A Conceptual Framework for Stochastic Neuromorphic Computing

Brian R. Gaines
University of Victoria, Victoria, BC, Canada
and University of Calgary, Calgary, AB, Canada

Editor's notes:
This keynote article is written by Brian Gaines, the inventor of stochastic computing. He shares both a view back on the history of neuromorphic computing and a view forward on deep learning as a new information processing technology. Gaines observes that computing has been a recursive technology: it supports other technologies that in turn support the progress of computing itself, leading to a positive exponential feedback loop and an exponential growth. He infers that the same holds for deep learning with its ability to meta-learn solutions to its own design problems.
—Ilia Polian, University of Stuttgart

It is significant to reflect, from a historic perspective on issues of engineering stochastic neuromorphic systems because the design objectives of the 1960s when the technology was first conceived were very similar to those of today, *to create technologies emulating human intelligence and incorporate them in systems emulating human capabilities*. It took far longer than expected to begin to achieve these objectives, but, after some six decades, there are now products coming into routine use that indicate that our early aspirations are achievable, and some of the technological innovations and conceptual frameworks of the earlier era that had no practical application at that time have become relevant to current research.

Stochastic computing (SC) is one such technology, invented in the early 1960s, that sustained a continuing low level of research activity for over four decades and then exhibits a major growth of research activity in the past decade as it became significant to the implementation of neuromorphic systems (Figure 1). The diversity of innovation and applications reported in recent publications on neuromorphic systems has become difficult to encompass conceptually, and integrative frameworks are needed to manage research strategies, understand the role of stochastic neuromorphic systems in mainstream information technology (IT), and develop principles for the design and test of systems based on the new technologies.

This article provides a framework for stochastic neuromorphic computing that situates it in the post war advances in modeling the brain that generated expectations that technology might soon be able to emulate and amplify human intelligence and in the infrastructure of IT that has evolved over the 80 years since the stored program digital computer was first invented. The evolution of SC and its current promises, challenges, and limits are already well-documented, and will not be detailed here [1]–[3].

Digital Object Identifier 10.1109/MDAT.2020.3031857
Date of publication: 16 October 2020; date of current version: 6 December 2021.

Post-war confluence of research in neurology and electronics

In the two decades following the second world war there was a remarkable confluence of research in the biological discipline of *neurology* and the engineering discipline of *electronics*. It had two sources: advances in electronic instrumentation were crucial

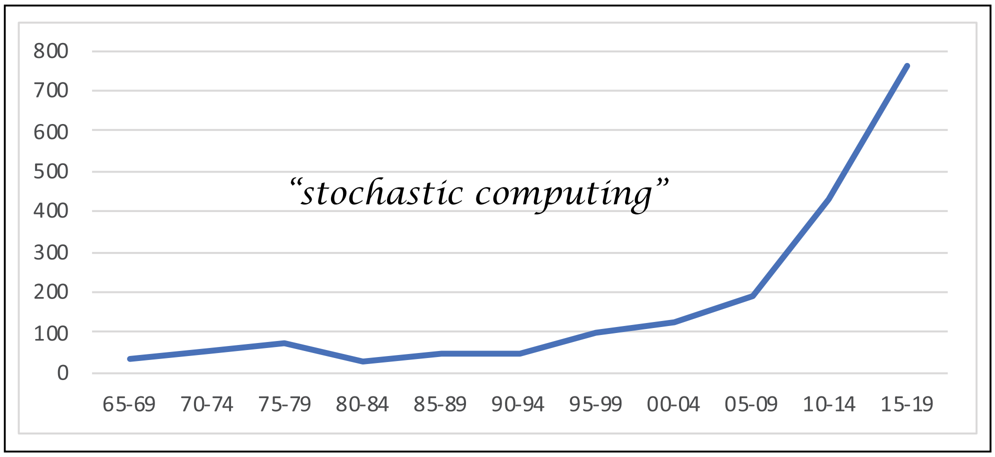

Figure 1. SC publication activity (noncumulative totals over five years intervals of publications including the phrase "stochastic computing" listed by Google Scholar).

to biological studies of information processes in the nervous system; and the mathematical models of the data collected were tested by simulating them in electronic circuits.

One outcome of this interdisciplinary interaction was that research results were published in both biological and engineering journals. For example, in 1959 the first of a major series of papers on the mechanisms of visual perception studied through microelectrodes capturing the electrical signal from single neurons [4] was published in the *Journal of Physiology*. A similar study titled "What the frog's eye tells the frog's brain" [5] that explicated the neurological basis of detecting and capturing moving prey was published in the *Proceedings of the IRE*.

Cutting-edge research in the discipline of neurobiology was being communicated to the discipline of electronics in an era where Wiener had already presented *cybernetics* as a systemic framework applicable to both biological and technological systems, and Steele was encouraging the design of technological systems based on biological analogies in his symposia on *bionics*. Papers on the electronic implementation of neurological theories to test their conformance with empirical data stimulated the design of circuits that were targeted on the replication of human capabilities such as pattern recognition and learning, even if those circuits were no longer behaving in the same way as their biological counterparts. Rosenblatt's [6] perceptron, Crane's [7] neuristor, and other similar innovations had biological origin but became targeted on engineering applications.

The neurological and technological research studies were also reported at conferences on cybernetics, bionics, and artificial intelligence (AI), generating expectations that technology with capabilities equivalent to human intelligence might soon be available. There is a remarkable continuity between the aspirations, research, and conceptual frameworks of the 1960s and those of the current era.

Advent of SC

One technology deriving from the interactions between neurological and electronics research was SC. The inspiration was derived by modeling the frequencies of the asynchronous pulse trains of neurons as generated by independent stochastic processes. From a bionic perspective, this suggested that an AND gate might provide a low cost multiplier device for analog computations if variables were represented by the generating probabilities of pulse trains [8], [9]. A research group at the University of Illinois led by Ted Poppelbaum investigated potential applications to image processing, and another at STL, ITT's U.K. research laboratory, led by John Andreae investigated applications to learning machines.

