Printed Stochastic Computing Neural Networks

Dennis D. Weller[†], Nathaniel Bleier[$], Michael Hefenbrock[†], Jasmin Aghassi-Hagmann[§], Michael Beigl[†]
Rakesh Kumar[$] and Mehdi B. Tahoori[†]
[†]Karlsruhe Institute of Technology, [$]University of Illinois Urbana-Champaign, [§]Offenburg University of Applied Sciences

***Abstract*—Printed electronics (PE) offers flexible, extremely low-cost, and on-demand hardware due to its additive manufacturing process, enabling emerging ultra-low-cost applications, including machine learning applications. However, large feature sizes in PE limit the complexity of a machine learning classifier (e.g., a neural network (NN)) in PE. Stochastic computing Neural Networks (SC-NNs) can reduce area in silicon technologies, but still require complex designs due to unique implementation tradeoffs in PE. In this paper, we propose a printed mixed-signal system, which substitutes complex and power-hungry conventional stochastic computing (SC) components by printed analog designs. The printed mixed-signal SC consumes only 35% of power consumption and requires only 25% of area compared to a conventional 4-bit NN implementation. We also show that the proposed mixed-signal SC-NN provides good accuracy for popular neural network classification problems. We consider this work as an important step towards the realization of printed SC-NN hardware for near-sensor-processing.**

***Index Terms*—printed electronics, stochastic computing, neural networks, electrolyte-gated transistors**

I. Introduction

In recent years, new application domains such as near-sensor processing, smart-home devices, and smart packaging for fast moving consumer goods (FMCG) have emerged that have requirements related to fabrication costs, conformity, and time-to-market that cannot be met by conventional silicon-based electronics. For instance, item-level tagging of FMCG items using any smart device has to be at least as cheap as a barcode [1], even less than a cent, a requirement that cannot be met by silicon-based systems [2].

Printed electronics (PE) [3] promises to enable these applications, driven by ultra-low cost additive manufacturing processes that can produce conformal hardware practically on-demand. However, large feature sizes in PE (3 orders of magnitude larger compared to complementary metal-oxide-semiconductors (CMOS)) mean that conventional digital architectures for complex hardware have exorbitantly high hardware overheads [4]. This includes digital PE implementation of machine learning algorithms such as neural networks, which are promising candidates to perform classification tasks in future application domains as part of direct sensor processing in smart electronics [5]. The implementation of artificial neuron functions such as multiply-accumulate (MAC) and non-linear activation functions requires a large number of transistors leading to large area, making conventional neural network (NN) implementations infeasible in PE [5].

Fortunately, stochastic computing (SC) [6] can be leveraged to reduce the cost of NN implementations. SC is a low-cost alternative to digital computing, where signals are encoded as a sequence of random bits instead of deterministic multi-bit representations used in conventional hardware. As all operations are performed bit-wise and sequentially, hardware footprint and wiring costs of SC components are extremely low. For example, a multiplier can be implemented by a single XNOR gate with nine transistors, compared to hundreds of transistors for a low-precision digital multiplier.

SC has not experienced wide applicability in silicon-based electronics since performance and throughput overheads are high due to bit-wise sequential processing. Also, area is not a concern in silicon due to the small feature sizes. In contrast, high-performance is not a primary requirement of PE applications. For instance, sensor readouts occur only every few seconds [7] or even minutes [8] for many applications. Also, as feature sizes in PE are in the micrometer range, area is a major concern.

There has been no prior work on SC for printed electronics. As we show in this paper, existing stochastic computing neural network (SC-NN) designs cannot be mapped efficiently to PE, since corresponding activation functions and stochastic number generators (SNGs) require complex circuit designs which are currently infeasible in PE. We propose efficient implementations of analog printed activation functions and SNGs in this work, which require only a small amount of area and power compared to digital realizations. Proposed designs also enable direct sensor interfacing by converting analog input signals directly into stochastic numbers.

In this paper, we make the following contributions:

1) We propose a stochastic computing-based mixed-signal neural network architecture for printed electronics. This is the first study of stochastic computation for PE.
2) We evaluate conventional and proposed SC-NNs in PE. The printed mixed-signal SC-NN consumes only 35% of power consumption and requires only 25% of area compared to a conventional 4-bit NN implementation.
3) We validate the proposed mixed-signal SC-NN architecture on popular benchmark datasets. We show that it provides good accuracy for most of these problems.

II. Preliminaries

A. Printed Electronics

Printed electronics denotes a set of printing methods which can realize ultra low-cost, large area, and flexible computing systems [3]. Analogous to color printing, PE methods can be based on screen printing, jet-printing or roll-to-roll processes [3]. These additive methods, where functional materials are directly deposited on the substrate, greatly simplify the production chain compared to subtractive silicon-based processes. This leads to savings in per unit-area costs and enables flexible hardware (since deposition can be performed on a flexible substrate).

