil, "Face detection in still gray images," Center for Biological and Computational Learning, MIT, Tech. Rep., 2000.

[34] Z. Wang, R. E. Schapire, and N. Verma, "Error adaptive classifier boosting (EACB): Leveraging data-driven training towards hardware resilience for signal inference," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 62, no. 4, pp. 1136–1145, 2015.

[35] N. R. Shanbhag, N. Verma, Y. Kim, A. D. Patil, and L. R. Varshney, "Shannon-inspired statistical computing for the nanoscale era," *Proceedings of the IEEE*, vol. 107, no. 1, pp. 90–107, 2019.

Supplementary Information: Boosted Spin Channel Networks for Energy-efficient Inference

Ameya D. Patil, *Student Member, IEEE,* Sasikanth Manipatruni, *Member, IEEE,* Dmitri E. Nikonov, *Senior Member, IEEE,* Ian A. Young, *Fellow, IEEE,* and Naresh R. Shanbhag, *Fellow, IEEE*

I. Simulation methodology

Our simulation methodology is shown in Fig. 1. We first develop SCNM layout and obtain its transfer function via SPICE-based simulations. We employ SPICE-based spin device models [1] for SCNM schematic simulations and use CAD drawing tools for corresponding layouts and λ-rule checks. We assume the material parameters provided in Supplementary Information Sec. II. The channel lengths between the clusters in the SCNM layouts are repeatedly adjusted until the appropriate spin current weighting is achieved and all λ-rules are satisfied. Fig. 11 in the main text shows the interpolated SCNM transfer function after carrying out detailed spice simulations for 289 different A and B values. The observed $(\frac{\sigma}{\mu})$ of the deviations from linearity in output spin current is 2%. Such good linearity was achieved, in part, because process variations (such as line edge roughness etc.) and length quantization between two clusters are not yet accounted for. However, this does indicate that there is still room to budget such impact of more variations within inherent tolerance of machine learning classifiers. Also, since all such variations within the layouts are static, one can employ retraining [2] to alleviate their impact. We leave this exploration for future work.

We use the SCNM transfer function and stochastic slicer model (based on (8) in the main text and analysis in [3]) for system-level classifier accuracy predictions. We estimate the classification error probability p_e of boosted t-SCN implementation in MATLAB. We use LIBSVM [4] to train all SVM classifiers. The classifier training always happens in floating-point precision in MATLAB. We then quantize the trained classifier model and test data to have 6 bit precision, which we found out to be sufficient. We carry out 6-fold cross validation over the available data to estimate the average accuracy of all classifier implementations.

We employ benchmarking methodology [5], [6] to estimate energy and delay for all classifier implementations compared in Fig. 13 and 14 in the main text. For ASL implementations, we assume ASL device consisting of nanomagnets with improved anisotropy reported in [7]. We also assume ASL gates to be clocked using a MOSFET (see Fig. 2 in the main text) to avoid static power consumptions. For digital CMOS implementations, we use the delay vs energy curve of 20 nm LV CMOS FO4 inverter reported in [7], and assume activity factor of 33%.

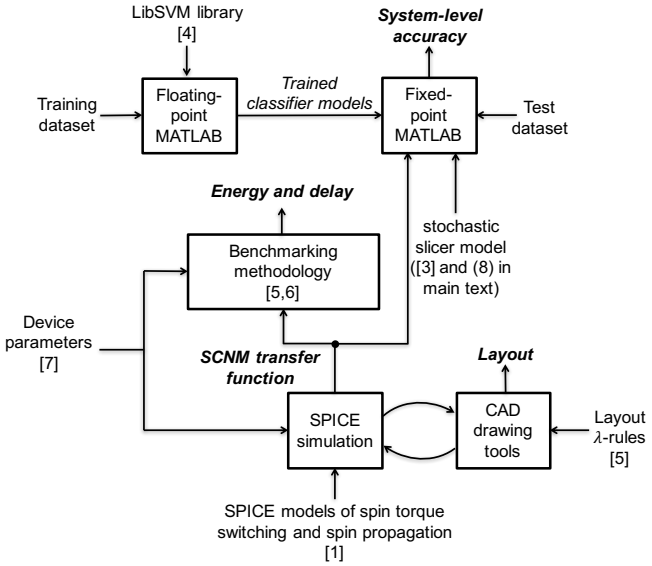

Fig. 1. Simulation methodology.

II. Material and device parameters

The SPICE-based models developed in [1] were employed to simulate all spin channel network designs in this paper. Hence, most of the device and material parameters of nanomangets and the Copper channel are chosen as mentioned in [1]. Few device parameters that were specifically chosen for the designs in this paper are given in the following table:

Variable	**Value**
Saturation magnetization [7]	250×10^3 A/m
Effective internal anisotropic field [7]	16×10^4 A/m
Damping coefficient [7]	0.007
Nanomagnet dimensions	30 nm×30 nm×10 nm
Slicer dimensions	30 nm×80 nm×2 nm
Spin channel width	30 nm
Spin channel thickness	100 nm

Ameya D. Patil and Naresh R. Shanbhag are with the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, 61801 USA.

Sasikanth Manipatruni, Dmitri E. Nikonov, and Ian A. Young are with Component Research, Intel Corp.

This work was supported in part by Systems on Nanoscale Information fabriCs (SONIC), one of the six SRC STARnet Centers, sponsored by MARCO and DARPA.

References

[1] P. Bonhomme, S. Manipatruni, R. M. Iraei, S. Rakheja, S.-C. Chang, D. E. Nikonov, I. A. Young, and A. Naeemi, "Circuit simulation of magnetization dynamics and spin transport," *IEEE Transactions on Electron Devices*, vol. 61, no. 5, pp. 1553–1560, 2014.

[2] Z. Wang, R. E. Schapire, and N. Verma, "Error adaptive classifier boosting (eacb): Leveraging data-driven training towards hardware resilience for signal inference," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 62, no. 4, pp. 1136–1145, 2015.

[3] W. H. Butler, T. Mewes, C. K. Mewes, P. Visscher, W. H. Rippard, S. E. Russek, and R. Heindl, "Switching distributions for perpendicular spin-torque devices within the macrospin approximation," *IEEE Transactions on Magnetics*, vol. 48, no. 12, pp. 4684–4700, 2012.

[4] C.-C. Chang and C.-J. Lin, "Libsvm: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[5] D. E. Nikonov and I. A. Young, "Overview of beyond-cmos devices and a uniform methodology for their benchmarking," *Proceedings of the IEEE*, vol. 101, no. 12, pp. 2498–2533, 2013.

[6] D. Nikonov and I. Young, "Benchmarking of beyond-CMOS exploratory devices for logic integrated circuits," *Exploratory Solid-State Computational Devices and Circuits, IEEE Journal on*, 2015.

[7] S. Manipatruni, D. E. Nikonov, and I. A. Young, "Material targets for scaling all-spin logic," *Physical Review Applied*, vol. 5, no. 1, 2016.