I worked for a year fabricating and testing mesa and planar transistors and tunnel diodes in the semiconductor research laboratories of ITT U.K., before going up to Cambridge in 1960 to study mathematics, theoretical physics and psychology, whilst working with John Andreae on learning systems and Richard Gregory on binocular vision and perceptual-motor coordination. I conjectured that the neural basis of depth perception based on disparity might be spatial correlation though the multiplication of asynchronous neural pulse trains, and modeled this mathematically and through simulation.

In 1964, I built an analog computer to study the training of pilots in a flight simulator, and a digital perceptron with discrete weights to emulate their learning behavior. I found that deterministic rounding of the perceptron's steepest descent calculation could lead to nonconvergent limit cycles even if a solution was available, but that random rounding did converge [10].

In 1965, I suggested to Andreae that the same technique could be used to provide learning elements for his STeLLA learning machine [11]. In designing the proof of feasibility prototype that STL implemented, I extended the technique to other computations relevant to machine learning such as Bayesian predictors, Markov modeling, and solutions to Laplace's equation [12], [13], and to a digital implementation of analog computing functionality [12].

The convergence issues with the digital perceptron without random rounding made it a useful "subject" for my research on adaptive training techniques [14], and I used it to develop a mathematical theory of training, proving that a randomized training sequence could also be used to ensure convergence [15]. Thus, the requisite variety [16] necessary for effective learning could be supplied either internally or externally by the introduction of uncertainty through a random process.

Early recognition of the role of randomness in neural networks is reflected today in studies of stochastic gradient descent (SGD) [17], neural networks with random weights [18], randomized algorithms for training neural networks [19], and in stochastic neuromorphic systems such as those presented in this special issue. Even if one implements SC primarily to achieve lower energy usage, one is also introducing some element of nondeterministic behavior which may itself be significant to achieving effective learning.

Comparison with technologies related to SC

The SC publication trajectory (Figure 1) shows: some growth for 15 years after the initial publications as the technology was replicated and enhanced; a decline over the next 15 years as no significant innovations ensued; a slow climb over the next 15 years as neural network research grew; and a major growth in the past decade as neuromorphic systems, such as audio and visual prosthetics, and deep learning in edge computing devices [20] required chips with very large numbers of devices but having low energy consumption [21] for their implementation [22].

To understand the pattern of research activity exhibited in Figure 1, it is useful to compare it with those of other related technologies for neuromorphic and analog computing, such as the analog computer, neuristor, and perceptron shown in Figure 2.

The analog computer publication trajectory shows: a major growth for 20 years; a major decline for 20 years as software running on the central processor unit (CPU) of a general-purpose computer replaced hardware implementations of analog computers; a plateau at a significant continuing level of publications; and growth in the domain-specific past 15 years as low cost, low power analog CMOS chips became a competitive alternative to software in many applications [23], [24] including neural networks [25]. The neuristor publication trajectory shows: a decline to a low level over 50 years; and a major growth in the past decade as memristor devices [26] were used to implement neuristors for neuromorphic computing systems [27].

The perceptron publication trajectory shows: a significant continuing interest for 40 years; and a massive rise in the past 30 years as advances in multilayer perceptron implementations, such as backpropagation [28], Boltzman machines [29], convolutional kernels [30], max pooling [31], SGD [17], and inception [32] were replicated and extended, leading to the deep learning architectures of today [33],[34].

The common feature of all these curves is a long period of stationary or declining interest followed by renewed attention to the technologies in recent years. A rationale for such revivals was given by the developers of Google's tensor processing unit (TPU) in noting the pitfall of "Being ignorant of history when designing a domain-specific architecture. Ideas that did not fly for general-purpose computing may be ideal for domain-specific architectures" [35, p. 57].

The following section discusses how the overwhelming success of mass market, general purpose, stored program computers undermined the market for domain-specific computer hardware, and how that situation has changed in the past decade ever since deep learning techniques became widely applied, recognized as the foundation of major new industries [36] and, in some major markets, require computational architectures with low energy requirements to be effective [37], [38].

