Chapter 6
Nonlinear Computing and Nonlinear Artificial Intelligence

Behnam Kia and William Ditto(✉)

Nonlinear Artificial Intelligence Lab, North Carolina State University, Raleigh, NC, USA
bkia@ncsu.edu, wditto@ncsu.edu

Abstract. The importance and the necessity of nonlinearity in Artificial Intelligence, AI, and deep learning are very well understood. A multi-layer neural network with linear activation function is equivalent to a single layer of neurons. It is nonlinearity of activation functions that adds complexity to each layer, transforming the network to a universal computing machine that can approximate any continuous function. However, nonlinearity and the complexity that it creates have not been investigated enough in AI and modern deep learning systems. NC State University's Nonlinear Artificial Intelligence Lab focuses on nonlinearity and the complexity that comes with it, and investigates how this can be an engine of artificial intelligence. We peruse our research at different levels with different goals. In this article we explain our approach, and present an overview of our results.

6.1 Introduction

We live in a nondeterministic, noisy, and stochastic world. Furthermore, it is believed that noise, stochasticity, and chaos play a crucial role in our brain and the way it processes information [1–3]

Transistors are the basic computer systems. The main approach to improve the performance of the computers has been following the Moore's law -scaling down the size of transistors and integrating more transistors into a computer chip [4]. The Moore's law has provided us with a roadmap to improve the performance of the computers for decades. But the challenge is that after decades of scaling the transistors, we have reached to a point that as we further scale down the size of transistors, we are reaching fundamental physical limitations of these devices, and we are losing the determinism of these binary switches. For example, electrons can tunnel through an open switch (quantum tunneling) [5]. And it is becoming exponentially harder and more expensive to design and fabricate fully deterministic systems that perform deterministic computing. On top of it, we are moving towards stochastic processing and computing,

V. In et al. (Eds.): *Proceedings of the 5th International Conference on Applications in Nonlinear Dynamics*, Understanding Complex Systems, https://doi.org/10.1007/978-3-030-10892-2_6

and the most notable example is AI. So why not utilize and embrace nonlinear, chaos-based hardware, and use it to perform computing methods that are inherently robust to noise? This is the approach that we have picked, and we design and fabricate nonlinear, chaotic hardware, and we utilize this platform to implement nondeterministic computation and AI. However, there is a lot of challenges facing adoption and utilization of nonlinear dynamics and chaos in artificial intelligence.

Adopting and engineering chaos and nonlinear dynamics into an engineering application is a two-edged sword. From one perspective, we can enjoy the great amount of processing power that chaos can deliver. For example, it is shown that a simple nonlinear circuit can represent an infinite number of different functions. On the other hand, chaos comes at a great cost too. Designing a robust, stable nonlinear, chaotic circuit, and manually or adaptively programing it to implement desired tasks is not a simple job, and furthermore, noise and fabrication nonidealities can degenerate the performance of the circuit.

There is a lot to learn from the story of deep learning. Deep neural networks—neural networks with multiple hidden layers—were very well known to researchers and machine learning practitioners, and their great performance as universal function approximators was very well understood. But they were deemed unpractical because when the nonlinear operations of multiple layers of neurons are composed together, the training of resulting function is mathematically intractable. In other words, training a multilayer deep neural network is a non-convex optimization problem to solve [6]. As a result, many abounded the idea of deep neural network in favor of simpler, but less powerful, machine learning methods such as Support Vector Machines (SVM) that are mathematically tractable [7]. But expressing the learning mechanism as a non-convex optimization problem brings immense representation, modeling, and learning power. In 2012, with the help of GPUs and large data sets for training, finally a practical method was introduced to optimize these non-convex learning problems, and after that AI never became the same [8]. The main take-home note from deep learning story is that if we manage to tame very complex nonlinear systems, we can unshackle the unprecedented high-performance that these complex systems can provide. This has been our mission in our research group from day one. Take a chaotic system that brings the maximum possible amount of diversity in behavior and complexity, tame it and utilize the performance that it can offer. In [9] we demonstrated that a simple nonlinear circuit contains an infinite number of different functions. In [10] we introduced nonlinear dynamics as an engine of computing.