Among the different printing methods, inkjet printing based on electrolyte-gated transistors (EGT) [9] - which is the targeted technology in this work - has the advantage of on-demand and on-site fabrication due to its mask-less fabrication process,

978-3-9819263-5-4/DATE21/©2021 EDAA

TABLE I: Typical Component sizes in EGT-based PE and silicon (FDSOI-32nm)

	Digital - μm^2			Analog - Passives - μm^2		
	n-type transistor	Inverter	SRAM cell	Resistor	Diode	Capacitor
PE	$7x10^4$	10^5	$3.4x10^5$	$3x10^4$	$7x10^4$	$1.5x10^4$
SI	0.05	0.1	0.3	1.8	1	7.3

also referred to digital printing, where jetting is controlled by a piezo-electric mechanism. Thus, inkjet-printing enables highly customized designs, generated by computer-aided design (CAD) software.

Despite the promising features, there are several limitations of PE compared to traditional silicon technologies. First, the functional density of printed circuits is very low (Table I) compared to silicon technology, which restricts the complexity of printed hardware. Secondly, EGTs have large intrinsic transistor gate capacitances which limits the performance of EGT-based circuits compared to nanometer technologies. Thirdly, due to the non-determinism in droplet printing, process variations in PE are much higher and can induce performance fluctuations or even hardware failures. As a result, complex Boolean digital logic designs are infeasible in PE. This encourages the use of SC in PE (so that the same functionality can be implemented with a much smaller number of transistors) as well as analog design paradigms.

B. Stochastic Computing

The basic idea of stochastic computing is to represent continuous values as a sequence of random bits. The sequentially encoded value of a stochastic bitstream can be obtained by counting the number of '1's and dividing them by the length of the bitstream. E.g. the sequence $(0,1,1,0,0)$ represents $Y = 2/5$. In order to obtain negative numbers, bi-polar encoding can be performed [10] by using the transformation $\widetilde{Y} = 2 * Y - 1$, where $\widetilde{Y}$ is the bi-polar representation with the value range $[-1,1]$.

Based on the bi-polar encoding of binary numbers, addition and multiplication can be performed using simple logic gates. The addition of two stochastic numbers is realized by a 2-input multiplexer whose select signal is driven by a bit stream with equal probability of containing '1's and '0's. As both input bit streams are stochastic and assumed to be uncorrelated with the multiplexer select signal, the addition is a random experiment [6] with expected value:

$$\mathbb{E}\big[\widetilde{Y}\big] = \frac{1}{2}\big(\mathbb{E}\big[\widetilde{A}\big] + \mathbb{E}\big[\widetilde{B}\big]\big) \tag{1}$$

, where $\widetilde{A}$, $\widetilde{B}$ are the bi-polar encoded inputs and $\widetilde{Y}$ is the bi-polar encoded output of the multiplexer. Note that the pair-wise addition operation is scaled by a factor of $1/2$.

The multiplication of two stochastic bi-polar bit streams is performed by an XNOR gate [10]. By applying the expected function to this logic expression we obtain [10]:

$$\mathbb{E}\big[\widetilde{Y}\big] = \mathbb{E}\big[\widetilde{A}\big] \cdot \mathbb{E}\big[\widetilde{B}\big] \tag{2}$$

C. Stochastic Computing for Neural Network Implementation

A stochastic computing NN (SC-NN) is similar to a conventional NN, using the same training and inference techniques, with a few key differences. First, some operations such as input adders, are scaled in SC by a factor of $\frac{1}{2}$ (or less depending on the number of inputs). Second, the latency of SC-NNs is not constant but dependent on the bit-stream length of the deployed stochastic number. E.g. when the inputs of the SC-NN are encoded by stochastic numbers with 64 bits, they are $4\times$ faster than using 1024 bits (the latter provides higher SC-NN inference accuracy). Also, as the NN classification outcome is related to the stochastic bit-stream at the NN output, an early classification result can be obtained using the first arriving bits.

The benefits of using SC-NNs compared to conventional NNs is a reduction in complexity. As illustrated in Fig. 1, circuit designs of multipliers and adders are highly simplified. A digital adder requires many logic gates and full adder blocks, leading to dozens of transistors, while an SC implementation requires only one multiplexer with 7 transistors. Similarly the multiplier complexity is reduced.

III. Proposed SC design for Printed Electronics

A. Motivation: Limitations of printed digital NN and SC-NNs

Fig. 1 depicts the implementation of an artificial neuron using digital logic as well as SC. For the SC-based neuron, the components for the stochastic number generators (SNGs) realized by True Random Number Generators (TRNGs) and the activation function (AF) are also presented. The SNGs are used to convert binary numbers into stochastic bit streams for the implementation of features, weights and select signals to drive the multiplexers for the add operation.