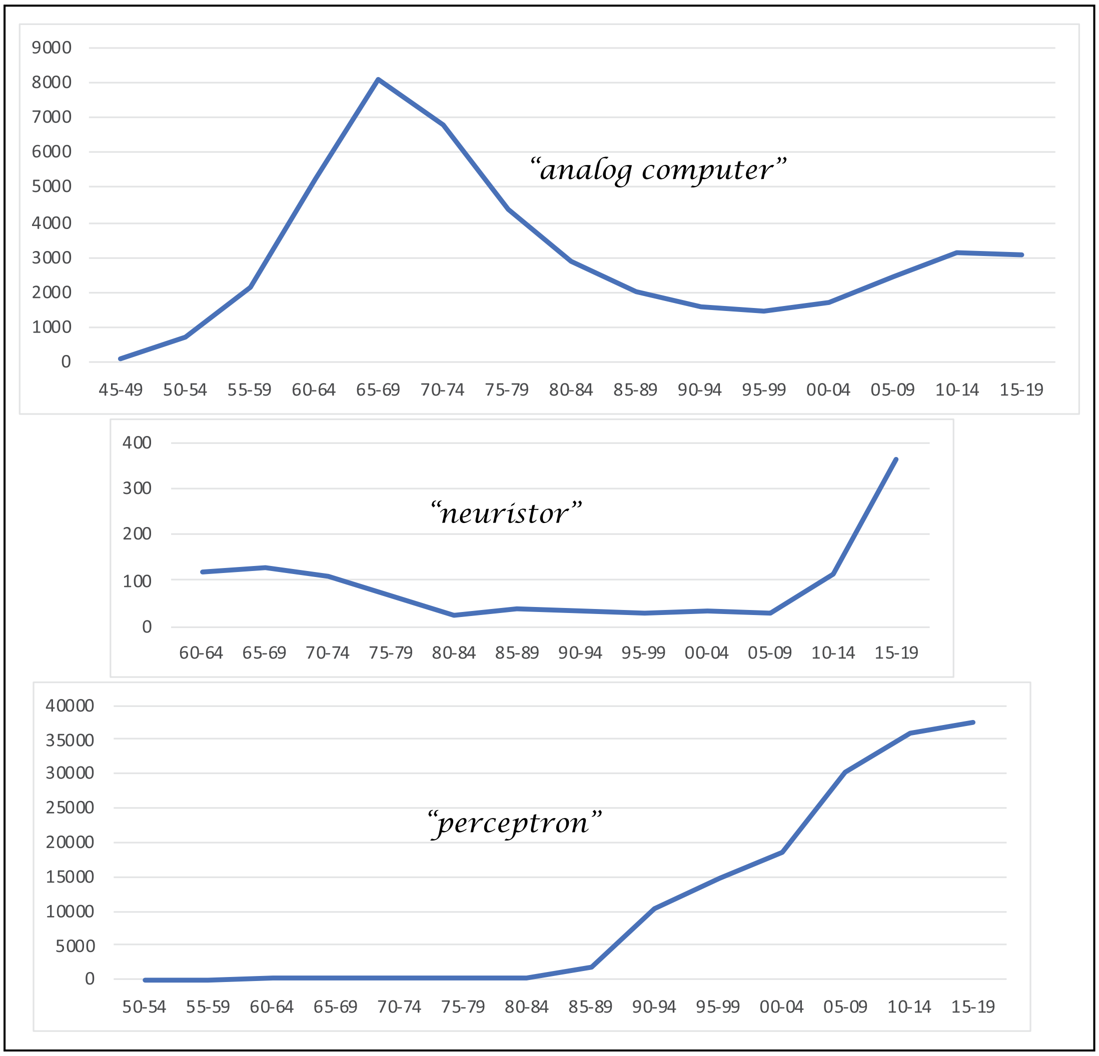

Figure 2. Research activity in other related technologies from publication counts.

Domain-specific versus general-purpose computers

The development of the silicon planar process in 1959 enabled large scale integration of circuits and systems on a single chip which facilitated the development of low cost, high speed, reliable general-purpose computers. The rapid growth of the number of transistors on a chip led to a high rate of performance improvement in computers continuing for over six decades.

Domain-specific computing hardware, such as an analog computer or digital differential analyzer, was replaced by an equivalent software virtual machine (VM) providing the same functionality on a general-purpose computer. Innovative domain-specific architectures addressing limitations of existing general-purpose computers quickly became irrelevant as advances in computer performance, real-time operating systems, software engineering, and algorithms overcame those limitations.

For example, in AI research microprogramming and coprocessors were used to provide functionality supporting declarative programming languages such as *Lisp* in the 1970s [39] and *Prolog* in the 1980s [40]. Some were marketed as domain-specific computers for AI applications. However, by the end of 1980s advances in the performance of general-purpose computers, compiler techniques and AI algorithms had undermined the competitive advantage of specialized AI computers.

Thus, computer architectures other than the general-purpose computer had no competitive advantage as hardware product innovations. The economic balance between hardware and software favored the mass produced, general-purpose computer that could provide diverse functionality. Any short-term performance advantage of domain-specific hardware was rapidly overcome by the performance trajectory of the general-purpose computer. To remain competitive the domain-specific hardware would have required continual upgrades, a requirement that could only be sustained for products with a massive market base such as general-purpose computers.

By the late 1980s ever-increasing number of devices on a chip enabled some coprocessors, such as the Intel 8087 floating point unit (FPU), to be included as part of the processor chip, and the usage of coprocessors declined. A significant exception to the migration from hardware to virual machines was the computation required for graphic displays which gradually migrated to graphics coprocessors in the late 1990s, a reverse transition where the existing graphics VM became implemented in hardware. The functionality required was readily integrated and evolved to become the multipurpose graphics processor unit (GPU) of today [41], a powerful multiprocessing coprocessor used not only for graphics but also for other number crunching functionality suited to parallel processing such as the matrix computations of deep learning.

Reconfigurable heterogeneous architectures and new devices

The 1990s saw the advent of the field-programmable gate array (FPGA) that enabled system designers to dynamically specify the interconnections on a reconfigurable chip providing a wide range of modules, digital and analog, such as gates and memory. Architecturally, FPGAs subsume concepts of microprogramming and coprocessors and blur the distinction between domain-specific and general-purpose computers. Their associated design automation tools enable system architectures to be specified in high-level languages, emulated, tested, and debugged, evaluated for performance and energy consumption, downloaded to chips and specified for manufacture.

FPGAs have made possible the rapid prototyping of new designs of a wide range of new architectures and potential products, including the implementation of SC systems. They facilitate the economic production of new products, even in small batches, and the trial implementation of new application-specific integrated circuits (ASICs).

In the past decade the inclusion of one or more general-purpose computer processors on FPGA chips has made it possible to build heterogeneous architectures on a single chip that provide the equivalent of domain-specific coprocessors, dynamic microprogramming, and other capabilities in application-specific systems.

This is the current technological infrastructure of IT for neuromorphic computing research and development: multiple core CPU chips provide what used to be the supercomputing capabilities of yesterday; GPU chips provide fast processing of matrix operations as well as a role model of what may be achieved with a specialized coprocessor; and FPGA chips provide a design testbed for new architectures and short-run production of application-specific systems, including coprocessors.