In Sect. 6.2 we explain the main idea behind how we can utilize chaos and nonlinear dynamics in computation. In Sect. 6.3 we will overview our recent nonlinear hardware designs. And describe what type of processing we can perform on top of this hardware. In Sect. 6.4 we review sample applications that we have implemented. In Sect. 6.5 we discuss where our designs fit in the industry, how much compatible they are with exiting technology, and we conclude the article.

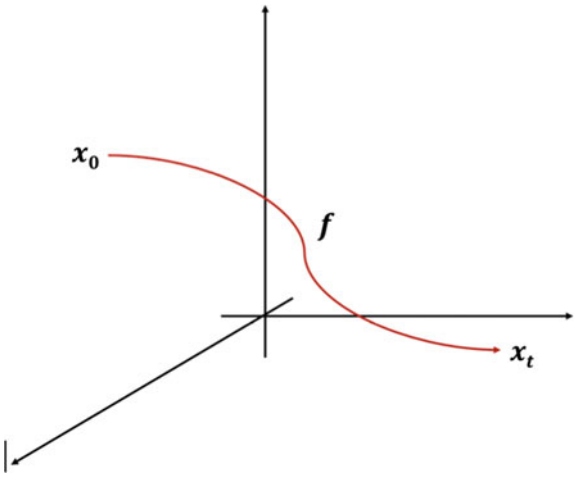

Fig. 6.1. A 3-dimensional dynamical system that maps an initial state x_0 to a future state x_t. Obviously, the dynamical system can be considered as a function

6.2 The Main Idea

A dynamical system is a system that evolves over time and maps states in its state space to some other future states. Let f be a dynamical equation, mapping initial states to future states:

$$f : \Re^n \rightarrow \Re^n \tag{6.1}$$

Figure 6.1 shows an example visualization of a dynamical system in 3-dimensional state space, $n = 3$.

It is clear from the definition and visualization of dynamical system that a dynamical system embodies a function, it implements a function.

A dynamical system can be linear or nonlinear. A linear dynamical system tends to build a simple, basic function, whereas a nonlinear dynamical system can implement much more complex functions. Much more importantly, a nonlinear dynamical system usually happens to be sensitive to its parameters. This provides us with a parametric function builder that given different parameters can implement different functions. See Fig. 6.2 where a parametric nonlinear dynamical system f_p is implementing two different functions for two different p values.

It is shown that indeed a nonlinear dynamical system contains an infinite number of functions [9], and nonlinear dynamics can be considered as an engine of computation [10]. In [10] it is shown that the number of distinguishable functions

$$N_f \propto e^{\lambda_C n} \tag{6.2}$$

increases exponentially with evolution time, where λ_C is the computing exponent, and n is the number of iterations the iterative dynamical system makes

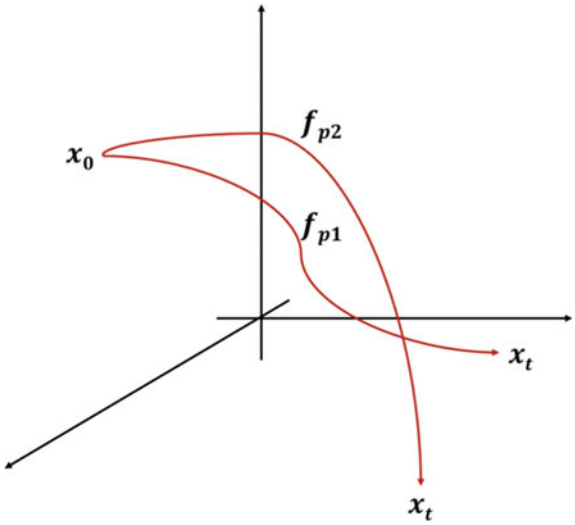

Fig. 6.2. A 3-dimensional parametric dynamical system that maps an initial state x_0 to two different future state x_t. With different parameter values one can potentially implement different functions

before producing the final state (or in continuous-time dynamical systems we will have evolution time t instead of iteration number n). The computing exponent λ_C was defined in parallel to Lyapunov exponent, with this difference that computing exponent measures and captures the number of different functions that a dynamical system can implement. Nonlinear dynamical system can have positive computing exponent, therefore the number of functions that they can implement exponentially increases as the iteration number n (or evolution time t) linearly increases. This demonstrate the capacity of the nonlinear systems in approximating and implementing different functions.

