Stochastic Neural Network with Polarized Weights

Jiang Xiao*

Department of Physics and State Key Laboratory of Surface Physics,

Fudan University, Shanghai 200433, China

We propose a realization of artificial neural network based on stochastic computing, where the multiplication and non-linear activation function are both realized by simple logic (XNOR and Majority) operation of random numbers. The realized stochastic neural network contains only extremely simple logic operations, and can be easily parallelized and mapped into hardware design. Furthermore, the stochastic neural network can produce inference result in a progressive way.

I. INTRODUCTION

Stochastic computing is an innovative computational paradigm that leverages probabilistic representations of data to perform arithmetic operations and process information [1, 2]. Unlike traditional binary computing, which relies on precise values represented by bits, stochastic computing encodes information as sequences of random bits, where the probability of a bit being '1' corresponds to the value being represented. This approach allows for efficient computation, particularly in applications involving uncertainty, such as machine learning, signal processing, and error-tolerant systems. By exploiting the inherent noise and randomness in the computation process, stochastic computing can achieve significant reductions in hardware complexity and power consumption, making it an attractive alternative for resource-constrained environments. As research in this field continues to evolve, stochastic computing holds the potential to revolutionize how we approach complex computational tasks in various domains.

Stochastic computing has emerged as a promising approach for enhancing the efficiency and performance of neural networks, particularly in resource-constrained environments. In traditional neural networks, weights and activations are represented using precise floating-point values, which can lead to high computational and memory demands. Stochastic computing addresses this challenge by encoding these values as probabilities, represented by streams of random bits.

* Corresponding author: xiaojiang@fudan.edu.cn

In this framework, the probability of a bit being '1' corresponds to the value of the weight or activation, allowing for the representation of a wide range of values with reduced precision. This probabilistic representation enables the implementation of arithmetic operations, such as addition and multiplication, using simple logic gates, which can significantly lower the hardware complexity and power consumption of neural network architectures.

Moreover, stochastic computing can enhance the robustness of neural networks against noise and variations in data. By leveraging the inherent randomness in the computation process, neural networks can become more resilient to errors, making them suitable for applications in environments where reliability is critical, such as embedded systems and IoT devices.

Recent research has demonstrated that integrating stochastic computing into neural networks can lead to competitive performance compared to traditional methods, while also achieving substantial reductions in energy consumption and processing time. As the demand for efficient and scalable neural network solutions continues to grow, stochastic computing presents a compelling avenue for advancing the capabilities of artificial intelligence and machine learning systems [3, 4].

II. STOCHASTIC REPRESENTATION

There are various ways of representing a real number using stochastic bitstream of length N. Let $N_{0,1}$ (with $N_0 + N_1 = N$) be the number of 0s and 1s in the bitstream, we may use the probability of 1s to represent a number in [0,1]

$$x = \frac{N_1}{N} \in [0, 1], \tag{1}$$

which is called unipolar representation. We may also use the probability difference between 1s and 0s to represent a real number in $[-1, 1]$:

$$x = \frac{N_1 - N_0}{N} = 2\frac{N_1}{N} - 1 \in [-1, 1], \tag{2}$$

which is called bipolar representation. For our purpose of neural network, we need to operate in both negative and positive numbers, so we use bipolar representation throughout the paper.

Let B_x be a bitstream of length N representing a number $x \in [-1, 1]$ in bipolar representation. Due to its stochastic nature, the stochastic representation of a real number is not unique. For

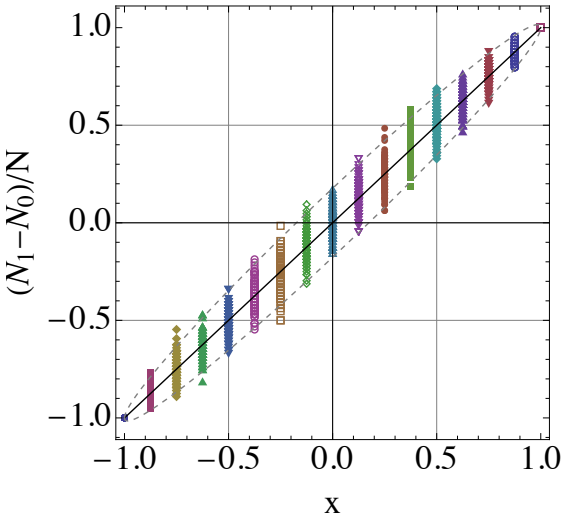

Figure 1. The bipolar representation of x has maximum error at $x = 0$. Simulation are performed with $N = 128$.

instance,

$$(01011010) = (11001100) = B_0 \tag{3a}$$

$$(01010010) = (10010100) = B_{-0.25}. \tag{3b}$$

We use B_x^k to denote the k-th bit of the bitstream in B_x.