The current IT infrastructure is based on silicon chip technology, but there is ongoing research on new information technologies based on alternative materials and techniques, some of which require less energy and offer capabilities relevant to SC such as intrinsic probabilistic behavior [42].

Deep learning as a new information processing technology

In the past decade the accumulation of 50 years of research on neuromorphic computing technologies has led to superhuman achievements in tasks previously seen as requiring human capabilities, such as championship-level go and chess, language translation, speech recognition, medical diagnosis, security trading, discovery of new materials, driving vehicles, and diverse other domains [36], [37]. Neuromorphic architectures supporting deep learning have become widely recognized as a new information processing technology that may have as revolutionary an impact on society as that of the general-purpose computer [36].

Deep learning systems have anywhere from three to several hundred layers, each layer having a different structure and communicating with the next layer in varying ways. They can be envisioned as self-tuning adaptive digital filters transforming a complex input signal to a simpler output signal, with the layers extracting features similar to those of dictionary learning [43] in signal processing. They are not just black boxes as it is possible to visualize what features each layer is extracting [44], provide explanations of the output [45], [46], and analyze the learned computation to "squeeze" it to a smaller size without reducing accuracy [47]. In this regard, neuromorphic systems have valuable capabilities beyond those of the human brain.

There was no single breakthrough in the perceptron architecture, but rather a continuing sequence of incremental improvements commencing in the mid-1980s that continues today and shows no signs of tailing off. Schmidhuber's [33] monumental review of the evolution of deep learning details each new technique and its impact up to 2015, and Ferlitsch's [48] 2021 book on deep learning design patterns extends this to the state of the art today. Together they provide a daunting account of the complexity of the underlying technology that creates a steep learning curve for newcomers to the field.

Role of data sets in the design of deep learning systems

It is customary to focus on the improvements in the technology, but it is also important to consider the role of the challenge data sets that enabled variations in the technology to be evaluated. Such data sets have long been an essential factor in knowledge acquisition research [49], and it was the success a decade ago of deep learning techniques in modeling the massive data set of ImageNet [50] that led to the widespread recognition of a powerful new technology.

This also highlights that a high-quality, comprehensive data set is an essential component of the design and test of a deep learning system for a particular application—to a large extent the network designs itself from one subset of the data and tests itself on another part. The human design team specifies the hyperparameters of the architecture for learning, but the learning algorithms develop the parameters of the inferential, problem-solving system. Increasingly also, the design hyperparameters are themselves being adjusted through machine learning [51], [52].

Hardware support of deep learning systems

Deep learning algorithms are numerical computations that may be programmed for a general-purpose computer but have a preponderance of vector operations such that domain-specific support can greatly improve performance through parallel processing. CPU instruction sets have been extended to support the multiply and accumulate computations of deep learning, for example, the AVX-512 advanced vector instructions of Intel's Knights Landing CPU. GPU's already support similar computations for image processing and were found to provide 50× speedups in the modeling of large data sets such as ImageNet [33], [53].

The need for domain-specific processors was recognized early on, and several ASICs supporting neural network computations were commercially available by the early 1990s [54]. These included neural semiconductor's NU32 based on SC with 32 neurons and 1,024 synapses on a chip [55], and Hitachi's wafer-level integration module with 1,152 neurons and 73,700 synapses. In 2017, the bibliography of a comprehensive survey of neuromorphic computing hardware lists 2,682 publications [56]. It includes many based on SC or otherwise using random processes, and also several based on a variety of novel materials rather than silicon.

Google initially used GPUs to provide deep learning training and inference services in its datacenters but later developed its TPU using an ASIC coprocessor to increase speed and reduce energy consumption [35]. GPU manufacturers have also extended the instruction sets to offer better support for deep learning, and there are studies comparing the performance of current CPUs, GPUs, and TPUs on deep learning benchmarks [57].

Energy usage in learning

Modeling large data sets using deep learning techniques currently requires super-computer configurations having several hundred GPU's [58] or several thousand CPU's [53], consuming some 100 kW of energy and running domain-specific distributed processing software [59]. The Cerebras CS-1 neural net computer uses a wafer-scale integration chip with 1.2 trillion transistors, 400,000 processor cores and 18 gigabytes of SRAM, and consumes 20 kW.

The high power consumption of such implementations of deep learning algorithms are often contrasted with the 20 W of the human brain. However, when one considers that the end-to-end modeling of large image data sets involves the learning of the complete visual perception system, much of which is innate in humans, then the power requirements are roughly similar.

For example, a computer with an NVIDIA M40 GPU coprocessor consuming some 500 W and taking 14 days to model ImageNet [53] uses the same energy as a person taking one year to acquire not only the specialized classification system but also general object perception capabilities.

A more realistic comparison would be between the palaeontologist, Shubin, taking several weeks to learn to perceive fossils in the Arizona desert [60, pp. 63–64], and a neural network trained on a similar fossil image data set through transfer learning [61], [62] of the final layer of a network that had been pretrained on image classification (see [62]).

Thus, in some significant domains, powerful computer systems are substantially faster at learning than people. The total energy consumed does not appear to be greater, and is being substantially reduced as processors are optimized for deep learning applications.

Energy usage in inference

One advantage of computer learning over human learning is that the learned structure may be readily replicated and widely applied to make inferences for particular cases. The computational requirements for inferences are very much lower than for end-to-end leaning, particularly if postprocessing is applied to squeeze the net to the smallest size that maintains accuracy [47], [48].

Even for a squeezed net, the energy requirements of a conventional computer may still be excessive for some applications such as edge computing [63] where functionality is migrated from a server to a local peripheral, for example, speech recognition controlling home devices or automatic language translation in a cell-phone. The design of domain-specific, low-energy processor architectures and technologies for inference from neural networks is now a major research area [64], [65].

Google addresses such applications with the Edge TPU version of its TPU, a low-cost ASIC consuming only 2 W that is designed for inference with squeezed networks and can also support lightweight transfer training. Together with supporting software, such as TensorFlow-Lite, TF-Slim, and MobileNet [66], it provides cost, energy, performance, and software support targets for any competing technologies.

SC has become one of the major options for low-energy neural net inference accelerators as detailed in recent books [1], surveys [2], [22], [67], and doctoral theses [68]–[70]. There is a large literature on techniques and issues, such as SC Bayesian networks [71], optimizing neural networks for SC [72], high-speed inference with SC [73], scaling SC to large data sets [74]. It has also been shown that SC architectures and those based on low precision integer weights and random rounding [75] are equivalent [76].

Some implementations are based on novel materials such as memristors [70], spintronics [1], GeSe ovonic threshold switching [70], nanophotonics [77], and quantum-flux parametrons [70]. The topic mentioned last is of particular interest because the technology offers orders of magnitude improvements in clock speeds and energy usage relative to CMOS, intrinsic random number generation, and is also being heavily researched as the basis of a new generation of supercomputers [78].

Incremental continuous learning

The massive divide between the time, cost, and energy requirements of the end-to-end processing of very large data sets and the more economical inferencing from them once learned is reminiscent of that between mainframe batch processing and minicomputer interaction in the 1960s. Batch processing delays are not supportive of research and innovation, or of applications that require adaption to changing circumstances such as autonomous agents.

There is significant research on continuous lifelong learning with artificial neural networks [79]. Techniques, such as transfer training or those being developed for incremental deep learning [80], [81], may also enable low-energy processors initially designed for inference to be extended to include continuous learning capabilities.

This usually supplements, rather than replaces, large scale learning at a server. For example, in applications such as autonomous vehicles the collection of data from local experience is used to improve the base model distributed to all peripheral units. Local units may also be developing additional capabilities specific to their situations. The combination may be regarded as a technological emulation of the interplay of individual and collective distributed learning in human societies.

Theoretical foundations of deep learning

We have become accustomed to engineering disciplines having secure theoretical foundations that are themselves sources of innovation, such as Hertz's investigation of the potential inherent in Maxwell's equations of propagating electromagnetic waves in free space which led to the development of wireless transmission systems. Historically, however, commercial applications of many major technologies have been developed pragmatically prior to such foundations, and the theories explaining their success have followed later, in part as rationalizations of what has been achieved. For example, the steam engine had significant commercial applications in pumps and locomotives well before Carnot and Clausius developed their phenomenological theories of heat and Boltzmann reduced them to statistical mechanics—which, incidentally, can also be used to model deep learning processes [82].

Multilayer perceptrons are the latest example of such pragmatically evolving technologies, originating in bionic emulation of neurological systems and gradually accreting empirical improvements until they achieved unexpected success in significant applications [83], [84]. Developing theoretical foundations to explain that success and facilitate further improvements is now a major research area in its own right.

For example, one problem for gradient descent algorithms is being trapped by local minima but it is widely reported that this is not occurring, and a recent study has provided proof of convergence in polynomial time to a global minimum for the data set under assumptions applicable to most currently successful algorithms [85]—there is already a wealth of citing studies confirming and extending this result.

Another problem where theoretical foundations for deep learning are needed is that of avoiding overfitting models to data and losing generality and predictive power. For example, the empirical success of the dropout heuristic [86] needs mathematical foundations, as do the heuristic methods used to achieve tractability [87] of Bayesian neural networks [88].

There are also studies of deep learning that address its relationship to the architecture of the brain, for example, investigating alternatives to backpropagation algorithms that might achieve the same effect in a more brain-like way [89]. There is a parallel large-scale research activity also termed *neuromorphic computing* [90], [91] that is targeted on modeling the neurology of the brain rather than its bionic emulation. Such studies may suggest foundations and techniques for artificial neural networks even if this is not their primary objective.

We have come a long way in these six decades since Rosenblatt first proposed the perceptron, particularly in the last decade when research on algorithms for multilayer perceptrons finally broke through to achieve superhuman performance in many significant domains of human achievement. This has triggered a massive research endeavor—the search term "deep learning" retrieves 1,500 articles a month from Google scholar that report further improvements, theoretical foundations and applications in a very wide range of disciplines including major commercial utilization.

Deep learning is a major new engineering technology that can be used to design systems that emulate human high-level skills. A major part of the design process is automated knowledge acquisition from experience that has been generated through simulation, interaction with the world, or already captured in large data sets. We have, at last, learned to design significant learning machines.

This is only the beginning. We have much to do to improve the technology, provide secure mathematical foundations, learn to test systems that are massively parametrized and potentially discontinuous such that their behavior in some situations may not be indicative of that in similar situations, reduce technology and energy costs of large-scale learning, and make the opaque black boxes more transparent so that they are explicable in their behavior. All these tasks are being addressed in ongoing research programs that are reported in the massive literature.

Computer technology became the economic driver of our society in major part because it is a highly recursive technology that supports those technologies that support it through computer-aided design and manufacturing [92]. This creates positive feedback loops leading to exponential growth in capabilities.

Deep learning is also a recursive technology that can be applied to meta-learn solutions to its own design issues [93], such as automating the "art" [94, p. 287] of specifying SGD parameters [51] and other hyperparameters [52], selecting the best features to represent the raw data rather than using supplied human constructs [95], and generating its own experience rather than relying on skilled human examples [96]. It also supports the design and functionality of technologies that support it such as electronic devices, computers and compilers.

We are already seeing exponential growth in the technology and its applications resulting from the positive feedback loops thus created. The socio-economic impact is difficult to predict and quantify but has become widely recognized and debated. What is presented in this special issue is SC's contribution to reducing the cost and energy requirements of the new technology to facilitate its widespread application. ■

Acknowledgments

I am grateful to the editors for perceptive comments that have greatly improved this article. My publications and early SC material are available at: http://cpsc.ucalgary.ca/~gaines/reports.

■ References

[1] W. J. Gross and V. C. Gaudet, Eds., *Stochastic Computing: Techniques and Applications*. Cham, Switzerland: Springer, 2019.

[2] A. Alaghi, W. Qian, and J. P. Hayes, "The promise and challenge of stochastic computing," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 8, pp. 1515–1531, Aug. 2018.

[3] F. Neugebauer, I. Polian, and J. P. Hayes, "On the limits of stochastic computing," in *Proc. IEEE Int. Conf. Rebooting Comput. (ICRC)*, Nov. 2019, pp. 1–8.

[4] D. H. Hubel and T. N. Wiesel, "Receptive fields of single Neurones in the cat's striate cortex," *J. Physiol.*, vol. 148, no. 3, pp. 574–591, Oct. 1959.

[5] J. Y. Lettvin et al., "What the frog's eye tells the frog's brain," *Proc. IRE*, vol. 47, no. 11, pp. 1940–1951, 1959.

[6] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychol. Rev.*, vol. 65, no. 6, pp. 386–408, 1958.

[7] H. D. Crane, "The neuristor," *IEEE Trans. Electron. Comput.*, vol. EC-9, no. 3, pp. 370–371, Sep. 1960.

[8] B. R. Gaines, "Stochastic computer thrives on noise," *Electronics*, vol. 40, no. 14, pp. 72–79, Jul. 1967.

[9] S. T. Ribeiro, "Random-pulse machines," *IEEE Trans. Electron. Comput.*, vol. EC-16, no. 3, pp. 261–276, Jun. 1967.

[10] B. R. Gaines, "Techniques of identification with the stochastic computer," in *Proc. IFAC Symp. Problems Identificat. Autom. Control Syst.*, 1967, pp. 1–10.

[11] B. R. Gaines and J. H. Andreae, "A learning machine in the context of the general control problem," in *Proc. 3rd Congr. Int. Fed. Autom. Control*, 1966, pp. 342–348.

[12] B. R. Gaines, "Stochastic computing," in *Proc. Spring Joint Comput. Conf.*, vol. 30, 1967, pp. 149–156.

[13] B. R. Gaines, "Stochastic computing systems," in *Advances in Information Systems Science*, J. Tou, Ed., vol. 2, New York, NY, USA: Plenum Press, 1969, pp. 37–172.

[14] B. R. Gaines, "Training the human adaptive controller," *Proc. Inst. Electr. Eng.*, vol. 115, no. 8, pp. 1183–1189, 1968.

[15] B. R. Gaines, "Training, stability and control," *Instructional Sci.*, vol. 3, no. 2, pp. 151–176, Jul. 1974.

[16] R. W. Ashby, "Requisite variety and its implications for the control of complex systems," *Cybernetica*, vol. 1, no. 2, pp. 83–99, 1958.

[17] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proc. 19th Int. Conf. Comput. Statist. (COMPSTAT)*, 2010, pp. 177–186.

[18] W. Cao et al., "A review on neural networks with random weights," *Neurocomputing*, vol. 275, pp. 278–287, Jan. 2018.

[19] L. Zhang and P. N. Suganthan, "A survey of randomized algorithms for training neural networks," *Inf. Sci.*, vols. 364–365, pp. 146–155, Oct. 2016.

[20] X. Wang et al., "Convergence of edge computing and deep learning: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 869–904, 2nd Quart., 2020.

[21] E. García-Martín et al., "Estimation of energy consumption in machine learning," *J. Parallel Distrib. Comput.*, vol. 134, pp. 75–88, Dec. 2019.

[22] Y. Liu et al., "A survey of stochastic computing neural networks for machine learning applications," *IEEE Trans. Neural Netw. Learn. Syst.*, pp. 1–16, 2020.

[23] B. Razavi, *Design of Analog CMOS Integrated Circuits*, 2nd ed. New York, NY, USA: McGraw-Hill, 2017.

[24] Y. Tsividis, "Not your father's analog computer," *IEEE Spectr.*, vol. 55, no. 2, pp. 38–43, Feb. 2018.

[25] T. P. Xiao et al., "Analog architectures for neural network acceleration based on non-volatile memory," *Appl. Phys. Rev.*, vol. 7, no. 3, Sep. 2020, Art. no. 031301.

[26] R. Kozma, R. E. Pino, and G. E. Pazienza, Eds., *Advances in Neuromorphic Memristor Science and Applications*. Dordrecht, The Netherlands: Springer, 2017.

[27] J. del Valle et al., "A caloritronics-based Mott neuristor," *Sci. Rep.*, vol. 10, no. 1, p. 4292, Dec. 2020.

[28] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.

[29] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A learning algorithm for Boltzmann machines," *Cognit. Sci.*, vol. 9, no. 1, pp. 147–169, 1985.

[30] K. Fukushima, "Neocognitron: A hierarchical neural network capable of visual pattern recognition," *Neural Netw.*, vol. 1, no. 2, pp. 119–130, Jan. 1988.

[31] J. Weng, N. Ahuja, and T. S. Huang, "Learning recognition and segmentation using the Cresceptron," *Int. J. Comput. Vis.*, vol. 25, no. 2, pp. 109–143, 1997.

[32] C. Szegedy et al., "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[33] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.

[34] W. Liu et al., "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, Apr. 2017.

[35] N. P. Jouppi et al., "A domain-specific architecture for deep neural networks," *Commun. ACM*, vol. 61, no. 9, pp. 50–59, Aug. 2018.

[36] T. J. Sejnowski, *The Deep Learning Revolution*. Cambridge, MA, USA: MIT Press, 2018.

[37] S. Sengupta et al., "A review of deep learning with special emphasis on architectures, applications and recent trends," *Knowl.-Based Syst.*, vol. 194, Apr. 2020, Art. no. 105596.

[38] M. Kang, S. Gonugondla, and N. R. Shanbhag, *Deep In-Memory Architectures for Machine Learning*. Cham, Switzerland: Springer, 2020.

[39] Pleszkun and Thazhuthaveetil, "The architecture of lisp machines," *Computer*, vol. 20, no. 3, pp. 35–44, Mar. 1987.

[40] P. Van Roy, "1983–1993: The wonder years of sequential prolog implementation," *J. Log. Program.*, vols. 19–20, pp. 385–441, May 1994.

[41] T. M. Aamodt, W. W. L. Fung, and T. G. Rogers, *General-Purpose Graphics Processor Architectures*. San Rafael, CA, USA: Morgan & Claypool, 2018.

[42] P. Ienne, T. Cornu, and G. Kuhn, "Special-purpose digital hardware for neural networks: An architectural survey," *J. VLSI Signal Process. Syst. Signal, Image Video Technol.*, vol. 13, no. 1, pp. 5–25, Aug. 1996.

[43] I. Tošić and P. Frossard, "Dictionary learning," *IEEE Signal Process. Mag.*, vol. 28, no. 2, pp. 27–38, Mar. 2011.

[44] W. Samek et al., "Evaluating the visualization of what a deep neural network has learned," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 11, pp. 2660–2673, Nov. 2017.

[45] W. Samek, Ed., *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Cham, Switzerland: Springer, 2019.

[46] G. Vilone and L. Longo, "Explainable artificial intelligence: A systematic review," 2020, *arXiv:2006.00093*. [Online]. Available: http://arxiv.org/abs/2006.00093

[47] F. N. Iandola et al., "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and &lt;0.5MB model size," 2016, *arXiv:1602.07360*. [Online]. Available: http://arxiv.org/abs/1602.07360

[48] A. Ferlitsch, *Deep Learning Design Patterns*. Shelter Island, NY, USA: Manning Early Access Program (MEAP), 2021.

[49] J. Quiñonero-Candela et al., "Machine learning challenges. Evaluating predictive uncertainty, visual object classification, and recognising tectual entailment," in *Proc. MLCW*, 2005.

[50] O. Russakovsky et al., "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[51] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

[52] J. Lorraine, P. Vicol, and D. Duvenaud, "Optimizing millions of hyperparameters by implicit differentiation," in *Proc. 23rd Int. Conf. Artif. Intell. Statist.*, 2020, pp. 1540–1552.

[53] Y. You et al., "ImageNet training in minutes," in *Proc. 47th Int. Conf. Parallel Process.*, Aug. 2018, Art. no. 1.

[54] M. A. Holler, "VLSI implementations of learning and memory systems: A review," in *Proc. 3rd Int. Conf. Neural Inf. Process. Syst.*, 1990, pp. 993–1000.

[55] Max Stanford Tomlinson, D. J. Walker, and M. A. Sivilotti, "A digital neural network architecture for VLSI," in *Proc. IJCNN Int. Joint Conf. Neural Netw.*, vol. 2, 1990, pp. 545–550.

[56] C. D. Schuman et al., "A survey of neuromorphic computing and neural networks in hardware," 2017, *arXiv:1705.06963*. [Online]. Available: http://arxiv.org/abs/1705.06963

[57] Y. Wang et al., "Benchmarking the performance and energy efficiency of AI accelerators for AI training," 2019, *arXiv:1909.06842*. [Online]. Available: http://arxiv.org/abs/1909.06842

[58] L. Song et al., "Large-scale training system for 100-million classification at alibaba," in *Proc. 26th ACM

SIGKDD Int. Conf. Knowl. Discovery Data Mining, Aug. 2020, pp. 2909–2930.

[59] K. S. Chahal et al., "A Hitchhiker's guide on distributed training of deep neural networks," *J. Parallel Distrib. Comput.*, vol. 137, pp. 65–76, Mar. 2020.

[60] N. Shubin, *Your Inner Fish: A Journey Into the 3.5-Billion-Year History of the Human Body*. New York, NY, USA: Vintage Books, 2009.

[61] C. Tan et al., "A survey on deep transfer learning," in *Artificial Neural Networks and Machine Learning--ICANN*, V. Kůrková, Ed. Cham, Switzerland: Springer, 2018, pp. 270–279.

[62] S. Taghavi Namin et al., "Deep phenotyping: Deep learning for temporal phenotype/genotype classification," *Plant Methods*, vol. 14, no. 1, pp. 1–14, Dec. 2018.

[63] V. Sze et al., "Efficient processing of deep neural networks: A tutorial and survey," *Proc. IEEE*, vol. 105, no. 12, pp. 2295–2329, Dec. 2017.

[64] A. Basu et al., "Low-power, adaptive neuromorphic systems: Recent progress and future directions," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 8, no. 1, pp. 6–27, Mar. 2018.

[65] F. Conti, M. Rusci, and L. Benini, "The memory challenge in ultra-low power deep learning," in *NANO-CHIPS 2030: On-Chip AI for Efficient Data-Driven World*, B. Murmann and B. Hoefflinger, Eds. Cham, Switzerland: Springer, 2020, pp. 323–349.

[66] A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*. [Online]. Available: http://arxiv.org/abs/1704.04861

[67] T. J. Hamilton et al., "Stochastic electronics: A neuro-inspired design paradigm for integrated circuits," *Proc. IEEE*, vol. 102, no. 5, pp. 843–859, May 2014.

[68] Y. Liu, "Design and evaluation of stochastic computing neural networks for machine learning applications," Ph.D. dissertation, Dept. Electr. Comput. Eng., Univ. Alberta, Edmonton, Canada, 2019.