As digital MAC operations require complex fixed point multipliers as well as multi-bit adders, using a fully digital multi-bit architecture for printed NN would require a large number of logic gates. For instance, a digital 8-bit and 4-input MAC operation contains 1310 transistors (Table II). Correspondingly, a digital 8-bit neuron with 3 inputs implemented in EGT technology would require $3174\mathrm{mm}^2$ area, with a delay of $243\mathrm{ms}$ and power consumption of $123\mathrm{mW}$, as obtained based on synthesis results using EGT standard cell library [4].

Obviously, such a neural network is infeasible to be printed and cannot be driven by any imaginable lightweight battery or an energy harvester system. In contrast, an SC-based MAC operation requires only 25 transistors ($52\times$ fewer), as it uses an XNOR gate for the multiplication, and a 2-input multiplexer (MUX) for the addition. Thus, by replacing MAC operations by SC-based counterparts, substantial area and power consumption savings are achieved and one may imagine that printed implementations may become feasible.

However, while multipliers and adders in SC have low transistor count, the use of complex SNGs and activation functions is still a hurdle. Consider SNGs, which are deployed in large numbers: for each NN input, weight, and multiplexer select signal. The digital SNGs are conventionally implemented by multi-bit registers, digital pseudo-random number generators and comparators [11], which altogether dominate the total chip area (Table II) and diminish the area and power consumption gains obtained by the low-complexity SC-based multiplier and adder operations. To reach a less complicated and feasible SC

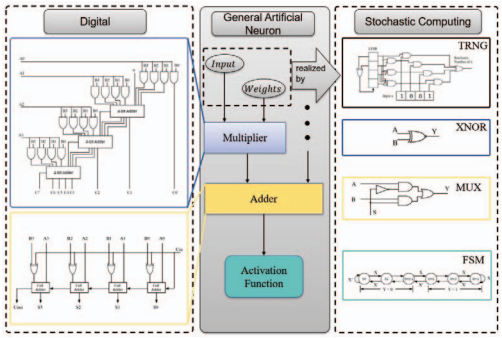

Fig. 1: Implementation of artificial neuron components using digital computing vs stochastic computing

design which to be printed, novel and inexpensive SNGs and AF designs have to be explored.

B. Analog Components for printed SC-NN

In order to reduce the transistor count of the expensive digital SC-based SNGs and AFs, we considered efficient analog implementations. The proposed printed analog stochastic number generator for NN weights (wSNG) is depicted in Fig. 2. Its functionality can be explained as follows: the wSNG consists of a ring oscillator (RINGO) for generating an oscillating signal [9]. This signal is then applied to an enable transistor of a tuned true random number generator (TTRNG). The TTRNG is implemented as a bi-stable back-to-back inverter (T_1, T_2, R_1, R_2). The meta-stability of the printed back-to-back inverter is caused by random noise (e.g. thermal noise, shot noise etc. [12]). Each time the TTRNG is enabled (T_3 turned on) the TTRNG output voltage 'OUT' is either logic '1' or '0', dependent on the random noise and ratio of the pull-up resistors R_1/R_2, which correspond to the pre-trained NN weight. As a result, when driving the TTRNG by an oscillating enable signal produced by the RINGO, a bit stream of random bits is generated at the 'OUT' port. To obtain the desired R_1/R_2 ratio, which is initially biased by process variations, the TTRNG is tuned by printing additional layers to the pullup resistor in a post fabrication step [12]. Each additional layer decreases the resistance as the layers behave as multiple resistors connected in parallel [12]. Thus, the probability of producing '1's and '0's can be adjusted to implement any stochastic number for the neural network weights. This is a unique capability enabled by PE, which is not feasible with subtractive processes such as in silicon-technology.

By adding a printed transistor (T_4) to the pull-up network, even analog neural network inputs/features (X) can be converted into stochastic numbers (iSNG). As a result, the proposed circuit can be used to realize SNGs for all NN signals such as inputs (iSNG), weights (wSNG) and multiplexer select signals (wSNG). While the wSNG is one-time programmable, the iSNG is input-controlled by the input voltage of the additional transistor T_4, which has to be within the range of $[-2V,2V]$ (maximum operating voltage of EGTs). Another advantage of the analog wSNGs is that the weights can be stored in the form of printed resistors and no explicit weight

