Artificial Neural Network with Stochastic Learning Methodology

Muskaan Chaudhary
Department of Computer Engineering
Gujarat Technical University, Gujarat, India

Abstract— this paper provides taxonomy of artificial neural networks (ANNs) and furnishes the reader with the stochastic approach in ANN applications. ANN proposes very fascinating alternatives and other applications that play a major role in the world of computer science today. Researchers have successfully attempted to show those levels of intelligence on silicon, inspired by the advanced complexity of human brains, where hundreds of billions of interacting neurons process information in tandem. The approach surveyed effectively processes the learning knowledge inherent in the stochastic relation between the signal and the subsequent noise processes without the need to directly calculate back-propagation style equations.

Keywords: Artificial Neural Network, Stochastic Learning, ANN, Probabilistic Computation

I. INTRODUCTION

Many real-world issues have dynamic behavior, such as unpredictable and nonlinear behavior, requiring models to capture their true characteristics in order to achieve proper treatment [1] [2]. Some existing models are however limited being linear models (the implementation of which leads to contradictory or insufficient solutions to non-linear problems), as they need a complicated formulation, or also because they rely on certain a priori assumptions, requiring a comprehensive knowledge of the problem, which is often not available.

This contributed to the advancement of a generalized stochastic process model focused on neural networks to be extended to a wider variety of problems, including stochastic behavior phenomena [3] and/or normal characteristics of their probabilistic properties, such as mean and variance. This essentially non-linear model is termed as the Neural Stochastic Process (NSP), which captures the behavior of historical series to produce synthetic time series, close to that of historical series, applicable to solving various types of problems such as those concerning environment or economic phenomena.

A. Artificial Neural Network

Primarily, an Artificial Neuron is an engineering approach to biological neurons consisting of multiple inputs and a single output. ANN consists of a vast number of basic processing components that are both layered and intertwined with one another [4].

In the architecture of neural networks, a hidden layer is placed in between the input and output layers, which applies the weights to the input nodes that are later directed through some activation function as the output.

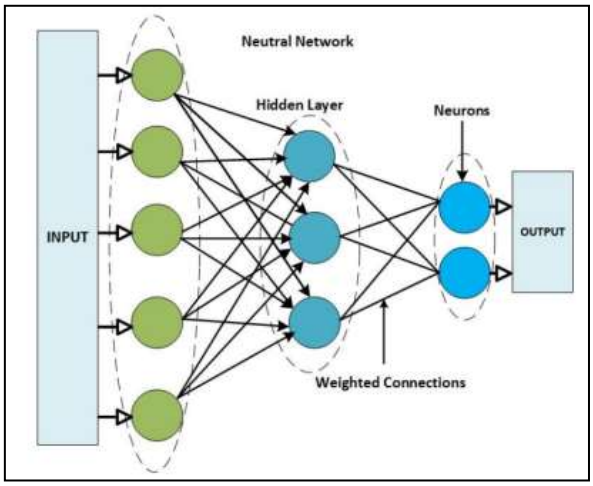

Fig. 1: Multilayered AN

A neural network is an instance of a distributed, parallel, data processing system. A neural network comprises a vast number of computing components or nodes with simple processing capacities, with the network size being the number of nodes. To form a fully connected or nearly fully connected network, the nodes are connected with each other. The network performs concurrent data processing centered on a constraint satisfaction model that has been shown to contribute to mutual computing capacities. The behavior of neural networks can be modeled as an iterative process. For each node, there is a special attribute associated. The value of each node is updated for each iteration, depending on the other node's value, which is usually a matrix-vector multiplication update operation. This algorithm for the update represents each node influencing the other network node. Many nonlinear operations, such as clipping or clamping, are often accompanied by the multiplication of the matrix-vector [5]. An associative memory system can be considered as one of the many uses of a neural network. For this specific system, if a collection of patterns, labeled 'library elements', has been stored in the network, by supplying the network with inconsistent or noisy information on the pattern in consideration, the user will recall a preferred pattern or library element. For example, if the library elements contain a set of two-dimensional images, a portion of one image can be entered into the neural network, which will return the entire image. A perceptual memory centered on neural network is versatile and fault-tolerant. The neural network, however, allows each node to have a separate processor. This can lead to substantial problems when integrating the network in the large systems necessary for practical applications.

