Advances in Computer Science and Information Technology (ACSIT)
p-ISSN: 2393-9907; e-ISSN: 2393-9915; Volume 10, Issue 2; April-June 2023, pp. 78-84
© Krishi Sanskriti Publications
http://www.krishisanskriti.org/Publication.html

Comprehensive Study of Stochastic Computing Methods for Hardware Acceleration of Deep Learning Systems

A Dinesh Babu and Dr C Gomathy

[1]Ph.D Scholar, Department of ECE, SRM Institute of Science and Technology, Vadapalani Campus
[2]Faculty, Department of ECE, SRM Institute of Science and Technology, Vadapalani Campus
E-mail: [1]dineshba@srmist.edu.in, [2]vp.academics.vdp@srmist.edu.in

Abstract —*In recent years, deep learning has become increasingly popular for solving complex problems in various fields, including computer vision, natural language processing, and robotics. However, deep learning models require large amounts of computing resources, and the traditional von Neumann computing architecture is not optimized for deep learning workloads. One way to improve the efficiency of deep learning systems is using hardware acceleration. Stochastic computing (SC) is a promising method for hardware acceleration of deep learning systems. SC represents numbers as sequences of stochastic bits, where the probability of a bit being 1 or 0 is determined by a probability distribution. This method can be used to perform various arithmetic operations, including addition, multiplication, and convolution. This comprehensive study focuses on the use of SC for hardware acceleration of deep learning systems. The study first provides an overview of the von Neumann architecture and its limitations for deep learning workloads. It then introduces the concept of SC and its advantages for hardware acceleration of deep learning systems. The study presents several approaches for using SC in deep learning, including SC-based neural network models, SC-based convolutional neural networks (CNNs), and SC-based accelerators for CNNs. The study also discusses the challenges and limitations of using SC in deep learning systems, such as the need for high-quality probability distributions and the difficulty of implementing complex operations using SC. To evaluate the performance of SC-based deep learning systems, the study presents several experiments comparing SC-based approaches with traditional approaches. The results show that SC-based approaches can provide significant improvements in energy efficiency and speed compared to traditional approaches. In conclusion, this comprehensive study demonstrates the potential of stochastic computing for hardware acceleration of deep learning systems. While there are still challenges and limitations to be addressed, the results suggest that SC-based approaches have the potential to significantly improve the efficiency of deep learning systems.*

INTRODUCTION

Hardware acceleration of deep learning systems refers to the use of specialized hardware components to speed up the training and inference of deep neural networks. Deep learning has become a fundamental technology behind modern artificial intelligence, and hardware acceleration has been essential in making this technology practical and widely applicable.

The history of hardware acceleration for deep learning can be traced back to the early 2010s, when graphics processing units (GPUs) were first used to accelerate neural network training. Since then, a range of specialized hardware components have been developed, including tensor processing units (TPUs), field-programmable gate arrays (FPGAs), and application-specific integrated circuits (ASICs), which are designed specifically for deep learning tasks.

Previous methods of training deep neural networks without hardware acceleration were very slow and required massive amounts of computing power, making it impractical to train large-scale models on general-purpose processors[1][2]. With the introduction of hardware acceleration, training times have been dramatically reduced, and it is now possible to train deep neural networks with billions of parameters in reasonable amounts of time.

However, previous methods of hardware acceleration had some disadvantages. GPUs, which were the first hardware accelerators for deep learning, were designed for graphics processing and were not optimized for deep learning workloads, leading to some inefficiencies.[3] Additionally, FPGAs and ASICs were very expensive to develop, making them inaccessible to most researchers and companies. Finally, hardware accelerators also required specialized software libraries and frameworks, which could be challenging for developers who were not familiar with them. Despite these disadvantages, hardware acceleration has been a game-changer for deep learning, enabling significant advances in the field and opening up new possibilities for real-world applications.