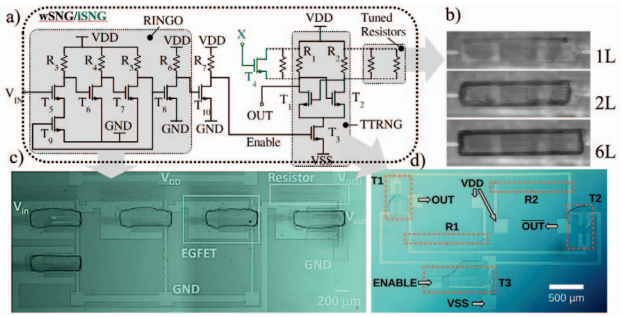

Fig. 2: Schematic and microscopic photos of SNG: a) Schematic of the proposed SNG consisting of a RINGO and a TTRNG for wSNG, and additional transistor T_4 (green) for iSNG. b) microscopic photo of resistor tuning in a post-fabrication step, by printing first 1 layer (1L), 2 layers (2L) and 6 layers (6L) of conductive materials c) microscopic photo of printed RINGO [9] d) microscopic photo of printed TRNG [12]

storage is required, further reducing the hardware footprint and rendering the implementation of SRAM cells unnecessary [10].

We also designed an analog circuit for the SC-based activation function. The circuit contains a capacitor for analog integration of voltage pulses from stochastic bit streams. In EGT technology, the 2-layer capacitor can be build from a printed electrolyte-semiconductor interface using the same functional inks as for the EGT fabrication. The area of this interface is adjusted according to the desired capacitance.

The circuit schematic is depicted in Fig. 3. At the input port of the activation function, a stochastic number in the form of random bit streams is applied. The activation function operation state depends on the enable signal (EN). When the enable signal is logic '1', the input signals (IN) are applied to a charging stage via transistor T_2 and if the incoming bits are logic '1', the capacitor C_1 is connected to VDD and charged. If the incoming bit is logic '0', T_3 is activated by the inverter (T_1, R_1) and the capacitor is connected to VSS by T_3 and thus discharged.

After the incoming bitstream is processed, the capacitor is loaded to a certain voltage level, proportional to the number of '1's in the input bitstream. Next, the second operation state begins by setting the enable signal to logic '0', and, consequently, the capacitor is disconnected from the charging stage and keeps its voltage. Moreover, the output stage is activated by turning transistor T_5 on. If the voltage level at the capacitor exceeds a certain threshold, the output stage is activated by turning transistor T_6 on and the input signal can propagate through $T_1 - T_7$ to the output (OUT). If the capacitor voltage is below the threshold, the output (OUT) is pulled down permanently to logic '0'. This functionality is akin to a rectified linear unit (ReLU) activation function. In the negative input range, however, the activation function output is constantly -1 due to the bi-polar encoding (in bi-polar SC-based encoding, a sequence of only '0's is represented as -1). Thus, we denote this component as bi-polar rectified linear unit (bi-polar ReLU). Additionally, the discharging transistor T_8 can be activated to reset the capacitor to its initial state at

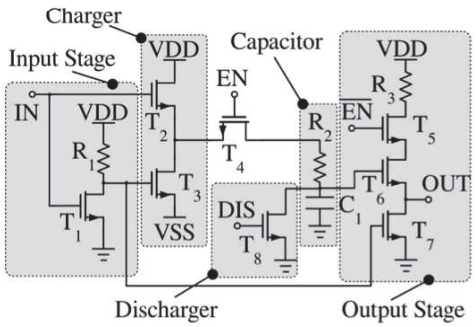

Fig. 3: Schematic of the analog activation function which resembles a bi-polar ReLU.

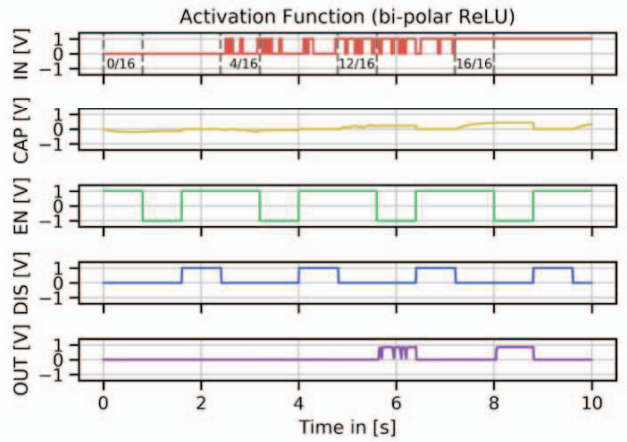

Fig. 4: Simulation of the printed analog bi-polar ReLU. The following signals are depicted: IN: input bit stream, CAP: capacitor voltage, EN: enable signal, DIS: discharging signal, OUT: output bit stream. As can be obtained, only when the input bit-stream has more than 8 '1's out of 16 ($> 8/16$), the input signal can propagate through the output.

0V and the next activation function evaluation can be initiated. The functionality of the bi-polar ReLU was validated using simulations as depicted in Fig. 4.

