Casting off the Old Guard:
Achieving Superior A.I. Performance through Simplification

Jerry Felix, Steve Brunker, Carol Hibbard
Brain-CA Technologies, Inc.

Training and deployment of huge machine learning models requires a vast amount of compute resources, and still dramatically underperforms the biological brain, which operates with six orders of magnitude less energy consumption. The trend of the conventional approach is toward more complicated engineered systems. At Brain-CA Technologies, we have designed a new architecture based on Cellular Automata. This simplified architecture casts off the old guard technologies, eliminating many of their inherent restrictions. Our projections show that this new architecture provides significant improvement in energy consumption, while performing AI/ML tasks equivalently or better than existing systems.

The longevity of John von Neumann's 1945 computer architecture is evidence of its brilliance. But it was designed from the outset to solve math problems, not to perform learning tasks.[1] A few years after his ground-breaking paper defining the architecture, Von Neumann experimented with a branch of mathematics called Cellular Automata (CA). Von Neumann recognized that with CA, extremely complex patterns can emerge from systems of autonomous cells, each operating independently using the same simple set of rules.[2] He, and many others, speculated that CA is a fundamental technology underlying biological systems including intelligence and the biological brain. Brain-CA Technologies, Inc. has designed a patented learning system from the ground up, using cellular automata, that eliminates the Von Neumann bottleneck.[3] Our innovative technology paves the way for more powerful AI systems which consume much less energy, while increasing portability and scalability.

The New Guard for Artificial Intelligence	
Tried and Tested	**Bold and Innovative**
Von Neumann architecture: separate memory and compute	BRAIN-CA™ Cellular Automata architecture: embedded memory with simple compute (logic)
Neural networks: weight and bias calculations, back propagation	Binary decomposition: simple correlation table lookups, The Cincinnati Algorithm
Highly complex, hot systems: focus on maximizing FLOPS	Simple systems with lower energy costs: focus on minimizing CA System complexity

[1] https://web.mit.edu/STS.035/www/PDFs/edvac.pdf
[2] Bhattacharya, Ananyo (2022). The Man from the Future: The Visionary Life of John von Neumann. W. W. Norton & Company. ISBN 978-1324003991.
[3] https://www.techtarget.com/whatis/definition/von-Neumann-bottleneck

The New Guard for Artificial Intelligence	
Tried and Tested	**Bold and Innovative**
Learning through complex calculus	Learning by associating observations
Absolute memory addressing	Relative cell addressing
Rigid node connections	Wave propagation and dynamic connections
Floating point representation	Bit, integer, and BRAIN-CA™ Estimator representation
Floating point math operations (add, subtract, multiply, divide, exponentiate, log, square root, trigonometric, etc.)	Logic only (compare, invert, And, Or, Not, increment, decrement)

Fundamentals of Cellular Automata (CA)

The CA concept was originally studied in the 1940s by Stanislaw Ulam and John von Neumann. A CA system consists of a regular grid of cells. Each cell maintains a state value (or state vector), and all cells independently update their states based on their current state and the states of neighboring cells, applying a simple set of rules to perform the update.[4]

In the 1970s, John Conway created a CA, "Conway's Game of Life" (CGoL), which defined state values as binary (alive or dead), and a simple ruleset for generation, continuance, and termination of life. Cells can communicate information over great distances using various patterns. One such pattern, called a glider, appears to move across the grid over time.[5] CGoL has been shown to theoretically be able to emulate a universal Turing Machine,[6] but has heretofore been proven to be impractical to implement.

The BRAIN-CA™ Approach

Our strategy was to start with a simple CA design, like CGoL, and add *purposeful complexity.* We aimed for the sweet spot between CGoL's simplicity and a GPU's complexity,[7] to create a system that is designed from the outset to perform AI tasks. Our solution slightly *increases* the complexity of each cell beyond CGoL, but dramatically *reduces* the complexity of the overall system necessary to perform AI tasks such as learning, prediction and inference, as compared to today's AI architectures. Every tiny bit

[4] Wolfram, Stephen (1983). "Statistical Mechanics of Cellular Automata". Reviews of Modern Physics. **55** (3): 601–644. Bibcode:1983RvMP...55..601W. doi:10.1103/RevModPhys.55.601

[5] https://www.ibiblio.org/lifepatterns/october1970.html

[6] https://web.archive.org/web/20090906014935/http://www.igblan.free-online.co.uk/igblan/ca/

[7] Jensen Huang, CEO, Nvidia, in a Wired Magazine interview: "The operating system of these large systems is insanely complex. How do you create an operating system in a computing stack that orchestrates the tens of millions, hundreds of millions, and now coming up to billions of little tiny processors that are in our GPUs? That's a very hard problem.", https://www.wired.com/story/nvidia-hardware-is-eating-the-world-jensen-huang/

of added complexity is magnified when replicated millions or billions of times in a CA grid, which is why each added bit of complexity must be purposeful.

Complexity of a cell can be measured by the number of transistors or logic gates necessary to implement it, while the complexity of the overall CA system is a function of both the number of transistors (or logic gates) in the system and the time necessary to achieve the overall goal of learning and predicting. As an example, CGoL has low complexity cells, but implementing AI on it would be highly complex, as CGoL would take a very long time to learn and predict.

A notable feature of the BRAIN-CA™ architecture is that the CA system has no master controller - no central processing unit (CPU or GPU). Typical implementations of CA using Von Neumann Architecture utilize memory to store an array of cells and a CPU to execute the logic of the CA rules to transform each element of the array from one state to the next. In the BRAIN-CA™ system, each cell is responsible for, and capable of updating itself. This is a *different* way of performing AI, a way that was designed for *both* learning and inference from the ground up, *using a unified architecture*.

Without a master controller, each cell must have enough complexity to store its state vector and the logic necessary to update its state over time. The challenge that we undertook was to design a cell with minimal complexity that is capable of learning in its native implementation, such that the overall CA system complexity is minimized. Utilizing this approach forced us to reject methods that have been extremely successful for the past eighty years, and that are common in today's AI processors.

The cellular automata concept can be observed in nature: ants and bees operate under simple sets of rules, and somehow, very complex structures (ant colonies and bee hives) and societies emerge. It's also similar to the biological brain: there aren't separate units for computation and memory; the brain *is* memory. Intelligence emerges from cells acting autonomously and establishing associations with other cells.

To learn, AI systems must find patterns in the streams of data that arrive concurrently, meaning that concurrent data must be brought together somehow. The task of comparing millions of inputs to each other, whether performed with hardwiring or in software, using traditional methods, scales at $O(n^2)$ or worse. Exponential growth in model size is causing today's technology to bump against physical limitations.

Overview of the BRAIN-CA™ Learning Strategy

The BRAIN-CA™ System looks for correlations between pairs of data streams. Training data is fed into the CA system in a binary representation, with each bit of the training data fed to a different cell of the CA, spread out sparsely within a vast cellular grid. The CA updates its state repetitively with a ruleset that causes the binary training values to ripple outward from their respective origins. These ripples collide with other ripples across the grid, and at the point of collision a measurement can be taken as to the bit pattern: Are they both positive? Both negative? First positive and second negative? Vice versa?

Over time, correlation patterns between pairs of data streams develop at the collision points. The system detects data streams which are correlated to other streams that happen concurrently, or slightly earlier, based purely on these historical observations. It's important to note, however, that while we just described "the system" detecting the pattern, in fact it is an individual collision cell, acting autonomously, that makes that determination. No external controller is required.

When an individual cell observes a recurring pattern between two other cells, it initiates a process to interconnect those two cells with a prediction accelerator. The collision cell tells two of its neighbors to build a connection, and those two cells set persistent values in their state vector. At the next CA generation (clock tick), those two cells tell their neighbors to do the same, until the prediction accelerator path joins the two original cells, to leverage the predictive capability that one cell has over the other.

Once the connection path is in place, the system can rapidly apply what it has learned. Test data arriving at a predictive pulse point can travel *immediately* down the prediction accelerator path to the correlation-tracking cell, which can make a prediction of the value of the other pulse cell, using historic statistics.

Analogies with Nature

While the BRAIN-CA™ system was designed as a learning system first, and not initially modeled after the biological brain, certain similarities have become apparent. The grid of cells in the BRAIN-CA™ system is analogous to a plane of cells in the biological brain. Signals are received from sensory organs (eyes, ears, finger tips, etc.), which are connected to various locations in the brain. In the same way, signals are fed into the various positions of the sparse BRAIN-CA™ cellular grid. Plasticity is a feature of both.

Waves are an efficient and naturally occurring communication phenomenon. They permit multidimensional information exchange in our system with minimal interference - similar to sound or brain waves. Using waves permits pattern recognition without the rigidness of connecting each pair of data stream sources. It allows exploration of correlations, such that accelerated prediction paths are only built when a pattern is found. And if a correlation proves to be spurious, the path can be easily removed. The Hebbian Theory ("Neurons that fire together wire together")[8] is analogous to the construction of accelerated prediction paths in the BRAIN-CA™ system.

Waves also provide an exceptional way to scale. An analogy is communication within a stadium of 100,000 fans: effective communication can be conducted via sound waves over a loudspeaker, rather than 100,000 individual connections. Note, too, that the fans can use waves to communicate among themselves simultaneously.

It shouldn't be surprising that an extremely simple system, modified to add purposeful complexity, is similar to biological systems that evolved over millions of years through Darwinian evolution.[9] Generally speaking, one would expect simple working solutions to evolve and propagate sooner in nature than complicated solutions.[10]

The Anatomy of a BRAIN-CA™ Cell

Each cell in the BRAIN-CA™ system is designed to natively perform the following functions, *without a CPU*: (1) Propagate signals from incoming training data, (2) detect collisions of two binary streams, (3) determine whether a collision should be tracked, (4) track which of the four possible combinations of two

---

[8] Hebb, D.O. (1961). "Distinctive features of learning in the higher animal". In J. F. Delafresnaye (ed.). Brain Mechanisms and Learning. London: Oxford University Press.
[9] Wallace, Alfred Russel. (1889). Darwinism: An Exposition of the Theory of Natural Selection, with Some of Its Applications. Macmillan and Company.
[10] https://www.scientificamerican.com/article/the-surprising-origins-of-evolutionary-complexity/

binary values occurred, (5) find correlations between pairs of data streams based on historical data, (6) initiate the construction (or removal) of an accelerated prediction path (APP), (7) set internal state values to represent APPs, (8) transmit incoming data along the APPs, (9) use incoming APP pulses to make a prediction (potentially combining it with a prediction provided by a neighbor cell), (10) buffer predictions as needed, and transmit predictions to the appropriate cell at the appropriate time.

We set out to design a cell to be as simple as possible, and perform these ten functions. We found creative ways to perform these functions without a CPU or a GPU, without requiring circuitry to perform math functions that are performed in typical AI systems, and without any floating point operations. Our target is 5000 transistors per cell, which rules out all of those conventional technologies.[11]

Better Efficiency from Appropriate Precision for AI

Around 1800, Carl Friedrich Gauss studied the concept of significant figures, which is taught in high school chemistry classes to this day. Gauss noted that measurements often lack precision (such as eyeballing a ruler in 1800), and that computation beyond the number of significant figures is a waste of valuable time and energy. Performing complex calculations by hand, involving many digits beyond the decimal point was a waste.

Nearly all modern CPUs and GPUs utilize the IEEE 754 floating point numeric representation defined in 1985 (and last modified in 2008).[12] Common floating point representations today perform operations with either 7.22 or 15.95 decimal digits of precision, depending on the programmers choice of representation and the architecture used.[13] Performing calculations with a high level of precision reduces the amount of round-off error caused by intermediate calculations. This standard has performed well for the scientific community for decades.

Standardizing on a common numeric representation allowed engineers to optimize logic design and algorithms to work with these numbers. In short, it's currently the only game in town.

But Gauss had a good point. Colloquially speaking, why calculate the "precise location of your seat" when you aren't even "in the right ballpark"? Few options beyond IEEE 754 floating point representation are available today. A system to implement significant figures on today's technology would be *more* complex - it would perform floating point calculations and then add overhead to reduce the significance.

We estimate that a substantial majority of the energy expended performing floating point calculations on modern AI training is wasted by performing calculations at unnecessary precision levels. This is a huge opportunity for energy savings, but it requires redesign from the ground up.

---

[11] An extremely simple CPU, such as the Intel 8008 processor (circa 1972) has about 3500 transistors, and is not capable of division. A CPU with the capability of integer division is the Intel 8080 processor (circa 1978) with about 6000 transistors. And Intel's first floating point co-processor had about 40,000 transistors. https://www.intel.com/content/dam/www/public/us/en/documents/case-studies/floating-point-case-study.pdf

[12] Goldberg, David (March 1991). "What Every Computer Scientist Should Know About Floating-Point Arithmetic" (PDF)

[13] "IEEE Standard for Floating-Point Arithmetic," in IEEE Std 754-2008 , vol., no., pp.1-70, 29 Aug. 2008, doi: 10.1109/IEEESTD.2008.4610935.

We patented a numeric representation optimized for AI called the BRAIN-CA™ Estimator[14]. It allows accuracy and precision to work hand in hand, as relationships are measured and patterns are recognized over time. Calculations are performed with simple bit manipulations, not complex math.

We found with the Estimator that by simply pairing each storage bit with a random bit, we are able to converge on mathematically accurate ratios without performing traditional math. Through simple bit manipulations - comparisons and inversions - we're able to implement an AI system with much lower energy costs. Even the random bits needed for the algorithm can be generated using electronic noise.[15]

Furthermore, we've defined a simple algorithm, that we've named "The Cincinnati Algorithm", to modify the Estimator's value as our AI system processes training data.[16] Our model is optimized for integrated learning and inference and is able to evolve rapidly as information streams in, and it is able to adapt more quickly to changes in underlying parameters than traditional methods. See our other publication for details on the Estimator and The Cincinnati Algorithm.[17]

The Cincinnati Algorithm

1. Initialize the Estimator as a null bit string.
2. Pair each bit of the Estimator with a random bit.
3. Observe a binary value (0 or 1, Red or Green, etc.).
4. Starting at the high order bit, find an Estimator bit that doesn't match its random bit.
5. If all match, then extend the Estimator with the observed bit.
6. Otherwise, if the unmatched random bit matches the observed bit, invert the low order Estimator bits until one is inverted to be the same as the observed bit.
7. Return to step 2.

Challenges Overcome

Designing a low-transistor-count system capable of performing these learning tasks forced us to tackle various challenges. The system is built on a hexagonal grid of cells. This allows for ripple patterns to more closely match the circular ripple patterns that were initially an inspiration, as well as facilitating connection paths which can pass through the cell at multiple angles.

The transient signals that are passed from cell to cell represent both the binary value of the input data and the relative position of the originating pulse. The system uses an attenuation method, such that the signal gets weaker the further it travels, much like in nature. The binary nature of the signal is represented by the sign, either positive or negative. This is analogous to creating ripples by, say, either dropping a pebble in a pond (creating a *valley* that ripples outward) or pulling a plunger out of water (creating a *hill* that ripples outward).

---

[14] US Patent 11,847,386 B1. https://patents.google.com/patent/US11847386B1/. Other patents pending.
[15] Stipcevic, "Non-deterministic Random Bit Generator Based on Electronics Noise", 2008, arXiv:physics/0309010v2 [physics.comp-ph]. https://arxiv.org/abs/physics/0309010.
[16] Felix, Brunker, Hibbard: "The BRAIN-CA™ Estimator and The Cincinnati Algorithm: Simplification Tools for Artificial Intelligence", 2024. https://brain-ca.com/the-brain-ca-estimator-and-the-cincinnati-algorithm/
[17] Ibid.

The signals travel outward from the originating cell to its six neighbors. At subsequent ticks of the CA clock, the signal advances away from its origin in six directions, attenuating each step of the way. Our attenuation function reduces the magnitude of the signal, which is stored as an integer. After one clock tick, the signal is at the six cells surrounding the original pulse; after second tick, the algorithm of the CA causes each of the signals to travel again in the same directions, but also branch off at oblique angles, creating a growing hexagon that surrounds the original pulse point and simulating (somewhat) the circular patterns of natural ripples such as water or electromagnetic pulses.

If the initial pulse signal is strong enough, then it will eventually reach every cell in the grid, providing an opportunity to interact with every concurrent pulse (and even pulses that are slightly offset in time). This allows correlation tracking of concurrent signals as well as signals which can predict future values of other signals due to the temporal offset.

Pulses generated by two cells will collide at numerous other cells. It is desirable to only track the correlation at one other cell on the grid. A proprietary algorithm allows a collision cell to self-determine whether it is the one and only cell to track the collision.

Predictions can be made for future values of pulses. They are queued, and combined with other predictions from other cells, if those predictions relate to the same pulse point and time. The BRAIN-CA™ ruleset routes the predictions, combining them with other predictions along the way, and the predictions arrive at the right place at the right time.

The rules also make a determination as to whether a correlation has been detected, and initiate the process of configuring a prediction accelerator. The rules set or reset persistent values representing the path for the prediction accelerator. The prediction accelerator is used in the inference phase to predict established correlations more quickly.

Another feature built into the BRAIN-CA™ system, called multi-bit technology, enables the pattern recognition of more complex patterns. By combining the relationship between two bit streams with a third bit stream to establish a new relationship, the system scales to detect complex relationships which are made up of three or more bit streams. Extremely complex patterns can be discerned. This is analogous to the way highly complex computer systems of today can be built with the clever arrangement of simple universal gates.

Designed for Simplicity

The BRAIN-CA™ system is made up of a grid of identical hexagonal cells in a tessellation pattern. Each cell is made up of six identical equilateral triangles of circuitry, each positioned at a different 60-degree angle. The challenge of creating a hardware cell capable of performing the bit manipulations to learn and predict, with a complexity of about 5000 transistors, is therefore broken down to a design challenge of about 800 transistors per equilateral triangle, which are duplicated six times.

The ruleset is fairly simple. It manipulates bits and integers. The most complicated math in the cell is bit inversion, bit comparison, and incrementing (or decrementing) an integer, which can easily be done with bit inversion and comparison, by inverting all the lower order bits until a 0 is inverted (or a 1, in the case of decrementing).

Conclusion

Brain-CA Technologies has developed a learning system using Cellular Automata technology capable of performing AI/ML tasks: creating a model through training or observation, and then applying that model through inference. We approached the challenge by starting with a simple CA system and adding purposeful complexity, to arrive at a simple cell design, capable of creating associations with other cells. The BRAIN-CA™ system can be implemented in hardware, without a master controller such as a CPU or GPU. It can replace conventional technology for AI applications, or co-exist alongside current technologies. The technology can also be simulated in software.

By designing the system with cells that are all alike, the cells can be tessellated on a grid in great quantities. The simplified architecture allows for efficient massive parallelism. We feel that it is a fundamental breakthrough to address today's AI challenges. It is a highly scalable system designed from the ground up to perform the learning tasks of pattern recognition, inference, and prediction more efficiently than has previously been possible.

About Brain-CA Technologies, Inc.

Founded by three former long-time Hewlett Packard (Enterprise) employees, Brain-CA Technologies incorporated in November 2023, and quickly secured patents on key technologies related to Artificial Intelligence.

Early in the founders' HP careers, HP was a pioneer in a technology called RISC, Reduced Instruction Set Computing. RISC technology is prevalent today, but back then, CISC (Complex Instruction Set Computing) was the norm. The trend from 1950-1985 was to *increase* the capability (and complexity) of the CPU.

Through extensive measurement, HP arrived at a design for simplified CPU chips that could out-perform the complex ones, and worked with Intel to transition the world to RISC computing. Brain-CA Technologies' efforts follow that line of thinking - keep simplifying. The simplicity of our design offers the opportunity for breakthroughs in performance through massive parallelism while decreasing energy consumption.

Authors

Jerry Felix: https://www.linkedin.com/in/jerfelix/
Steve Brunker: https://www.linkedin.com/in/stevebrunker/
Carol Hibbard: https://www.linkedin.com/in/carolhibbard/

Brain-CA Technologies

For additional information:
https://brain-ca.com/
info@brain-ca.com
513.360.8603