Stochastic computing is a promising technique for hardware acceleration of deep learning systems, which can significantly reduce the power consumption and hardware complexity compared to traditional methods. Stochastic computing is a technic in which random bit streams are used to represent numerical values, which allows for efficient implementation of arithmetic operations such as multiplication and addition using simple logic circuits. This approach has been shown to be particularly effective for neural network inference, where the computation can be decomposed into a series of matrix-vector multiplications and nonlinear activation functions.[4]

Several recent studies have demonstrated the potential of stochastic computing for hardware acceleration of deep learning. For example, a study by Srinivasan et al. (2019) showed that a stochastic computing-based accelerator can achieve up to 40x energy efficiency improvement compared to a traditional digital accelerator for convolutional neural networks. Another study by[5] Wang et al. (2020) demonstrated that a stochastic computing-based neural network accelerator can achieve state-of-the-art performance on several benchmark datasets, with significantly lower power consumption and hardware complexity compared to traditional digital accelerators. While stochastic computing has shown great promise for hardware acceleration of deep learning, there are still some challenges to overcome, such as the need for specialized hardware and software tools to design and optimize stochastic computing-based accelerators. However, the potential benefits in terms of energy efficiency and hardware complexity make stochastic computing an area of active research in the field of deep learning hardware acceleration. The following chapter discusses in detail about the various literatures, which address the efficient hardware acceleration through inculcation of the stochastic computing methods in implementations highlighting its advantages and shortcomings. The chapter 3 details about the different methods of stochastic computing for implementing deep learning systems. Chapter 4 discusses the comparison chart of different methods with salient points.

RELATED WORKS

Srinivasan et al. (2019) proposed a stochastic computing-based accelerator for convolutional neural network (CNN) inference. The authors show that their approach can achieve up to 40x energy efficiency improvement compared to a traditional digital accelerator for CNNs.[4]

Zhou et al., (2020) presented a stochastic computing-based accelerator for neural network inference. The authors demonstrate that their approach can achieve significant energy efficiency improvements compared to traditional digital accelerators [6]. A stochastic computing-based neural network accelerator for edge computing is proposed in [5]. The authors show that their approach can achieve state-of-the-art performance on several benchmark datasets, with significantly lower power consumption and hardware complexity compared to traditional digital accelerators. An efficient stochastic computing approach for convolutional neural networks (CNNs) was proposed in [7]. The results shows that their approach can achieve comparable accuracy to traditional digital approaches while reducing hardware complexity and power consumption. Another work on stochastic computing-based accelerator for CNNs on field-programmable gate arrays (FPGAs) shows that the approach can achieve similar accuracy to traditional digital approaches with significantly reduced hardware complexity and power consumption [8]. A stochastic computing-based framework for real-time object detection is demonstrated and the approach achieved high accuracy with low hardware complexity and power consumption [9]. Nourani, M., & Mostafa, presented an analysis of stochastic computing for deep neural networks, considering accuracy, power consumption, and noise. The authors show that their approach can achieve high accuracy with reduced power consumption compared to traditional digital approaches [10]. [11] Proposes a stochastic computing-based deep neural network for accelerated sensing. The authors show that their approach can achieve similar accuracy to traditional digital approaches with significantly reduced hardware complexity and power consumption. Work in [12] presents a stochastic computing-based neural network for object recognition in real-time applications. The authors demonstrate that their approach can achieve high accuracy with low power consumption. Zhang et al., (2021) [13] proposed a stochastic computing-based approach for deep learning inference on resistive random-access memory (ReRAM) crossbar arrays. The authors show that their approach can achieve comparable accuracy to traditional digital approaches with significantly reduced power consumption and hardware complexity. They also demonstrate that their approach can handle large-scale neural networks and achieve high throughput.

STOCHATIC COMPUTING NEURON - AN OVERVIEW

A stochastic computing (SC) circuit has a lower hardware cost than traditional binary circuits, uses less energy, and is more fault tolerant to soft and computational faults [14]. Many basic arithmetic circuits, including adders, subtractors, and multipliers, are made smaller by SC [15], [16], [17], and [18]. Linear finite state machines (FSMs) may be used to create several functions, including the sigmoid function, the hyperbolic tangent (tanh), and the exponential function [19]. By somewhat compromising computation accuracy, these architectures allow SC NNs to be implemented at a much reduced hardware cost. Stochastic sequences are also used by SC to encode actual values. As a result, it adds noise to the SC NNs and stochasticity. The overfitting problem could be resolved by using noise thus by improving accuracy.