C. Overall Architecture

The overall architecture of the artificial neuron for SC-NNs is presented in Fig. 5. Before fabrication, both the NN topology and pre-trained weights are determined by the end-user during NN training deployed in software. In each SC-neuron, the inputs and weights are then converted into stochastic numbers by the iSNGs/wSNGs. After multiplication using XNOR gates, the signals are added by the multiplexer. These two operations implement the MAC operation. It is important to note that the addition result is multiplied by $1/2$, as discussed in Section II. After the addition, the bi-polar ReLU (AF) is applied to the adder result and passes the output to either the input of a neuron in the next layer or provides information about the classification result as part of the output layer. Thus, in all subsequent layers, the iSNGs from the input layer are replaced by the node output 'Y'. The whole SC-NN is then printed by the end-user using an inkjet printer for point-of-use fabrication (see Section II-A).

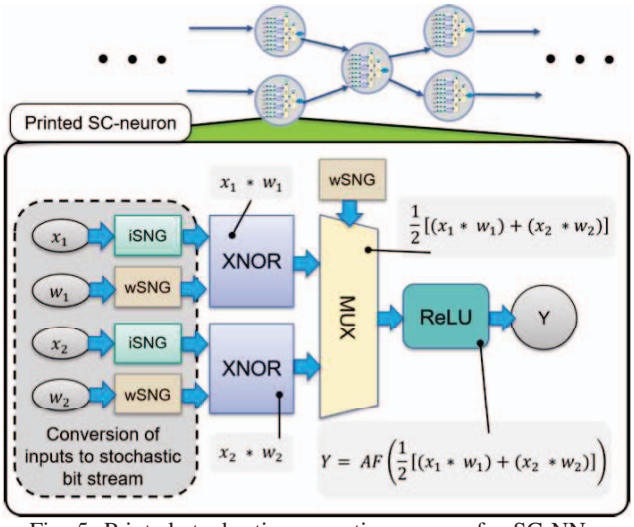

Fig. 5: Printed stochastic computing neuron for SC-NNs

D. Training of proposed SC-NN

We consider technology-dependent constraints which have to be taken into account during training of the proposed SC-NN.

The first constraint is due to the bi-polar encoding of the stochastic numbers which represent the inputs and weights (see Section II). The range of values is:

$$\vec{x}_i, \vec{w}_i \in [-1, 1] \tag{3}$$

The second constraint is due to the stochastic number adder, which realizes a weighted addition operation by the factor $\frac{1}{2}$ (see Section II). As additions are only performed pair-wise, the scaling factor changes according to the total number of summands. For N summands, the addition operation becomes:

$$y = \frac{1}{2^{\lceil \log_2 N \rceil}} \sum_{i=1}^{N} x_i, \tag{4}$$

where x_i are the adder inputs and y the adder output. To respect these constraints in the training procedure, the network is directly trained using (4) instead of the classical weighted sum operation, while the proposed bi-polar ReLU is used as an activation function in training. Additionally, the range of feasible values $w \in [-1, 1]$ is guaranteed through clipping the weights after each update step $(t + 1)$, i.e.

$$\vec{w}^{(t+1)} = \text{clip}\left(\vec{w}^{(t)} + \alpha^{(t)} \Delta \vec{w}^{(t)}\right),$$

where $\alpha^{(t)}$ denotes the learning rate and

$$\text{clip}(z) = \begin{cases} -1 & z < -1 \\ z & z \in [-1, 1] \\ 1 & z > 1 \end{cases}$$

refers to an element-wise operation projecting the entries on the feasible range of $w_i \in [-1, 1]$. Finally, the weight update $\Delta \vec{w}^{(t)}$ can be obtained using any optimizer of choice. As an example, classical gradient descent would use the negative gradient of the loss with respect to the weights as $\Delta \vec{w}^{(t)}$.

E. Discussion

It is important to note that analog designs are usually more susceptible to noise and variation than digital counterparts. However, previous work has validated experimentally that the printed SNG can be unbiased and compensated with respect to process variations [12].

For the analog activation function, we performed a sensitivity analysis of the process variation parameters which impact the circuit functionality. In the sensitivity analysis experiment, all variational parameters such as transistor threshold values and resistor resistances were varied by 20% of the nominal value. Moreover, the worst case was considered, where the input bitstream has 50% '1's and '0's, as this stochastic number is represented by 0, which is the threshold value of the activation function. As the activation function threshold is determined by the capacitor voltage, the impact of component variations was observed with respect voltage fluctuations at the capacitor. We observed that the capacitor value was biased by only $1\mathrm{mV}$, on average, under variation. This corresponds to 2% of the maximum capacitor voltage during variation-free operation. Thus, the activation function operates similar to the variation-free case.