1) Applications of Neural Networks

In reality, following this definition of neural networks (NNs), how their function and their real-world implementations and uses, NNs have widely applied in industry, education, economics, and in many areas of life problems. NNs are also applicable for intrusion detection, optimization method, and data classification. Classification is seen as a type of challenging optimization challenge. Most researchers have

used machine learning (ML) approaches to solve the issue of classification [6]. NNs are exemplary identifiers of patterns and data, like certain things listings, are ideal for forecasting and prediction needs. ANNs have been effectively used in estimating banks' success or loss and evaluating the stock market. It is also commonly used in weather and climate change forecasting, which is helpful in the protection and defense of resources such as infrastructure, environment, installations, homes, and transportation for human welfare [7]. In addition, ANNs have been successfully applied to various agricultural fields, such as remote sensing, especially in the classification of crop types and estimation of crop production.

2) ANN Classification

ANN can be categorized as illustrated in Fig. 2. A feed-forward neural network (FFNN) is a classification algorithm for machine learning which consists of ordered layers similar to human neuron processing units.

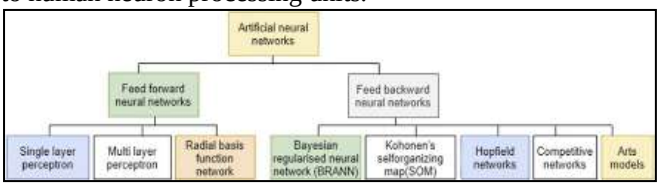

Fig. 2: ANN classification

Network connection weights calculate the probable value of network information. NN modules are also known as nodes. In the network, information transmission entails data entry from the input units and travels across the network, flowing from one layer to another layer before the output units are reached. If NN typically operates, that's when it functions as a classifier, and then there's no feedback between layers. In FFNN, information is only conveyed in one direction, from the input nodes to the hidden nodes, if there are any, and then to the output nodes. They are termed as feed forward neural networks for this behavior. Single layer perception and multi-layer perception are examples of FFNNs. The hidden layer is neither an input layer nor an output layer that is, Fig. 3, has a single hidden layer and 1 layer of output and displays all the relations in the layers between the modules. It is apparent that an individual layer links to the previous layer only.

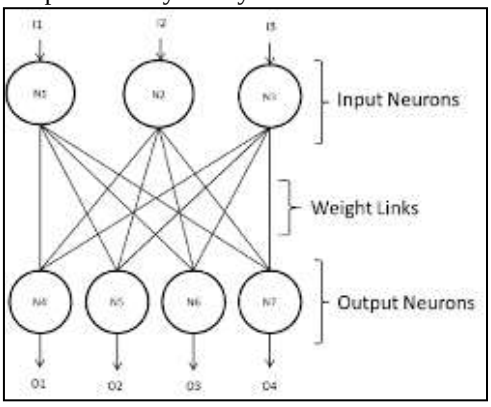

Fig. 3: Feed-forward neural network

FFNN implementations are summed up into two kinds, such as spaces, and dynamic systems management where classical machine learning methods are used. NNs having two or more hidden layers are labeled as deep networks because, with more than 1 hidden layer, the network has become complicated. Unlike FFNN, the internal state "memory" (store information) can be used by the feed-back neural network (FBNN) to process data input sequences. This suggests that according to first come first serve input bases, FFNN can logically manage tasks. Feed-backward NN can be used for things such as un-segmentation and pattern recognition (connected handwriting recognition). Relations between the nodes in feedback NNs or back propagation produce a structured graph in sequence [8]. The coordinated graph in the model for a time series contains feedback NNs to explain the complex terrestrial behavior. A training algorithm based on the back-propagation idea can be built in order to adjust the network parameters, taking into account the fact that this network has no repetitive connections between neurons. The estimated output is propagated through the hidden layers with dynamic filters back to the inputs [9]. As a consequence, Extended Dynamic Back Propagation can be presented [9]. In both the off-line and on-line modes, this algorithm will function. The choice of the right algorithm is based on the particular problem.