However, because of the long sequence length and high number of stochastic number generators (SNGs) required in the circuit, it is difficult for a SC NN to achieve reduced computational latency and energy consumption when compared to traditional designs. To address this issue, numerous better SC encoding methods [20-22] have been developed to minimise sequence length, hence enhancing performance and energy efficiency. Some solutions concentrate on the enhancement and reuse of random number generators (RNGs), resulting in improved hardware and energy efficiency [23], [24]. When compared to traditional binary implementations, these novel methodologies make SC NNs competitive in terms of both hardware efficiency and compute performance.

In SC, the value p/q is encoded in the closed interval gate and an SNG if and only if a random binary bit stream of length q contains p 1's [25]. or the value (2 p q)/q in the closed interval [1, +1] in the bipolar form. Arithmetic circuit complexity is greatly reduced by SC due to its handling of the bit streams. It has found extensive usage in several fields, including low-density parity check (LDPC) decoding [26], image processing [27,28], digital filter design [29,30] and circuit reliability evaluation [31,32].

Most SC neurons, being the fundamental unit of NNs, have a similar structure depicted in figure 1. It is made up of an SNG array, a SC arithmetic circuit, and a probability estimator (PE). To convert a binary input into a stochastic sequence, an SNG composed of a RNG and a comparator is employed. The neuron's function is implemented by the SC arithmetic circuit. According to (1) and (2), multipliers, adders, and activation circuits can be used to build the SC neuron. These arithmetic circuits are necessary for inference in many types of NNs [33] and may be achieved by various SC designs.

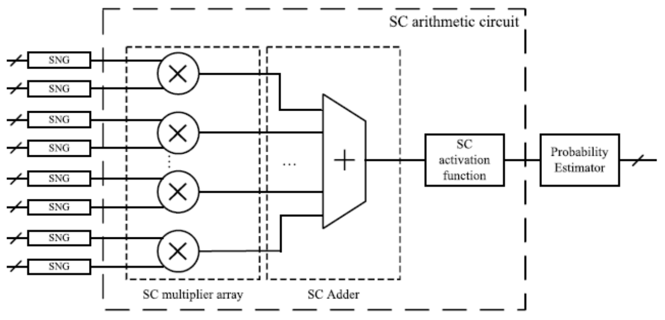

Figure 1: A Stochastic Neuron

$$(Y) = Pi \sum_{k=1}^{m} wk.xk \qquad (1)$$

$$\text{Pi}(v) = \begin{cases} \tanh(2v), & \tanh \\ 1/1+\exp(-v), & \text{Sigmoid} \\ \max(0, v), & \text{ReLU} \end{cases} \qquad (2)$$

To transform a stochastic sequence back into a binary value, PE is used. An adaptive digital element based circuit may be used to put it into practise. This architecture analyses the probabilities contained in the input sequence and the sequence generated by the SNG; when the input sequence's probability is greater, the value in the up-down counter is increased, and vice versa, until the same probability is reached. Assumed to be an estimate of the probability encoded in the input sequence after convergence, the value in the up-down counter stays constant.

One of the primary arithmetic circuits in a neuron is the multiplier. As illustrated in Figure 2, the SC multiplier is implemented by either an AND gate for the unipolar form or an XNOR gate for the bipolar model. When compared to traditional binary multipliers, the SC multiplier considerably lowers the area while speeding up calculation. With the probability of the select input set to 0.5, a two-input multiplexer (MUX) implements the conventional SC adder, as illustrated in Fig. 3. There is no association between the input signals that would alter the likelihood of the output signal. In order to cut down on hardware and energy, the RNGs might be shared across the input signals. However, more RNGs are still needed since the select signal needs to be uncorrelated with the input signals. Multiple input signals can be used to create a SC adder tree, and its output is scaled by 0.5, that is given by

$$P3 = (p1 + p2) / 2 \qquad (3)$$

where p1, p2, and p3 are the values encoded in the input and output sequences, respectively. Due to the computation's halving of the resolution, the output signals must be renormalized before moving on to the next stage of processing, adding to the adder tree's hardware requirements.