IV. Results

The presented hardware results, comparing coventional and SC-based NNs in PE, were extracted from high-level synthesis tools (Synopsis design compiler) using a physical design standard cell library based on a printed process design kit [4]. By substitution of digital SC components by analog designs, area usage and power consumption were substantially improved. As depicted in Table II, the analog SNG implementation requires only 2.5% of the area and 0.8% of power consumption compared to the digital 8-bit SNG implementation. Similarly, for the activation function (AF), the area usage and power consumption is 17% and 7% of the digital components, however, at the expense of increased circuit delay.

The proposed mixed-signal approach is compared against a 4-bit and 8-bit digital implementation as well as a digital stochastic implementation. The evaluations are carried out for a single 3-input neuron (Table III) and a full NN (Fig. 6) with the topology 9-3-2 (9 inputs, 3 hidden nodes, 2 output nodes).

As we can see from the neuron results (Table III), the proposed mixed-signal neuron contains only 0.6% of transistors compared to an 8-bit digital SC-NN. Consequently, the area requirement and power consumption is 0.6% and 0.6% respectively. For reference, a conventional digital 8-bit neuron requires 138× more transistors to implement the same neuron. The digital 4-bit NN, while much smaller than the 8-bit equivalent, contains 6.9× more EGTs than the proposed design.

Similar trend can be observed for the full NN design, depicted in Figure 6. Figure 6 also shows that the improvements in transistor count for the proposed mixed-signal SC-NN are at the expense of higher inference time compared to the digital SC-NN. It is important to mention that the delay of the SC-NNs cannot be directly compared to the NNs, as also the length of the stochastic bitstream has to be considered. Choosing long bitstreams improves the accuracy, but also increases delay, and vice versa. For the following discussion on the inference results, we fixed the bit stream length to a constant value (1024).

TABLE II: Comparison between 4 and 8-bit digital components, 4 and 8-bit digital SC components and analog SC implementations. MAC operation has 4 inputs.

	Components	Delay	Area	Power	#EGTs
Digital 4-bit	ADDER	13ms	$7.9\mathrm{mm}^2$	289µW	59
	MULT	13.6ms	$15\mathrm{mm}^2$	550µW	103
	MAC	26.6ms	$37.9\mathrm{mm}^2$	1389µW	265
	AF	2.5ms	$1.7\mathrm{mm}^2$	80µW	10
	ADC	13.8ms	$25.4\mathrm{mm}^2$	328µW	185
Digital 8-bit	ADDER	29ms	$22\mathrm{mm}^2$	793µW	144
	MULT	28ms	$85\mathrm{mm}^2$	3100µW	583
	MAC	57ms	$192\mathrm{mm}^2$	6993µW	1310
	AF	2.55ms	$3.7\mathrm{mm}^2$	120µW	22
	ADC	154ms	$957\mathrm{mm}^2$	37 200µW	5938
Digital SC 4-/8-bit	ADDER	2.4ms	$0.97\mathrm{mm}^2$	33µW	7
	MULT	3.9ms	$1.4\mathrm{mm}^2$	51µW	9
	MAC	6.3ms	$3.77\mathrm{mm}^2$	135µW	25
	4-bit SNG	15ms	$34\mathrm{mm}^2$	5990µW	228
	8-bit SNG	23ms	$71.7\mathrm{mm}^2$	11 030µW	436
	4-bit AF	9.9ms	$16.7\mathrm{mm}^2$	1920µW	115
	8-bit AF	5.3ms	$15.15\mathrm{mm}^2$	1670µW	103
Analog	**SNG**	**50ms**	$\mathbf{1.76mm^2}$	**94.53µW**	**11**
	AF	**50ms**	$\mathbf{2.62mm^2}$	**116.7µW**	**8**

TABLE III: Comparison between digital 3-input artificial neuron for 4 and 8-bit conventional NN, digital SC-based neuron for 4 and 8-bit digital SC-NN, and and mixed-signal neuron (proposed) all with stochastic bitstream length of 1024.

	Method	Delay	Area	Power	#EGTs
Neuron	4-bit NN	55.9ms	$138.7\mathrm{mm}^2$	3.292mW	992
	8-bit NN	242.55ms	$3174\mathrm{mm}^2$	123mW	19873
	4-bit SC-NN	47ms	$372\mathrm{mm}^2$	51mW	2542
	8-bit SC-NN	191ms	$3466\mathrm{mm}^2$	202mW	21453
	Proposed	**108ms**	$\mathbf{23mm^2}$	**1.12mW**	**144**

The inference accuracy results for the different design points are shown in Table IV. In total, 13 benchmark datasets chosen from the UCI ML Repository [13] were used for evaluation. Design points that were evaluated include a hardware-agnostic NN (unconstrained weights, true ReLU), a PE specific NN (weights bounded, stochastic adder (Equ. (1)), bi-polar ReLU), deterministic 4-bit and 8-bit NNs (fully digital), and the SC-NNs (random bit stream length of 1024) - both 4-bit and 8-bit digital implementations and mixed-signal SC (proposed) implementation.