B. Stochastic Learning

Stochastic refers to a vector approach where there are some randomness and some ambiguity in the outcome [10]. It is a statistical term that is loosely connected to "probabilistic" and "randomness" and can be compared with the "deterministic" principle. In order to efficiently understand the behavior of several predictive models, the stochastic aspect of machine learning algorithms is an essential fundamental principle in machine learning and is required to be understood. A variable is stochastic if randomness or ambiguity is included as the occurrence of events or outcomes. "Stochastic" implies that there is some sort of randomness in the model [10]. A method is stochastic if one or more stochastic variables are regulated by it [11]. Card games and board games have a random aspect which makes them stochastic, such as shuffling or rolling a die.

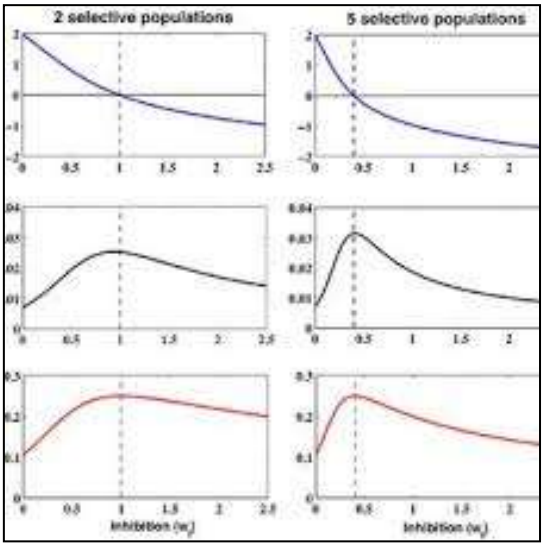

Fig. 4. Stochastic prediction graphs

1) Stochastic Optimization Algorithm

Stochastic optimization corresponds to an area of optimization algorithms that use randomness specifically to identify the optima of an objective function or maximize an objective function that has randomness (statistical noise)

itself [12]. Stochastic optimization algorithms more frequently attempt a balance between exploring the search space and using what has already been known about the search space in an attempt to hone in on the optimum. The options for the next location in the search space are picked stochastically, which is presumably based on the areas recently scanned. Stochastic hill-climbing chooses from among the uphill moves at random; with the steepness of the uphill step, the likelihood of selection will vary [12]. Simulated Annealing, Particle Swarm Optimization, and Genetic Algorithm are typical examples of stochastic optimization algorithms.

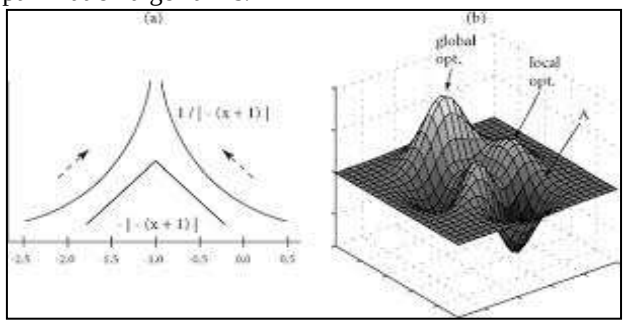

Fig. 5: Stochastic optimization algorithm

2) Stochastic Learning Algorithm

When the learning processes are recurring, many machine learning algorithm are termed as stochastic because of their random nature. Instead of being a bug, randomness has been useful as a feature. This helps the algorithms to stop being trapped and obtain outcomes that cannot be accomplished by deterministic (non-stochastic) algorithms [13]. For example, several machine learning algorithms, including Stochastic Gradient Boosting, Stochastic Gradient Descent (optimization algorithm), also have "stochastic" in their name. Stochastic gradient descent enhances a model's specifications, such as an artificial neural network, which requires arbitrarily shuffling the training dataset before every iteration, allowing the model parameters to be modified in various orders [13]. Furthermore, model weights are always initialized to a random starting point in a neural network. Many algorithms for deep learning are centered on an algorithm for optimization termed as stochastic gradient descent [11]. Stochastic gradient boosting is an algorithmic group of decision trees. The stochastic component points to the arbitrary subset of rows selected, precisely the split points of trees, from the training dataset used to build trees [11].

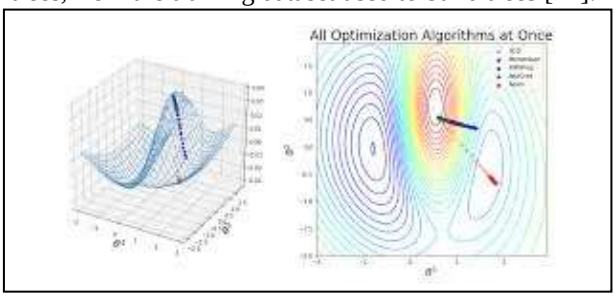

Fig. 6: Gradient boosting algorithm

II. PROBABILISTIC COMPUTATION

Probabilistic computation was adopted and established by two research groups during the 1960s [14]. The Standard Telecommunications Laboratories (England), formed by Gaines and Andreae, were the first of these groups. This group created an internal publication named Stochastic Analog Computer, which provided a technique for functional stochastic computation. At the University of Illinois, the second group was composed of Afuso and Poppelbaum, who specialized in the area of image processing systems. Both research groups are known to be the pioneers of stochastic computation.

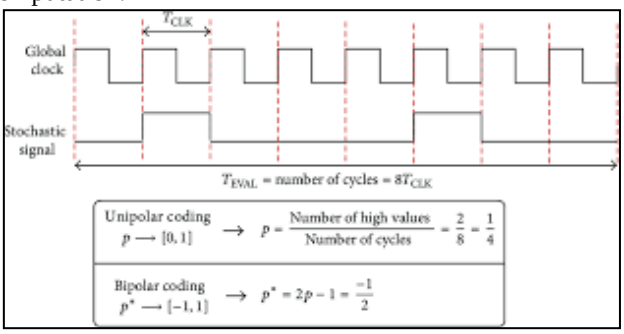

Fig. 7: Basic concepts of stochastic signals, unipolar, and bipolar coding

Stochastic computation is the result of the application of probabilistic laws to computer logic systems [15], enabling stochastic pulse frames to carry out pseudo analog operations (stochastic signals). A global clock presents the time period in stochastic-based simulations (Fig.7) during which all stochastic signals are stable (settled to high or low value). Each node poses a probability p of remaining in a high state within a clock cycle.

This probabilistic-based programming serves a natural way of using digital circuitry to work with analog quantities (as probabilities are identified between 0 and 1). When they are measured via logic gates, pulsed signals observe probabilistic rules [16]]. The AND gate, for example, produces an output signal along a switching probability directly proportional to the product of its inputs (i.e. the probability of collision between signals), while a multiplexer or an OR gate is used to execute the correlation between two switching signals [16]. Using this probabilistic framework, negative numbers cannot be explicitly represented, because probabilities are described within the [0, 1] range. To achieve negative numbers, we must make a vector shift p*= 2 p − 1 (switching probability of the signal is denoted by p), leading to the bipolar code (Unlike simplified unipolar coding, which is based on the straightforward use of probability) [17]. Accordingly, p being delimited between 0 and 1 makes p* bound to the range [-1, 1], and the value of zero is placed at p = ½. An advantage of this notation is that a single logic gate (XNOR gate) also implements the product. The representation set, however, is still confined to a limited interval [−1, 1]. A potential solution to this issue may be normalization (i.e. the use of modified magnitudes for signals in such a way that their functional values are held within the permitted interval [−1, 1]) [18].

One of the areas where stochastic computing can provide greater advantages is the implementation of parallel computing systems, provided that conventional parallel processing frameworks have the downside of needing lots of hardware blocks/cells to execute functions [19]. As a consequence, the amount of functions that traditional processes can execute in parallel is limited when opposed to

stochastic ones. Nevertheless, there are many drawbacks in stochastic logic, such as the need for a huge proportion of uncorrelated arbitrary signal generators to generate pulsed signals (representing a considerable amount of logic components and integration area) and the limited precision of the signal. Historically, the use of stochastic reasoning in the domain of hardware implementation of NN poses the downside of not being able to use the right weights and bias values given by standard training algorithms, due to the limitation of the stochastic signal representation range [-1, 1] [20]

A. Probabilistic Computing in Artificial Neural Network

With a few hardware resources, simple ANNs can be implemented. Artificial neurons are essential components of computation that conduct a basic process over the inputs. For every neuron output y_i, there is an input x_j relatively, according to,

$$y_i = f(U_i) = f\left(\sum_{j=1}^{n}(w_{ij}x_j) + b_i\right) \quad (1)$$

Where the relation weight between, the jth and the ith neuron is expressed by w_{ij}, while b_i is the ith neuron bias. The selection of the f(x) function depends on the type of neuron that is required. It may be a function of identity (for linear neurons), a nonlinear function of more complexity, such as log-sigmoid or tan-sigmoid, or function of threshold [21].

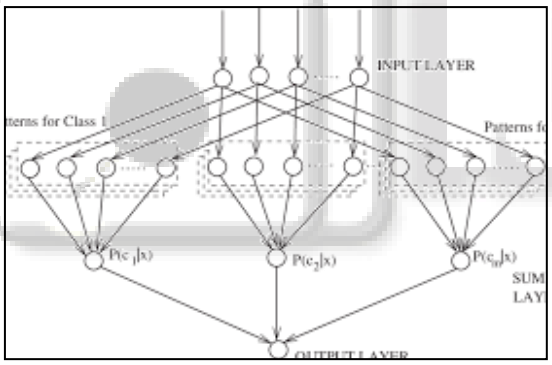

Fig. 8: Probabilistic Neural Network Model

III. STOCHASTIC NEURAL NETWORK MODELS

A continuous-time class containing n units of recurrent neural network is represented by the following equations,

$$\frac{d}{dt}x_i(t) = -x_i(t) + \sum_j w_{ij}S_j + \xi_i(t), \quad i = 1,2,\dots,n \quad (2)$$

Here, the internal state of ith unit is denoted by $x_i(t)$, whereas, the time-independent weight of the relation between the unit ith and jth is represented by w_{ij} and, the ith unit signal is denoted by S_i. The non-linear function for signal is specified as,

$$S_i = g[x_i(t)] = \frac{1}{1+\exp(-x_i)} \quad (3)$$

Where, the logistic function is represented by g. As a representative class of recurrent neural networks, the current model in question does not reduce or limit the generality of subsequent development, since several versions of the additive neural network can be equally transformed into the elementary modules mentioned by (2) and (3) [22]. The Gaussian white noise injection (with unit variance and zero mean) is concluded in (2), which is,

$$E[\xi_i(t)] = 0, \qquad E[\xi_i(t)\xi_j(\sigma)] = \delta_{ij}\delta(t-\sigma) \quad (4)$$

Where expectation operator is denoted by $E[\cdot]$, $\delta(t)$ represents the Dirac delta function, and the Kronecker delta is δ_{ij}. $\xi_i(t)$ representing input noise can be classified either as intrinsic system noise (thermal noise), artificial noise signal (externally controlled) or a network dynamics simulation error induced by noisy system environments or unknown variables.

An important factor at notice is that the internal state vector (n-dimensional) is a random variable x(t), considering a degree of uncertainty caused by $\xi(t)$, which represents additive noise term. Here, (2) can be viewed as the stochastic differential equation Ito, from a quantitative mathematical perspective.

$$dx(t) = f(x)dt + d\beta_t, \qquad x(0) = x^o \quad (5)$$

The initial state given is denoted by x°, whereas the standard limit techniques corresponded with the Brownian diffusion process β_t acquires the stochastic differential denoted by d β_t, and

$$f_i(x) = -x_i(t) + \sum_j w_{ij}S_j(x_j) \quad (6)$$

The formal relation that connects (2) and (5) is presented as,

$$\xi(t) = \frac{d}{dt}\beta_t \quad (7)$$

Here, (7) is viewed as a fictitious expression, considering Brownian motion being continuous but no differentiable; although on formal grounds, (2) can still be approximately justified. This method is contrary to the simulated annealing in which stochastic algorithms are implemented using the Boltzmann distribution.

A solution of the following Kolmogorov forward equation or Fokker-Planck is the probability density function p(x,t) correlated with the stochastic differential equations (2) or (5),

$$\frac{\partial p}{\partial t} = \frac{1}{2}\sum_i \frac{\partial^2 p}{\partial x_i^2} - \sum_i \frac{\partial}{\partial x_i}(f_i p) \quad (8)$$

The understanding of p(x; t) facilitates useful stochastic quantities such as averages, sensitivities, and variances to be computed. For more details on (5) and (8), readers are referred to the literature on stochastic differential equations by Skorohod [23] and others

IV. STOCHASTIC LEARNING LAWS FOR A LAYERED NETWORK

Here is the familiar layered network whose signal is denoted by Si at the output unit or hidden ith unit.

$$S_i = g\left(\sum_i w_{ij}S_j + \xi_i\right) \quad (9)$$

Where, logistic function is represented by g as described in (3). The Gaussian white noise is noise signal denoted by ξ_i with,

$$E[\xi_i] = 0, \qquad E[\xi_i\xi_j] = \delta_{ij} \quad (10)$$

When the input unit is ith, the respected θ_i external input is received and transmitted to other units.

$$S_i = \theta_i \quad (11)$$

Let the functional output be L(w, θ; ξ), in which w and θ represent the set of all w_{ij} and external input vector connection weights, respectively. The concern then becomes one of optimizing the average value, E[L(w; θ; ξ)], where, with respect to ξ and θ, E [·] signifies the expectation operator. The following corollary is there for this problem.

Corollary: The stochastic neural network method (9)–(11) with the learning algorithm under suitable smoothness [24] conditions guarantees a monotonic reduction in the average of the provided functional output L(w, θ; ξ).

$$\frac{dw_{ij}}{d\tau} = -\frac{\partial L}{\partial w_{ij}} - L(w, \theta; \xi)\xi_i S_j \quad (12)$$

It is noticed that the second term on the right-hand side of (12) depicts a stochastic connection, multiplied by the negative value of the performance function, between noise ξ_i and signal S_j. Readers are pointed to [24] for proof of the corollary.

V. Conclusion

The objective of this study was to discuss a general theoretical framework for stochastic computation with artificial neural network. The analysis algorithms contain the derivation with additive Gaussian white noise signals for neural networks. This paper considers some of the ambitious learning algorithms for Neural Stochastic Process models, such as probabilistic computing, logistic function, Stochastic Gradient Boosting, Stochastic Gradient Descent optimization algorithm. Convergence properties are commonly shared by all these algorithms.

The problem with neural machine learning is of the essence, so when the learning process is unsupervised and automated, evolvable knowledge will emerge. A globally convergent and quicker learning resource can be constructed for a general stochastic neural networks class by gradually decreasing the variance of the input noise phase (i.e. gradient descent's tuning parameter) to zero as well as by imitating the simulated annealing.

Reference

[1] O. Knill, *Probability Theory and Stochastic Processes with Applications*. 2021.
[2] KK, A. S. Weigend, and N. A. Gershenfeld, "Time Series Prediction: Forecasting the Future and Understanding the Past. Proceedings of the NATO Advanced Research Workshop on a Comparative Time Series Analysis Held in Santa Fe, New Mexico, 14-17 May 1992.," *J. Am. Stat. Assoc.*, 1994, doi: 10.2307/2290964.
[3] S. Haykin and N. Network, "A comprehensive foundation," *Neural Networks*, 2004.
[4] E. Y. Li, "Artificial neural networks and their business applications," *Inf. Manag.*, 1994, doi: 10.1016/0378-7206(94)90024-8.
[5] M. Fraccaro, S. K. Sønderby, U. Paquet, and O. Winther, "Sequential neural models with stochastic layers," 2016.
[6] R. Boutaba *et al.*, "A comprehensive survey on machine learning for networking: evolution, applications and research opportunities," *J. Internet Serv. Appl.*, 2018, doi: 10.1186/s13174-018-0087-2.
[7] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. E. Mohamed, and H. Arshad, "State-of-the-art in artificial neural network applications: A survey," *Heliyon*. 2018, doi: 10.1016/j.heliyon.2018.e00938.
[8] K. Xiang, B. N. Li, L. Zhang, M. Pang, M. Wang, and X. Li, "Regularized Taylor Echo State Networks for Predictive Control of Partially Observed Systems," *IEEE Access*, 2016, doi: 10.1109/ACCESS.2016.2582478.
[9] A. De Martino and D. De Martino, "An introduction to the maximum entropy approach and its application to inference problems in biology," *Heliyon*. 2018, doi: 10.1016/j.heliyon.2018.e00596.
[10] J. K. Blitzstein and J. Hwang, *Introduction to probability*. 2014.
[11] J. Siekmann, *Advanced Lectures on Machine Learning*. 2003.
[12] M. G. Kendall and B. B. Smith, "Randomness and Random Sampling Numbers," *J. R. Stat. Soc.*, 1938, doi: 10.2307/2980655.
[13] J. C. Spall, *Introduction to Stochastic Search and Optimization*. 2005.
[14] J. L. Doob, "Stochastic Processes and Statistics," *Proc. Natl. Acad. Sci.*, 1934, doi: 10.1073/pnas.20.6.376.
[15] J. Von Neumann, "Probabilistic logics and the synthesis of reliable organisms from unreliable components," *Autom. Stud.*, 1956.
[16] B. D. Brown and H. C. Card, "Stochastic neural computation I: Computational elements," *IEEE Trans. Comput.*, 2001, doi: 10.1109/12.954505.
[17] V. Canals, A. Morro, A. Oliver, M. L. Alomar, and J. L. Rossello, "A New Stochastic Computing Methodology for Efficient Neural Network Implementation," *IEEE Trans. Neural Networks Learn. Syst.*, 2016, doi: 10.1109/TNNLS.2015.2413754.
[18] B. R. Gaines, "Stochastic Computing Systems," in *Advances in Information Systems Science*, 1969.
[19] B. R. Gaines, "R68-18 Random Pulse Machines," *IEEE Transactions on Computers*. 1968, doi: 10.1109/TC.1968.226901.
[20] C. L. Janer, J. M. Quero, J. G. Ortega, and L. G. Franquelo, "Fully parallel stochastic computation architecture," *IEEE Trans. Signal Process.*, 1996, doi: 10.1109/78.533736.
[21] A. Torralba, F. Colodro, E. Ibáñez, and L. G. Franquelo, "Two Digital Circuits for a Fully Parallel Stochastic Neural Network," *IEEE Trans. Neural Networks*, 1995, doi: 10.1109/72.410370.
[22] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, 1986, doi: 10.1038/323533a0.
[23] S. Grossberg, "Nonlinear neural networks: Principles, mechanisms, and architectures," *Neural Networks*. 1988, doi: 10.1016/0893-6080(88)90021-4.
[24] P. A. P. Moran and A. V. Skorohod, "Studies in the Theory of Random Processes," *Rev. l'Institut Int. Stat. / Rev. Int. Stat. Inst.*, 1966, doi: 10.2307/1401784.
[25] M. Koda, "Stochastic sensitivity analysis method for neural network learning," *Int. J. Syst. Sci.*, 1995, doi: 10.1080/00207729508929062.