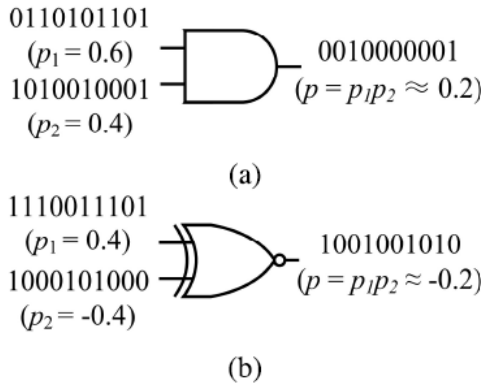

Figure.2. (a) Unipolar SC Multiplier

(b) Bipolar SC Mutiplier

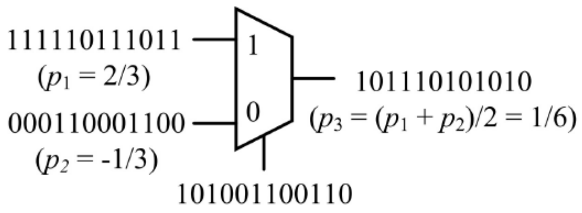

Figure 3: SC Adder

To enhance performance and get around the scaling issue, an accumulative parallel counter (APC)-based SC adder is presented. The 1s in the D-dimensional input sequences (Si, where I = 1, 2,..., D) are simply added in the APC-based SC adder. The RNGs can be shared among the input signals with no loss in computation accuracy since the probability of the output signal is only governed by the sum of the probabilities of the input signals. This design uses less hardware than the original SC adders for the same amount of input signals since no SNGs are needed to create the choose signals.

Because computation is carried out in parallel without the aid of an adder tree, the APC-based adder processes data faster than the original SC design. Additionally, the APC-based adder's output is in binary form. By using an approximate parallel counter in place of the APC, the design may be made smaller (AxPC). In order to use less hardware, pairs of OR and AND gates are used in the first layer in place of full adders (FAs) in the APC. A toggle flip-flop (TFF)-based SC adder is introduced. Assume that the values encoded in the input and output sequences in the unipolar representation are px , py, and pz . Hence the function is determined by the following equation

$$Pz = (px + py) / 2 \quad (4)$$

The output of the circuit is not impacted by the autocorrelation in the input signals since the sequence created by the TFF is uncorrelated with the sequences used as inputs. This adder has less circuit areas than the traditional SC adder since it doesn't need any additional stochastic sequences to send the choose signal to the MUX.

A FSM-based approach and a SC polynomial arithmetic circuit are commonly used to implement the activation functions of (2), respectively. Multiple activation functions are implemented using various state transition settings using FSM-based computational components. A saturating counter is a part of the architecture, and a closed loop regulates how it is operating. In Fig. 4, the exponentiation and tanh circuit state changes are depicted.

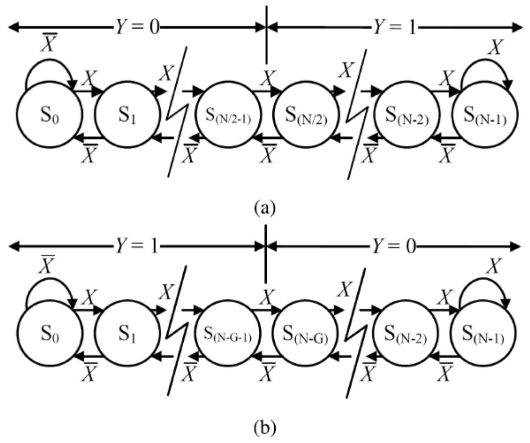

Figure 4: (a) Exponenitation (b) tanh

The technique delivers correct answers and speeds up processing even when the sum of the probabilities of the input signals exceeds [1,+1]. By substituting a linear approximation unit for the counters in the Btanh circuit, it is significantly enhanced. Configuring the values of p, r, and s in the LAU allows for the implementation of many types of activation functions. As an illustration, p = 0, r = 4, s = 1/2 is used to implement the sigmoid function, whereas p = 0, r = 1, s = 0 is used to create the ReLU function.

A SC based ReLU circuit is presented, the circuit's input is tallied and compared with one-half of the previous clock cycles. The CMP output simultaneously serves as the MUX's input and select signal. The CMP produces a "1" and is chosen as the output by the MUX if the accumulation result is less than the reference number. Otherwise, the Btanh circuit, which uses an up-down counter, controls the output. The circuit makes sure that the value encoded in the output sequence is at least 0.5 or 0 in the unipolar or bipolar representation. Therefore, assuming that the values encoded in the input sequence and the output sequence are x and y in the bipolar representation, the function of the circuit is then given by

$$Y = \max (0, \tanh(2x)) \quad (5)$$

Activation functions are frequently implemented using SC polynomial arithmetic circuits. First, Bernstein polynomials or Taylor series are used to extend the nonlinear activation functions. Then, SC polynomial circuits or exponentiation circuits compute a finite number of terms. An SC polynomial activation circuit has a bigger size but is simpler to change than the FSM technique. The activation function is roughly represented by linear functions in a recent implementation[34]. In SC adders, the AxPC is used to cut down on space and power requirements.

COMPARISON OF STOCHSTIC NETWORKS - ACCURACY AND HARDWARE EFFICIENCY

Stochastic computing is a method for representing numerical values as random bit sequences. In this method, a sequence of binary digits (0 or 1) is generated using a probability distribution, where the probability of each digit being 1 or 0 is determined by the distribution. Stochastic computing can be used to perform various arithmetic operations, including addition, multiplication, and convolution, using simple logic circuits. In this section, we discuss several methods of using stochastic computing for deep learning systems.

Stochastic Computing-Based Neural Network Models

Stochastic computing is a technique that can be used to encode the weights and activations of neural networks. It involves representing each weight and activation as a sequence of stochastic bits and carrying out the computation using stochastic logic circuits. Stochastic neural networks have been demonstrated to be effective for various applications such as speech recognition and image classification. One of the main benefits of this approach is its capacity to withstand errors resulting from computational noise. As stochastic computing is intrinsically noisy, stochastic neural networks can be trained to handle this noise, making them appropriate for low-power and high-reliability applications.

Stochastic Computing-Based CNNs

Convolutional neural networks (CNNs) are commonly used in image and video processing and employ convolution operations to extract features. Stochastic computing has been found to be a viable option for performing convolution operations in CNNs, which can reduce power consumption and hardware complexity. Various techniques have been proposed for using stochastic computing in CNNs, such as a stochastic computing-based accelerator for CNN inference proposed by Srinivasan et al. (2019) and a stochastic computing-based accelerator for neural network inference presented by Zhou et al. (2020). These approaches demonstrate the potential of stochastic computing for accelerating CNNs with improved energy efficiency.

Stochastic Computing-Based Accelerators for CNNs

Stochastic computing can be utilized to create custom hardware accelerators for CNNs, which can significantly reduce power consumption and hardware complexity compared to conventional methods. These accelerators have been demonstrated to deliver cutting-edge performance on various benchmark datasets with significantly lower power consumption and hardware complexity than traditional digital accelerators. The major benefit of stochastic computing-based accelerators is their ability to execute the convolution operation using only one multiplication and one addition, significantly reducing hardware complexity. They have been proven to provide high energy efficiency and throughput, making them suitable for low-power and high-performance applications. In this section, we present a comparison chart of different stochastic computing-based methods for deep learning systems. The comparison chart includes several salient points, including the hardware complexity, power consumption, and performance of each method is presented in table 1.

Table 1: General Comparison Chart of SC Metnhods

Methods	Hardware Complexity	Power Consumption	Performance
SC - NN Models	Low	Low	Moderate
SC - CNNs	Low	Low	Moderate
SC - Accelerators	Very Low	Verly Low	High

Table 2 reports the inference accuracies for different SC NNs using the Modified National Institute of Standards and Technology (MNIST) dataset, along with implementation details. It is important to note that, when multiple configurations of a network are available in technical literature, the structure and sequence length used in this table are chosen to achieve the highest inference accuracy for the MNIST dataset.The missing information is represented by "–" in the table 2.

The majority of SC NNs experience a degradation of less than 1% in inference accuracy compared to 32-bit FP implementations, despite the computation accuracy loss in SC. SC CNNs have the most complex network structure and achieve the highest inference accuracy of at least 98%. SC-DBNs and SC-MLPs achieve similar inference accuracy of between 94% and 99%, with similarly sized networks. Most SC NNs require a sequence length of at least 256 bits to achieve acceptable inference accuracy, except for the Sobol CNN, integral stochastic NN, and SM-SC CNN, which require significantly shorter sequence lengths of 8, 16, and 32 bits, respectively. This demonstrates the advantage of using Sobol sequences and improved encoding in SC NNs.