Our research has bifurcated into two avenues: first, manually finding and setting the parameters in order to program the nonlinear dynamical system to implement a desired function, and second, letting the nonlinear dynamical system itself learns which parameters it needs to select in order to implement the desired function. In the next sections we explain these two avenues, and what type of applications we can implement.

6.3 Hardware Design

We have designed and developed multiple generations of hardware for nonlinear computing. We have followed a similar path to design and develop nonlinear dynamics-based hardware that is simple in design, while complex in behavior. Such nonlinear hardware can implement complex and diverse tasks and functions using fewer transistors and less energy [11]. And they create an ideal hardware

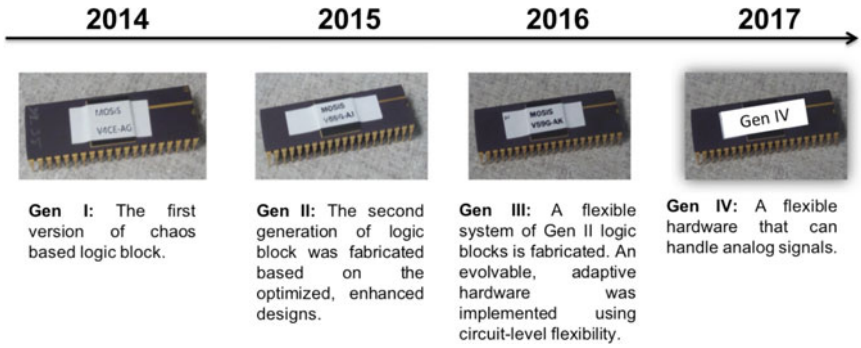

Fig. 6.3. Four generations of hardware developed by Nonlinear Artificial Intelligence Lab

platform to implement nonlinear computation. Currently we have designed and developed four generations of nonlinear dynamics-based hardware, and with each generation we have advanced both the hardware as well as the applications that it can enable and implement (Fig. 6.3).

It is important to note that this is a *technology platform* in the sense that many different applications can be designed and deployed. Figure 6.4 shows a model for our technology platform.

Device Level: We use conventional CMOS devices to design our circuits and we use conventional CMOS technology to fabricate our circuits and chips. Our hardware technology is a new design method that makes use of current devices in order to design nonlinear circuits that exhibit very complex behaviors.

It is worth noting that *beyond CMOS* devices can also be used to design nonlinear circuits. As an example, memristors can be suitable nonlinear devices to implement nonlinearity and complex behavior at the circuit level. However, for practical reasons at this point, we are mostly focused on conventional CMOS devices as the building blocks of our circuits.

Circuit Level: At the circuit layer, we design circuits that have nonlinear, complex behavior. This circuit design is nothing more that connecting a series of basic CMOS devices together, but with the crucial difference that we purposefully create nonlinearity and complexity in behavior, and thus derive complex processing out of this complex behavior. This is a philosophical and engineering departure from the conventional norm. In conventional design methods, designers make sure that all of their circuits have simple, fully predictable, stable dynamics. And then they put together many of these simple circuits in order to implement complex systems. In other words, complexity is achieved through a complex design with many devices and circuits. But in our approach, we develop simple-in-design, but complex-in-behavior, circuits and systems. Therefore, complex processing emerges from the complex dynamics of simple circuits that have fewer transistors and lower energy requirements.

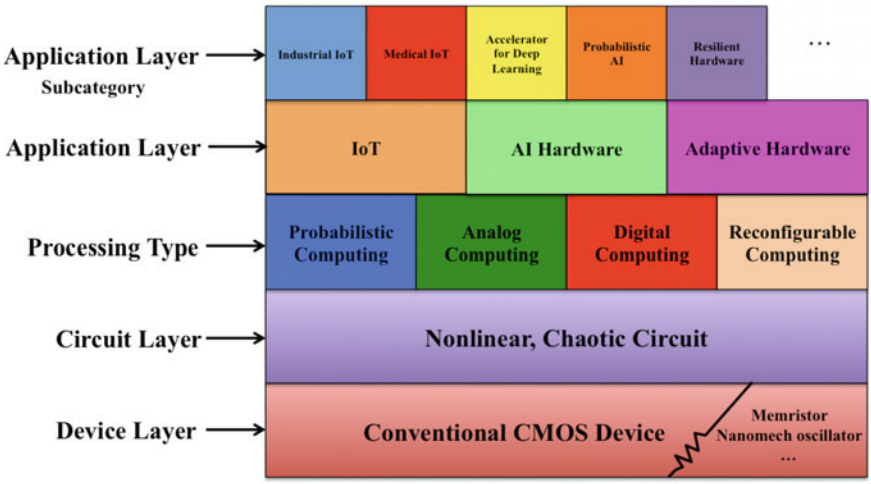

Fig. 6.4. Model of Nonlinear Artificial Intelligence's technology platform, showing its different layers of design

Processing Type: The main processing capabilities of this hardware emerge from its complex dynamics, and since complex dynamics is flexible in behavior, the nonlinear circuit can implement many different functions and tasks. More specifically, we have shown that the hardware can implement all of the following types of processing:

- Digital Computing: The circuits can emulate operations of different digital functions.
- Reconfigurable computing: Complex dynamics is flexible and contains many different behaviors; therefore it can emulate many different functions. And reconfiguration is instant since they all coexist within the same circuit as opposed to FPGAs, which require halting the processing and loading new control bits.
- Probabilistic Computing: The complex dynamics of the nonlinear circuits can operate as a probabilistic system and therefore can perform probabilistic computing.
- Analog computing: These nonlinear circuits are analog in nature, and they can receive and process both analog and digital inputs.

Application Layer: This hardware is a platform with all of the unique processing capabilities listed above, so many different applications can be designed and developed based on it. The Fig. 6.2 model shows some of these applications. These applications are enabled by one or more processing capabilities in the processing layer. We have designed different proof-of-concept examples to demonstrate the processing capabilities and possible applications the hardware can perform. Some of these examples are listed below.

6.4 Example Applications

In introduction we mentioned that a nonlinear chaotic system contains many different functions. Basically, what this means is that a chaotic system embodies many different functions that are selectable. This provides us with a platform for representation; representation of different functions or behaviors. We can take two different approaches to utilize this rich library of functions, (1) manually pick and choose them, and (2) let the system learn to pick and choose automatically. We first started from the manual selection, where the designer/programmer picks and choose it by direct coding. The result was an ALU unit.

6.4.1 Adaptive Hardware

Since our new hardware is flexible and programmable, it can adapt to different internal or external changes, and also adapt to its changing environment. This adaptation can be manually administrated, or it can be autonomous. For example, we purposefully overheated one of our fabricated hardware to a level (82 °C) well beyond its specification and tolerance level. As a result, it eventually failed to do what it is was programmed to do. However, because the hardware was flexible, we reprogrammed with a new set of control inputs to perform the same task, albeit using different control inputs [12] (Fig. 6.5).

6.4.2 Learning and Artificial Intelligence

By utilizing nonlinear dynamics, living systems exhibit diverse and complex behaviors while conserving their energy. And they can explore many different behaviors or reactions that their nonlinearity provides to them in order to (adaptively) pick and choose the ones that best meet their needs and conditions at the time. We explore such connections, and design and build intelligent hardware based on this concept. Our main hypotheses toward achieving artificial intelligence with morphable nonlinear systems are that: (1) nonlinear dynamics provides flexibility and morphability, and therefore it creates a suitable platform for

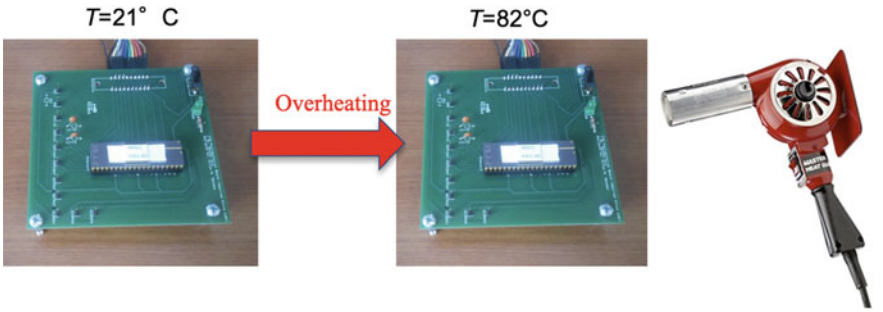

Fig. 6.5. An adaptive hardware maintaining operational capability despite external and internal changes (overheating in this specific experiment)

plasticity and learning (or intelligence in general); and (2) machine intelligence should be hardware based, as opposed to being software based. In nature there is no separate software; it is the physical organism itself that shows intelligence, and that intelligence is intertwined with the inherited genetics and physical make up of the organism. Combining these two hypotheses, we propose that to achieve nature-like intelligence, we need a nonlinear dynamics-based hardware that provides flexibility and plasticity at the hardware level. We have trained one of our fabricated hardware chips to evolve and learn different tasks, such as summation or subtraction, with no need for direct programing. The problem of automatically training a chaotic system to implement a given function can be formulated as an optimization problem below:

$$p_f = \underset{p}{argmin} \sum_i cost(x_i, y_i, \hat{y}_i) \tag{6.3}$$

where p is parameter of the chaotic system, x_i,y_i is a pair of input-output that the chaotic system is supposed to learn how to map (such pairs of given inputs-outputs are called training data in the context of AI; the data drawn from a desired function that maps x to y, and we use this training data to tune the parameters of chaotic system to implement the desired function), $\hat{y}_i$ is what chaotic system produces as the output to x_i, cost function can be defined as squared error if the outputs are continuous valued, or as binary hit/miss if the outputs are binary, i.e. $cost(x_i, y_i, \hat{y}_i) = 0 \quad \text{if} \quad y_i = \hat{y}_i, \text{otherwise} \quad 1$, and we calculate cost function over the entire training data (all i values). Now the problem of learning a desired function using a chaotic system is transformed to an optimization problem where we reduce the distance between y_i,$\hat{y}_i$ for all i values, and different optimization techniques can be used to minimize this cost function. The results of this experiment are under review to be published as a separate research article.

6.4.3 IoT Hardware

This application is a mixture from the examples above. We are introducing hardware for IoT nodes, where there is a massive influx of sensor data, and this data is filtered and processed to extract information to be sent to the higher layers of an IoT network. Figure 6.6 below shows the conventional general data acquisition and processing signal chain for IoT nodes and edge computing.

Our new hardware can implement the IoT node and computing at the node (edge computing) with a much more efficient chain shown in Fig. 6.7 below:

Our nonlinear dynamics-based hardware can:

- Directly receive analog inputs from sensors;
- Filter noise from analog signals;
- Convert analog signals to digital;
- Digitally process these digital inputs;
- Morph into new configurations at any cycle, and therefore digital processing can be reconfigurable, adaptive, and evolvable;

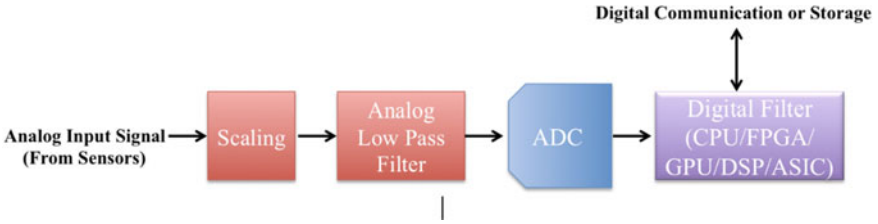

Fig. 6.6. Conventional data acquisition and processing signal chain

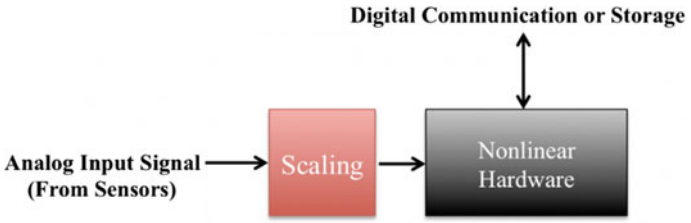

Fig. 6.7. Alternative chain, enabled by our hardware

- Implement many different operations including multiplications efficiently, which means it can implement multiplication-intensive applications such as deep learning with minimal power and silicon area requirements.

6.5 Conclusion

A chaotic system is hard to work with, it scares the engineers away, it is unstable, hard to design, fabricate, and utilize. But if all is done correctly, a chaotic system provides an unprecedented amount of performance, unmatched by any conventional linear system. The AI community has fully experienced this transformation of moving from tractable, elegant methods and mathematics to intractable, hard to optimize models, and this move resulted in huge leap in AI. We believe chaos is another uncharted territory that despite the challenges that come with it, can provide huge rewards.

Here we discussed our fabrications, sample applications, and the results. The main conclusion is that chaos can provide extremely fascinating features and capabilities with unique applications, however, there are challenges to overcome. NAIL has been following multiple different tracks to AI. On one extreme, we teach and practice the conventional AI and deep learning and team with government, research and technology companies to apply conventional AI to their needs. On the other extreme, NAIL is pioneering a novel approach to AI based on nonlinear dynamics and chaos to develop AI systems that demonstrate awareness, cognition and deeper intelligence and interactions.

References

1. T.M. McKenna, T.A. McMullen, M.F. Shlesinger, The brain as a dynamic physical system. Neuroscience **60**(3), 587–605 (1994)
2. M.D. Fox, A.Z. Snyder, J.L. Vincent, M. Corbetta, D.C. Van Essen, M.E. Raichle, The human brain is intrinsically organized into dynamic, anticorrelated functional networks. Proc. Natl. Acad. Sci. U.S.A. **102**, 9673-9678 (2005)
3. R.T. Canolty, M. Soltani, S.S. Dalal, E. Edwards, N.F. Dronkers, S.S. Nagarajan et al., Spatiotemporal dynamics of word processing in the human brain. Front. Neurosci. **1**, 185-196 (2007)
4. Chris A. Mack, Fifty years of Moore's law. IEEE Trans. Semicond. Manuf. **24**(2), 202–207 (2011)
5. Thomas N. Theis, H.-S. Philip Wong, The end of Moore's law: a new beginning for information technology. Comput. Sci. Eng. **19**(2), 41–50 (2017)
6. A. Blum, R.L. Rivest, Training a 3-node neural network is NP-complete. Advances in neural information processing systems (1989)
7. Marti A. Hearst et al., Support vector machines. IEEE Intell. Syst. Appl. **13**(4), 18–28 (1998)
8. A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems (2012)
9. B. Kia, J.F. Lindner, W.L. Ditto, A simple nonlinear circuit contains an infinite number of functions. IEEE Trans. Circuits Syst. II: Express Briefs **63**(10), 944–948 (2016)
10. Behnam Kia, John F. Lindner, William L. Ditto, Nonlinear dynamics as an engine of computation. Philos. Trans. R. Soc. A **375**(2088), 20160222 (2017)
11. B. Kia, K. Mobley, W.L. Ditto, An integrated circuit design for a dynamics-based reconfigurable logic block. IEEE Trans. Circuits Syst. II: Express Briefs (2017)
12. B. Kia et al., Nonlinear dynamics-based adaptive hardware, in *2017 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)* (IEEE, 2017)