To generate a stochastic representation of a real number $x \in [-1, 1]$, we may use following subroutine to generate the bitstream using a random number generator rand() $\in [0, 1]$:

$$\Theta\left[\text{rand}() - \frac{x+1}{2}\right]. \tag{4}$$

where $\Theta(x)$ is the Heaviside step function. The above subroutine is run N times to generate N random bits to form the bitstream representing x.

The inherent randomness of stochastic computing can hinder the reliable representation of values, as illustrated in Fig. 1. The uncertainty associated with N random bits is proportional to $\sqrt{N}$, resulting in a relative accuracy of approximately $1/\sqrt{N}$. To achieve a relative accuracy of 2^{-n}, a binary representation necessitates n bits, whereas a stochastic bitstream demands $N = 2^{2n}$ bits. Consequently, stochastic computing requires exponentially more bits to attain equivalent accuracy compared to traditional binary numbers, presenting a significant challenge. However, in the context of neural networks, which may not demand exceptionally high precision, stochastic computing offers potential advantages.

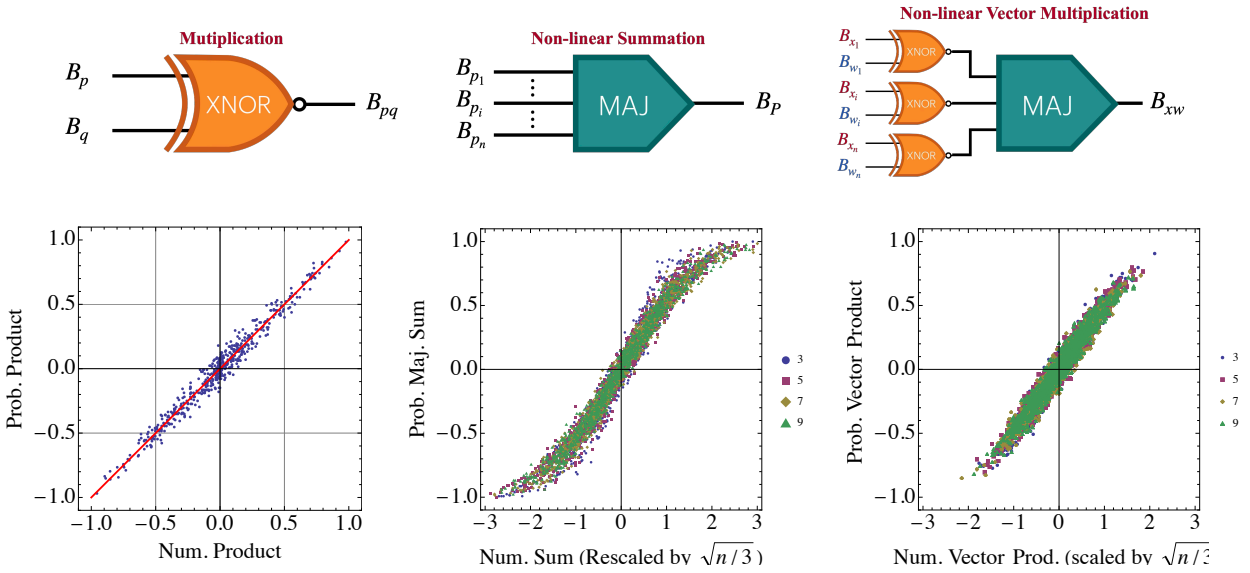

Figure 2. Left: the numerical product pq vs. probabilistic product $B_p \odot B_q$ of two numbers $p, q \in [-1, 1]$. Right: the numerical vector product $\mathbf{p}\cdot\mathbf{q}$ vs. probabilistic product with majority gate summation. Simulation are performed with $N = 256$.