[69] A. Ardakani, "Complexity reduction of deep neural networks for efficient hardware implementations," Ph.D. dissertation, Dept. Electr. Comput. Eng., McGill Univ., Montréal, QC, Canada, 2020.

[70] R. Cai, "Emerging opportunities in machine learning hardware acceleration: From advanced neural networks implementation to ultra-efficient deep learning framework using next generation technology," Ph.D. dissertation, Dept. Electr. Comput. Eng., Northeastern Univ., Boston, MA, USA, 2020.

[71] C. S. Thakur et al., "Bayesian estimation and inference using stochastic electronics," *Frontiers Neurosci.*, vol. 10, pp. 1–15, Mar. 2016.

[72] J. Oh et al., "Retraining and regularization to optimize neural networks for stochastic computing," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI)*, Jul. 2020, pp. 246–251.

[73] W. Romaszkan et al., "ACOUSTIC: Accelerating convolutional neural networks through or-unipolar skipped stochastic computing," in *Proc. Design, Autom. Test Eur. Conf. Exhibit. (DATE)*, Mar. 2020, pp. 768–773.

[74] Z. Li et al., "HEIF: Highly efficient stochastic computing-based inference framework for deep neural networks," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 38, no. 8, pp. 1543–1556, Aug. 2019.

[75] H. Qin et al., "Binary neural networks: A survey," *Pattern Recognit.*, vol. 105, Sep. 2020, Art. no. 107281.

[76] Y. Wang et al., "Universal approximation property and equivalence of stochastic computing-based neural networks and binary neural networks," in *Proc. 33rd AAAI Conf. Artif. Intell. (AAAI)*, 2019, pp. 5369–5376.

[77] H. El-Derhalli, S. Le Beux, and S. Tahar, "Design space exploration of stochastic computing architectures implemented using integrated optics," *IEEE Trans. Emerg. Topics Comput.*, early access, Jan. 27, 2020, doi: 10.1109/TETC.2020.2969435.

[78] A. Chen et al., "A survey on architecture advances enabled by emerging beyond-CMOS technologies," *IEEE Design Test. Comput.*, vol. 36, no. 3, pp. 46–68, Jun. 2019.

[79] G. I. Parisi et al., "Continual lifelong learning with neural networks: A review," *Neural Netw.*, vol. 113, pp. 54–71, May 2019.

[80] F. M. Castro et al., "End-to-end incremental learning," in *Proc. 15th Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 241–247.

[81] M. Farajtabar et al., "Orthogonal gradient descent for continual learning," in *Proc. 23rd Int. Conf. Artif. Intell. Statist.*, 2020, pp. 3762–3773.

[82] Y. Bahri et al., "Statistical mechanics of deep learning," *Annu. Rev. Condens. Matter Phys.*, vol. 11, no. 1, pp. 501–528, 2020.

[83] A. Plebe and G. Grasso, "The unbearable shallow understanding of deep learning," *Minds Mach.*, vol. 29, no. 4, pp. 515–553, Dec. 2019.

[84] T. J. Sejnowski, "The unreasonable effectiveness of deep learning in artificial intelligence," *Proc. Nat. Acad. Sci.*, Jan. 2020, Art. no. 201907373.

[85] Z. Allen-Zhu, Y. Li, and Z. Song, "A convergence theory for deep learning via over-parameterization," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 242–252.

[86] N. Srivastava et al., “Dropout: A simple way to prevent neural networks from overfitting,” *J. Mach. Learn. Res.*, vol. 15, no. 56, pp. 1929–1958, 2014.

[87] C. Blundell et al., “Weight uncertainty in neural networks,” 2015, *arXiv:1505.05424*. [Online]. Available: http://arxiv.org/abs/1505.05424

[88] L. Valentin Jospin et al., “Hands-on Bayesian neural networks—A tutorial for deep learning users,” 2020, *arXiv:2007.06823*. [Online]. Available: http://arxiv.org/abs/2007.06823

[89] T. P. Lillicrap et al., “Backpropagation and the brain,” *Nature Rev. Neurosci.*, vol. 21, no. 6, pp. 335–346, 2020.

[90] X. Fan and H. Markram, “A brief history of simulation neuroscience,” *Frontiers Neuroinf.*, vol. 13, p. 32, May 2019.

[91] U. Rueckert, “Update on brain-inspired systems,” in *NANO-CHIPS 2030: On-Chip AI for Efficient Data-Driven World*, B. Murmann and B. Hoefflinger, Eds. Cham, Switzerland: Springer, 2020, pp. 387–403.

[92] B. R. Gaines, “The learning curves underlying convergence,” *Technological Forecasting Social Change*, vol. 57, nos. 1–2, pp. 7–34, Jan. 1998.

[93] T. Hospedales et al., “Meta-learning in neural networks: A survey,” 2020, *arXiv:2004.05439*. [Online]. Available: http://arxiv.org/abs/2004.05439

[94] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[95] M. Bojarski et al., “End to end learning for self-driving cars,” 2016, *arXiv:1604.07316*. [Online]. Available: https://arxiv.org/abs/1604.07316

[96] D. Silver et al., “Mastering the game of go without human knowledge,” *Nature*, vol. 550, no. 7676, pp. 354–359, Oct. 2017.

Brian R. Gaines retired in 1999 and is now an Emeritus Professor. He is also an Adjunct Professor with the Department of Computer Science, University of Victoria, Victoria, BC, Canada. His current research interests are in the foundations of logic, the role of knowledge and scholarship in human civilization, technological forecasting, system theory, human–computer interaction, artificial intelligence, and computer architectures.

■ Direct questions and comments about this article to Brian R. Gaines, Department of Computer Science, University of Victoria, Victoria, BC, Canada and University of Calgary, Calgary, AB, Canada; gaines@uvic.ca, gaines@ucalgary.ca.