For all datasets, the input features were normalized to a range of $[-1, 1]$. Then, all NNs were trained using the Adam optimizer for 200 epochs using full-batch updates and the mean squared error loss function. The training was repeated several times with different seeds, label smoothing factors, initial learning rates and learning rate schedules (halving the learning rate after a given number of epochs). The respective NN with the best training accuracy was reported in Table IV. The topology of the NNs was kept fixed at $\#input \times 3 \times \#output$. Training/testing was performed using a random 67%/33% split.

As can be seen from Table IV, all trained SC-NNs exceeded the accuracy of the random guess (baseline) substantially for 10 of the 13 datasets. Moreover, we can observe that the SC-NNs have only small variations on the inference result, smaller than 2.9%. Also, the proposed NN achieves similar

TABLE IV: Comparison of NN inference results for a hardware-agnostic NN (unconstrained weights, true ReLU), a PE specific NN (weights bounded, stochastic adder, bi-polar ReLU), deterministic 4-bit and 8-bit NNs (fully digital), and the SC-NNs (random bit stream length of 1024) - both 4-bit and 8-bit digital implementations and mixed-signal SC (proposed) implementation. Both average and 1-sigma confidence interval (±) are included. Also, random guess is provided as a baseline. For the hardware-agnostic and PE-specific NNs, also inference results on the train/test-split are shown.

		Hardware -Agnostic		PE Specific		Deterministic		Stochastic			Baseline
Dataset	Topology	Train	Test	Train	Test	4-bit	8-bit	4-bit	8-bit	**Proposed**	Random Guess
Acute Inflammation	6-3-2	1.0	1.0	1.0	1.0	1.0	1.0	0.920 ±0.029	0.915 ±0.028	0.918 ±0.03	0.475
Balance Scale	4-3-3	0.914	0.903	0.900	0.874	0.852	0.914	0.874 ±0.008	0.872 ±0.008	0.874 ±0.008	0.440
Breast Cancer Wisconsin	9-3-2	0.970	0.965	0.955	0.952	0.967	0.970	0.714 ±0.009	0.714 ±0.008	0.714 ±0.008	0.667
Energy efficiency (y1)	8-3-3	0.889	0.882	0.870	0.850	0.881	0.889	0.793 ±0.013	0.794 ±0.013	0.791 ±0.014	0.433
Energy efficiency (y2)	8-3-3	0.909	0.890	0.883	0.862	0.885	0.905	0.748 ±0.013	0.750 ±0.012	0.749 ±0.015	0.465
Iris	4-3-3	0.990	1.0	0.910	0.860	0.96	0.99	0.684 ±0.01	0.6837 ±0.011	0.684 ±0.009	0.280
Mammographic Mass	5-3-2	0.838	0.833	0.824	0.780	0.812	0.835	0.756 ±0.009	0.755 ±0.009	0.755 ±0.009	0.550
Seeds	7-3-3	0.979	0.986	0.921	0.914	0.942	0.971	0.867 ±0.018	0.869 ±0.017	0.868 ±0.018	0.271
Tic-Tac-Toe Endgame	9-3-2	0.991	0.975	0.989	0.978	0.808	0.990	0.668 ±0.004	0.668 ±0.004	0.668 ±0.004	0.640
Vertebral Column (2 cl.)	6-3-2	0.855	0.903	0.768	0.786	0.812	0.850	0.559 ±0.022	0.559 ±0.022	0.555 ±0.024	0.690
Vertebral Column (3 cl.)	6-3-3	0.768	0.806	0.647	0.670	0.469	0.768	0.297 ±0.026	0.290 ±0.023	0.297 ±0.022	0.515
Cardio	21-3-3	0.839	0.826	0.784	0.766	0.802	0.826	0.784 ±0.0006	0.784 ±0.0003	0.784 ±0.0005	0.766
Pendigits	16-3-10	0.518	0.520	0.501	0.509	0.441	0.526	0.216 ±0.004	0.218 ±0.002	0.219 ±0.004	0.099

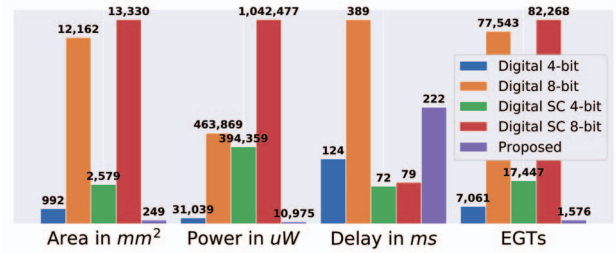

Fig. 6: Comparison between digital NN, digital SC-NN and mixed signal SC-NN (proposed) for topology: 9-3-2 (all NNs based on EGTs). Average accuracy across all datasets is also provided.

accuracy compared to a 4-bit digital NN (deterministic) for a few datasets. Overall, we can validate that the proposed stochastic SC-NN is capable of performing classification tasks, and leads to less than 13% inference accuracy loss compared to an unconstrained NN (hardware-agnostic) for a majority of the benchmark datasets.