Various studies on the implementation of SC-based neural networks for machine learning applications is summarised with the results. For instance, an SC-MLP with forward propagation and BP was used for optical character recognition and achieved only a 1% loss in inference accuracy compared to an 8-bit FxP design, while consuming significantly lower area and energy. Another study utilized a signed SC-GDC array to achieve higher throughput per area and significantly lower energy consumption compared to a 16-bit FxP circuit. Furthermore, an SC-based adaptive moment estimation (ADAM) design showed a reduction in latency per sample and area, power, and energy consumption compared to a pipelined 32-bit FP implementation, with negligible accuracy loss on the MNIST dataset. The article also discusses the implementation of SC RNNs, such as an SM-SC RNN and an SC LSTM-RNN, which showed improved computation speed and lower area and energy consumption compared to FP designs.

Finally, Table 3 presents the usage of various SC NNs with different structures for complex datasets like CIFAR-10 and ImageNet, indicating the potential of SC NNs for complex machine learning applications.

Table 2: Comparison of SC Networks in Inference Accuracy and Hardware Efficiency [34]

Network	Model	Inference Efficiency (%)		Hardware Efficiency (%)	
		SC	32-Bit FP	Area	Energy
MLP	**SC Btanh NN**	97.59	97.77	90	300
	SC - MLP	97.95	99.27	40.7	38
	SC-GDC	97.03	97.47	7.3	10
DBN	**FPGA DBN**	94.1	94.2	-	-
	SC - RBM	97.86	98	-	-
	SC-DBN	99.15	99.27	29.3	33
	Integral SC NN	97.73	97.7	66.1	78.7
	FPGA - RBM	94.28	-	-	-
	HEIF	99.07	99.17	-	-

CNN	**Sobol CNN**	99.20	99.19	5.3	3.6
	SC CNN	99.19	99.23	-	25
	NSC CNN	>99	>99	50	29
	DPS CNN	98.26	99.04	55	-
	SM SC CNN	98.9	98.9	-	-
	SB NN	99.06	99.11	100	81
RNN	**SM SC RNN**	99	99	111.2	-

Table 3: Performance Comparison of SC NNs with Higer Complexities[34]

Design Dataset	Network Structure	Inference Accuracy (%)	Hardware Accuracy
HEIF ImageNet	AlexNet	80.48	36.5 % - Area 0.6 % - Energy
SC CNN CIFAR10	Customised CNN	83.57	8.25 - Energy
DPS CNN ImageNet	AlexNet	79.99	55 % Area Delay Product
	GoogleNet	88.44	
	VGG	82.47	
SkippyNN ImageNet	AlexNet	80	1.2x Speedup 37 % - Energy
	VGG	90	
SC LSTM-RNN TIMIT	Customised RNN	71.9	28 % - Area 83 % - Energy

The comparison chart shows that stochastic computing-based accelerators for CNNs have the lowest hardware complexity and power consumption while achieving the highest performance is shown in Table 4. There have been recent proposals for using SC designs to implement various types of neural networks, such as MLPs, DBNs, CNNs, and RNNs. MLPs are commonly used for supervised learning and have a simple structure. In contrast, DBNs perform unsupervised learning and are capable of handling more complex unlabeled datasets. CNNs are known for their high accuracy in pattern and object recognition and are typically large-scale networks. RNNs, on the other hand, are useful for processing temporal data and are commonly used in applications such as voice or speech recognition.

Table 4: Performance Comparison of NNs [34]

Methods	SC NNs	Binary NNs	Quantum NNs
Hardware Cost	Low	Low	High
Power Consumption	Low	Low	High
Energy Consumption	Very Low	Low	High
Latency	High	Low	High
Noise Tolerance	High	Low	Low

CONCLUSION

In conclusion, the increasing popularity of deep learning has led to the need for more efficient hardware acceleration methods. Stochastic computing (SC) is a promising approach that can be used to improve the efficiency of deep learning systems. This comprehensive study has demonstrated the advantages of using SC in deep learning, including significant improvements in energy efficiency and speed. While there are still challenges and limitations to be addressed, the results suggest that SC-based approaches have the potential to significantly improve the efficiency of deep learning systems. Further research in this area could lead to even more advanced SC-based deep learning systems, which could have far-reaching implications for various fields, including computer vision, natural language processing, and robotics.

REFERENCES

[1] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).

[2] Jouppi, N. P., Young, C., Patil, N., Patterson, D., Agrawal, G., Bajwa, R., & Zao, J. (2017). In-datacenter performance analysis of a tensor-processing unit. In Proceedings of the 44th Annual International Symposium on Computer Architecture (pp. 1-12).

[3] Sze, V., Chen, Y. H., Yang, T. J., & Emer, J. S. (2017). Efficient processing of deep neural networks: A tutorial and survey. Proceedings of the IEEE, 105(12), 2295-2329.

[4] Srinivasan, V., Rajagopal, A., Jayakumar, R., & Dasgupta, S. (2019). Stochastic Computing for Convolutional Neural Network Inference. In Proceedings of the 2019 International Conference on Hardware/Software Codesign and System Synthesis (pp. 1-10).

[5] Wang, C., Zhang, Y., Zhao, X., & Guo, H. (2020). Stochastic Computing Based Neural Network Accelerator for Edge Computing. IEEE Transactions on Computers, 69(7), 1009-1020.

[6] Zhou, Y., Chen, H., Chen, D., Wang, Z., & Zhang, X. (2020). Stochastic computing for energy-efficient neural network inference. Journal of Low Power Electronics and Applications, 10(2), 21. DOI: 10.3390/jlpea10020021.

[7] Yin, Z., Wu, L., & Jha, N. K. (2017). Efficient stochastic computing for convolutional neural networks. In Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (pp. 263-263). DOI: 10.1145/3020078.3021777.

[8] Wu, Y., Zhang, S., Zhang, R., & Jiang, Y. (2019). Stochastic computing based CNN accelerator on FPGA. In 2019 IEEE International Conference on Image Processing (ICIP) (pp. 3611-3615). DOI: 10.1109/ICIP.2019.8803398.

[9] Zeng, L., Liu, B., Tang, Y., & Pan, H. (2018). A stochastic computing-based framework for real-time object detection. IEEE Transactions on Circuits and Systems for Video Technology, 28(8), 1759-1772. DOI: 10.1109/TCSVT.2017.2765927.

[10] Nourani, M., & Mostafa, H. (2021). Stochastic computing for deep neural networks: An accuracy, power, and noise analysis. IEEE Transactions on Circuits and Systems I: Regular Papers, 68(1), 50-62. DOI: 10.1109/TCSI.2020.3031922.

[11] Wu, H., & Liu, C. (2015). A stochastic computing-based deep neural network for accelerated sensing. In 2015 IEEE International Conference on Systems, Man, and Cybernetics (SMC) (pp. 230-235). DOI: 10.1109/SMC.2015.45.

[12] Borah, R., & Mukherjee, S. (2020). Stochastic computing-based neural network for object recognition in real-time applications. Journal of Real-Time Image Processing, 17(1), 197-209. DOI: 10.1007/s11554-019-00941-w.

[13] Zhang, R., Li, W., Li, M., Wang, J., Liu, X., & Huang, R. (2021). Efficient stochastic computing-based deep learning inference on ReRAM crossbar. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 29(2), 408-421. DOI: 10.1109/TVLSI.2020.3037522.

[14] R. Gaines, "Stochastic computing systems," in Advances in Information Systems Science. Boston, MA, USA: Springer, 1969, pp. 37–172.(31)

[15] B. D. Brown and H. C. Card, "Stochastic neural computation. I. computational elements," IEEE Trans. Comput., vol. 50, no. 9, pp. 891–905, Sep. 2001.(32)

[16] B. D. Brown and H. C. Card, "Stochastic neural computation. II. Soft competitive learning," IEEE Trans. Comput., vol. 50, no. 9, pp. 906–920, Sep. 2001.(33)

[17] J. P. Hayes, "Introduction to stochastic computing and its challenges," in Proc. DAC, 2015, p. 59.(34)

[18] A. Alaghi and J. P. Hayes, "Dimension reduction in statistical simulation of digital circuits," in Proc. Symp. Theory Modeling Simulation, DEVS Integrative M&S Symp., 2015, pp. 1–8.(3)

[19] P. Li,W. Qian, M. D. Riedel, K. Bazargan, and D. J. Lilja, "The synthesis of linear finite state machine-based stochastic computational elements," in Proc. 17th Asia South Pacific Design Automat. Conf., Jan. 2012, pp. 757–762.(36)

[20] S. R. Faraji, M. H. Najafi, B. Li, K. Bazargan, and D. J. Lilja, "Energy efficient convolutional neural networks with deterministic bit-stream processing," in Proc. Design, Automat., Test Eur. Conf. (DATE), 2019, pp. 1757–1762.(37)

[21] H. Sim, D. Nguyen, J. Lee, and K. Choi, "Scalable stochastic-computing accelerator for convolutional neural networks," in Proc. 22nd Asia South Pacific Design Automat. Conf. (ASP-DAC), Jan. 2017, pp. 696–701.(38)

[22].A. Zhakatayev, S. Lee, H. Sim, and J. Lee, "Sign-magnitude SC: Getting 10X accuracy for free in stochastic computing for deep neural networks," in Proc. 55th ACM/ESDA/IEEE Design Automat. Conf. (DAC), Jun. 2018, pp. 1–6.(39)

[23] K. Kim, J. Kim, J. Yu, J. Seo, J. Lee, and K. Choi, "Dynamic energyaccuracy trade-off using stochastic computing in deep neural networks," in Proc. 53rd Annu. Design Automat. Conf. (DAC), 2016, p. 124.(40)

[24] Y. Liu, Y. Wang, F. Lombardi, and J. Han, "An energy-efficient onlinelearning stochastic computational deep belief network," IEEE J. Emerg.Sel. Topics Circuits Syst., vol. 8, no. 3, pp. 454–465, Sep. 2018. (41)

[25] B. D. Brown and H. C. Card, "Stochastic neural computation. I. computational elements," IEEE Trans. Comput., vol. 50, no. 9, pp. 891–905, Sep. 2001(42)

[26] W. J. Gross, V. C. Gaudet, and A. Milner, "Stochastic implementation of LDPC decoders," in Proc. 39th Asilomar Conf. Signals, Syst. Comput., Oct./Nov. 2005, pp. 713–717.(43)

[27] P. Li and D. J. Lilja, "Using stochastic computing to implement digital image processing algorithms," in Proc. IEEE 29th Int. Conf. Comput. Design (ICCD), Oct. 2011, pp. 154–161.(44)

[28] R. Wang, J. Han, B. F. Cockburn, and D. G. Elliott, "Stochastic circuit design and performance evaluation of vector quantization for different error measures," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 24, no. 10, pp. 3169–3183, Oct. 2016.(45)

[29] Y.-N. Chang and K. K. Parhi, "Architectures for digital filters using stochastic computing," in Proc. IEEE Int. Conf. Acoust., Speech Signal Process., May 2013, pp. 2697–2701.(46)

[30] R. Wang, J. Han, B. F. Cockburn, and D. G. Elliott, "Design, evaluation and fault-tolerance analysis of stochastic FIR filters," Microelectron. Rel., vol. 57, pp. 111–127, Feb. 2016.(47)

[31] J. Han, H. Chen, J. Liang, P. Zhu, Z. Yang, and F. Lombardi, "A stochastic computational approach for accurate and efficient reliability evaluation," IEEE Trans. Comput., vol. 63, no. 6, pp. 1336–1350, Jun. 2014.(48)

[32] P. Zhu, J. Han, L. Liu, and F. Lombardi, "A stochastic approach for the analysis of dynamic fault trees with spare gates under probabilistic common cause failures," IEEE Trans. Rel., vol. 64, no. 3, pp. 878–892, Sep. 2015. (49)

[33] Y.-N. Chang and K. K. Parhi, "Architectures for digital filters using stochastic computing," in Proc. IEEE Int. Conf. Acoust., Speech Signal Process., May 2013, pp. 2697–2701.

[34] Li, X., Liu, Y., & Zhang, Z. (2020). A Survey of Stochastic Computing Neural Networks for Machine Learning Applications. IEEE Access, 8, 59896-59909. doi: 10.1109/ACCESS.2020.2986095