III. STOCHASTIC ARITHMETIC OPERATIONS

A. Number Multiplication via XNOR Gate

The multiplication of two numbers can be simply realized by logic operation of bit-wise XNOR between the two bitstreams representing the two numbers:

$$B_p \odot B_q \simeq B_{pq} \quad \text{with} \quad B_p^k \odot B_q^k = B_{pq}^k, \tag{5}$$

where $\odot$ represents the bit-by-bit XNOR logic operation. A comparison of the probabilistic product and the exact numerical product is shown in Fig. 2(a). It should be noted that the error in the stochastic multiplication decreases as the bitstream length N increases.

B. Non-linear Summation via MAJORITY Gate

Let now consider the summation of multiple numbers within range $r_i \in [-1, 1]$. The numerical sum is estimated within the range:

$$R \equiv \sum_{i=1}^{n} r_i \sim \left(-\sqrt{n/3}, +\sqrt{n/3}\right). \tag{6}$$

The stochastic representation of p_i with N-bitstream is B_{p_i}, we define the majority summation as

$$B_R \equiv \coprod_{i=1}^{n} B_{r_i} \quad \text{with} \quad B_R^k = \coprod_{i=1}^{n} B_{r_i}^k \equiv \begin{cases} 1 & \text{more 1s than 0s in } B_{r_i}^k \text{ for } i = 1, \ldots, n \\ 0 & \text{otherwise} \end{cases} . \tag{7}$$

Fig. 2(b) shows the majority-summation with random $r_i \in [-1, 1]$ for $n = 3, 5, 7, 9$.

C. Quasi-linear Vector Multiplication via XNOR and MAJORITY Gates

Even though the simple probabilistic summation via MAJORITY gate above is non-linear, we shall see that the vector dot product realized by the combination of the XNOR multiplication and MAJORITY summation can be effectively linear. Considering two vectors $\mathbf{x} = \{x_i\}$ and $\mathbf{w} = \{w_i\}$ with $i = 1, \ldots, n$, we make the following operations

$$B_{xw} = \coprod_{i=1}^{n} B_{x_i w_i} = \coprod_{i=1}^{n} B_{x_i} \odot B_{w_i}, \tag{8}$$

which is similar to Eq. (7) but with the single number r_i to the MAJORITY gate replaced by a product of two numbers $x_i w_i$: $r_i \to x_i w_i$. Because all numbers are within the range $r_i, x_i, w_i \in [-1, 1]$, this replacement basically squeezes the range of inputs to the MAJORITY gate closer to zero, thus more into the linear regime in Fig. 2(b), as seen in Fig. 2(c). Such influences assume the inputs are purely random without correlation between x and w. However, if x and w are correlated in a similar direction, then it can reach the non-linear regime.

D. Non-linear Activation via MAJORITY Gate

A non-linear activation function with one input and one output, such as the sigmoid function or ReLU function, can also be realized by simple logic operations. For example, a MAJORITY gate can realize a Sigmoid-like activation function, with input bitstream with probability p_{in} and output bistream with probability p_{out}:

$$p_{\text{out}} = \frac{3p_{\text{in}} - p_{\text{in}}^3}{2}. \tag{9}$$

Similarly, an AND-OR gate realizes a ReLU-like activation:

$$p_{\text{out}} = \left(\frac{p_{\text{in}} + 1}{2} \right)^3 \tag{10}$$

The test of these probabilistic activation functions are shown in Fig. 3.

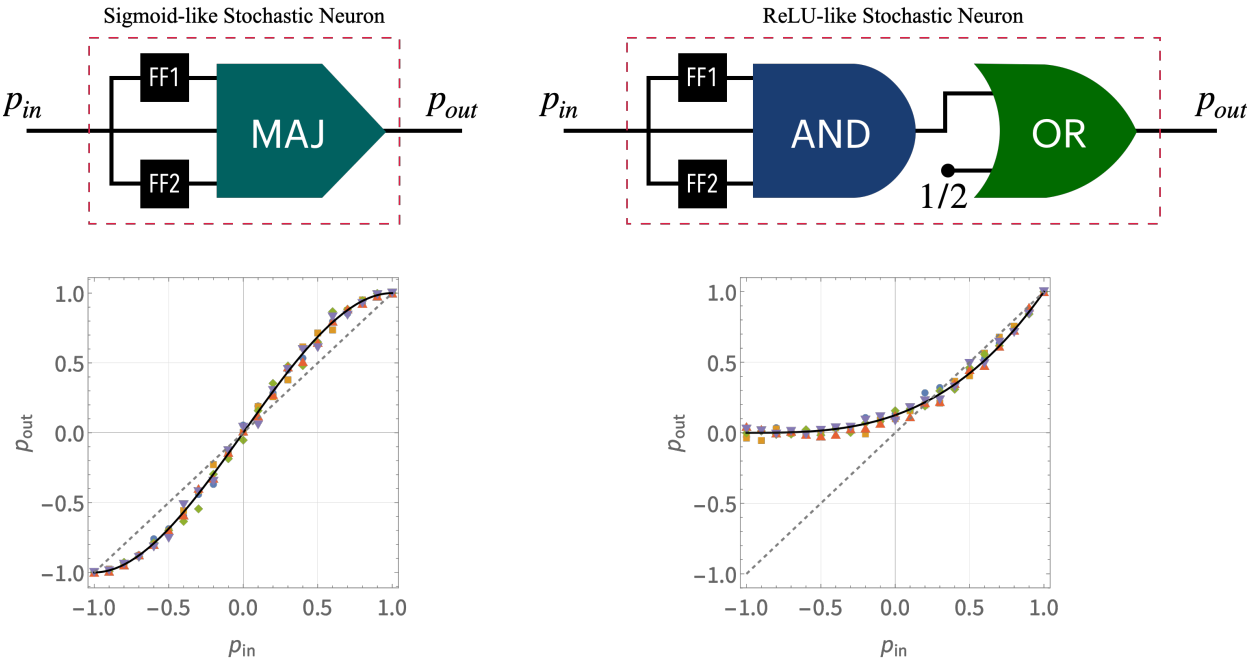

Figure 3. Sigmoid-like Stochastic Neuron and ReLU-like Stochastic Neuron.

IV. STOCHASTIC NEURAL NETWORK

With the fundamental elements above, we are ready to construct a stochastic neural network.

A. Convolution Kernel

We first consider building a convolutional kernel of size 3×3 performing the following operations:

$$y = \sum_{i,j=1}^{3} x_{ij} w_{ij} = \sum_{i=1}^{9} x_i w_i. \tag{11}$$

V. ACKNOWLEDGEMENTS

This work was supported by National Natural Science Foundation of China (Grants No. 12474110), the National Key Research and Development Program of China (Grant No. 2022YFA1403300), the Innovation Program for Quantum Science and Technology (Grant No.2024ZD0300103), and

Shanghai Municipal Science and Technology Major Project (Grant No.2019SHZDZX01).

---

[1] B. R. Gaines, Stochastic computing, in *Proceedings of the April 18-20, 1967, spring joint computer conference*, AFIPS '67 (Spring) (Association for Computing Machinery, New York, NY, USA, 1967) pp. 149–156.

[2] A. Alaghi and J. P. Hayes, Survey of Stochastic Computing, ACM Transactions on Embedded Computing Systems **12**, 92:1 (2013).

[3] M. W. Daniels, A. Madhavan, P. Talatchian, A. Mizrahi, and M. D. Stiles, Energy-Efficient Stochastic Computing with Superparamagnetic Tunnel Junctions, Physical Review Applied **13**, 034016 (2020), publisher: American Physical Society.

[4] M. W. Daniels, W. A. Borders, N. Prasad, A. Madhavan, S. Gibeault, T. Adeyeye, L. Pocher, L. Wan, M. Tran, J. A. Katine, D. Lathrop, B. Hoskins, T. Santos, P. Braganca, M. D. Stiles, and J. J. McClelland, Neural networks three ways: unlocking novel computing schemes using magnetic tunnel junction stochasticity, in *Spintronics XVI*, Vol. 12656 (SPIE, 2023) pp. 84–94.