V. Conclusion

Printed electronics (PE) offers flexible, extremely low-cost and on-demand hardware due to its additive manufacturing process, enabling emerging ultra-low-cost applications, including machine learning applications, not realizable in conventional silicon technologies. However, large feature sizes in PE limit the complexity of a machine learning classifier (e.g., a neural network (NN)) in PE. Stochastic computing Neural Networks (SC-NNs) can reduce area in silicon technologies, but increase overhead in PE due to unique implementation tradeoffs in PE. In this work, we presented printed mixed-signal stochastic computing-based neural network hardware, which substitutes complex and power-hungry conventional stochastic computing (SC) components by printed analog designs. The area/power consumption of the proposed neural network is only 25%/35% of a 4-bit digital implementation and 10%/3% of a conventional 4-bit SC-based NN. In addition, the architecture is designed to allow arbitrary length of stochastic bit stream which, in turn, allows a reduction in the variance of classification estimates at the expense of longer neural network inference time. The proposed approach is particularly interesting for printed near-sensor processing hardware in applications where silicon-based technology is not an option.

Kumar would like to thank NSF for partial support of this work.

References

[1] B. Shao, "Fully printed chipless rfid tags towards item-level tracking applications," Ph.D. dissertation, KTH Royal Institute of Technology, 2014.

[2] W. Maly, "Cost of silicon viewed from vlsi design perspective," in *Proceedings of the 31st annual Design Automation Conference*, 1994, pp. 135–142.

[3] Z. Cui, *Printed electronics: materials, technologies and applications*. John Wiley & Sons, 2016.

[4] N. Bleier, M. Mubarik, F. Rasheed, J. Aghassi-Hagmann, M. Tahoori, and R. Kumar, "Printed microprocessors," in *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 2020.

[5] M. Mubarik, D. Weller, N. Bleier, M. Tomei, J. Aghassi-Hagmann, M. Tahoori, and R. Kumar, "Printed machine learning classifiers," in *Proceedings of the 53rd Annual IEEE/ACM International Symposium on Microarchitecture*, 2020.

[6] Y. Liu, S. Liu, Y. Wang, F. Lombardi, and J. Han, "A survey of stochastic computing neural networks for machine learning applications," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[7] J. Kim, I. Jeerapan, S. Imani, T. N. Cho, A. Bandodkar, S. Cinti, P. P. Mercier, and J. Wang, "Noninvasive alcohol monitoring using a wearable tattoo-based iontophoretic-biosensing system," *Acs Sensors*, vol. 1, no. 8, pp. 1011–1019, 2016.

[8] P. Mostafalu, W. Lenk, M. R. Dokmeci, B. Ziaie, A. Khademhosseini, and S. R. Sonkusale, "Wireless flexible smart bandage for continuous monitoring of wound oxygenation," *IEEE Transactions on biomedical circuits and systems*, vol. 9, no. 5, pp. 670–677, 2015.

[9] G. C. Marques, F. von Seggern, S. Dehm, B. Breitung, H. Hahn, S. Dasgupta, M. B. Tahoori, and J. Aghassi-Hagmann, "Influence of humidity on the performance of composite polymer electrolyte-gated field-effect transistors and circuits," *IEEE Transactions on Electron Devices*, vol. 66, no. 5, pp. 2202–2207, 2019.

[10] A. Ren, Z. Li, C. Ding, Q. Qiu, Y. Wang, J. Li, X. Qian, and B. Yuan, "Sc-dcnn: Highly-scalable deep convolutional neural network using stochastic computing," *ACM SIGPLAN Notices*, vol. 52, no. 4, pp. 405–418, 2017.

[11] S. C. Shenoi, "A comparative study on methods for stochastic number generation," Ph.D. dissertation, University of Cincinnati, 2017.

[12] A. T. Erozan, G. Y. Wang, R. Bishnoi, J. Aghassi-Hagmann, and M. B. Tahoori, "A compact low-voltage true random number generator based on inkjet printing technology," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 6, pp. 1485–1495, 2020.

[13] D. Dua et al., "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml