


The road to commercial success for neuromorphic technologies

Received: 15 November 2023

Accepted: 18 February 2025

Published online: 15 April 2025

 Check for updates

Dylan Richard Muir ^{1,2,4} & Sadique Sheikh^{1,3,4}

Neuromorphic technologies adapt biological neural principles to synthesise high-efficiency computational devices, characterised by continuous real-time operation and sparse event-based communication. After several false starts, a confluence of advances now promises widespread commercial adoption. Gradient-based training of deep spiking neural networks is now an off-the-shelf technique for building general-purpose Neuromorphic applications, with open-source tools underwritten by theoretical results. Analog and mixed-signal Neuromorphic circuit designs are being replaced by digital equivalents in newer devices, simplifying application deployment while maintaining computational benefits. Designs for in-memory computing are also approaching commercial maturity. Solving two key problems—how to program general Neuromorphic applications; and how to deploy them at scale—clears the way to commercial success of Neuromorphic processors. Ultra-low-power Neuromorphic technology will find a home in battery-powered systems, local compute for internet-of-things devices, and consumer wearables. Inspiration from uptake of tensor processors and GPUs can help the field overcome remaining hurdles.

Two main processing architectures currently dominate commercial computing: von Neumann architectures, comprising the majority of computing devices and now spread like dust in every corner of the planet; and tensor processors such as GPUs and TPUs, which have seen a rapid rise in use for computation since 2010.

Inspiration for computing systems came early from human and animal nervous systems. McCulloch-Pitts¹ proposed simplified logical units that communicate with binary values, inspired directly by activity in the nervous system. Subsequent to his state-machine model of computation which underlies modern procedural programming, Turing proposed self-modifying neurally-inspired computing systems². Von Neumann himself was fascinated by information processing in the brain³, and embarked on a research program to define principles for self-constructing computational machines (automata)⁴, predating the concept of DNA as a computational substrate.

Von Neumann highlighted the need to program a device in terms of the fundamental computational functions it can perform³. In the case of tensor processors these are matrix multiplications; in the case

of CPUs these are primarily basic arithmetic and branching operations. In the case of Neuromorphic (NM) technologies, the basic computational elements are directly inspired by biological nervous systems—temporal integration and dynamics; binary or low-bit-depth multiplication; and thresholding and binary communication between elements.

These technologies comprise a third computational architecture, in addition to CPUs and tensor processors, and several ongoing commercial and research projects are taking steps to bring NM technologies to widespread use. How can we learn from the wild success stories of tensor processors (particularly NVIDIA's GPUs) and older CPUs, as they rose to dominate the computing landscape? And how can we adapt their success to NM processors?

While the workings of our own brains are fundamentally parallel, convoluted, and remain obscure, we often use simple linear sequences of instructions when explaining a task to others. Consequently, Turing/Jacquard inspired procedural programming models came to dominate digital computing, bringing with them von

¹SynSense, Zürich, Switzerland. ²University of Western Australia, Perth, Australia. ³Present address: Unique, Zürich, Switzerland. ⁴These authors contributed equally: Dylan Richard Muir, Sadique Sheikh. ✉ e-mail: dylan.muir@synsense.ai

Neumann fetch-and-execute processing architectures (Fig. 1a). These systems provide a straightforward recipe for general-purpose computing—if you can describe step-by-step what you want a computer to do, then you can code and deploy it. This success came at the expense of analog computers, which had been highly important for scientific and war-time computations until around the middle of the 20th Century. Alternative massively-parallel multi-processor architectures fell out of favour in the 1990s, partially due to the difficulty in developing applications for these systems^{5,6}.

In the late 1990s and early 2000s the consumer graphics hardware market emerged primarily to satisfy the computational needs of home gaming. But computational scientists soon realised that the parallel vector processing cores of GPUs could be applied efficiently to data-parallel problems. Neural Network (NN) researchers found that GPUs could dramatically accelerate NN computations, permitting larger and more complex architectures to be trained faster than ever previously. Since 2010 the rise of deep neural networks has driven and been driven by a concomitant rise in tensor processor architectures, with enormous commercial success in particular for the largest GPU manufacturer NVIDIA. This shift back to parallel processing was directly enabled by the success of deep learning in providing a programming model for tensor computations – the ability to map almost any arbitrary use case to tensor computing, via a data-driven machine-learning approach⁷ (Fig. 1b). Similarly, the commercial success of NVIDIA was driven by their foresight in providing a software Application Programming Interface (API; i.e. CUDA) for their GPUs⁸, which allowed this new programming model to be mapped efficiently to their hardware. In a beneficial feedback process, availability of GPUs and increasing demand for neural network processing have driven development of both. Formerly designed to accelerate graphical loads,

more recent high-end GPUs are now explicitly designed for tensor-based computational tasks. Coupled with the provision of simple, high-level APIs embodying the deep learning programming model (i.e. Tensorflow/Keras, Pytorch), tensor processors are accessible to developers in a way that earlier multi-processing architectures were not.

In this perspective we outline the early forays of Neuromorphic hardware towards commercial use; describe the new steps the field has made in the last few years; compare and contrast the current design alternatives in nascent commercial Neuromorphic hardware; and sketch a path that we believe will lead to widespread consumer adoption of Neuromorphic technology. We focus primarily on processors making use of sparse event-driven communication with limited precision, with or without analog computing elements, and incorporating temporal dynamics as an integral aspect of computation. We include some discussion of compute-in-memory architectures, where these overlap with our focus. This covers digital implementations of spiking neurons and some analog compute-in-memory arrays, along with traditional sub-threshold silicon Neuromorphic processors.

Early steps for Neuromorphic sensing and processing

The starting line for Neuromorphic engineering lies in the 1960s in California, when the neurobiophysicist Max Delbrück buttonholed the analog circuit designer Carver Mead⁹. Intrigued by similarities between the biophysics of synapses in the nervous system and the behaviour of silicon transistors, Mead designed circuits which accurately simulated aspects of neuronal dynamics using the subthreshold analog dynamics of transistors and the physics of the silicon substrate. In the 1980s this

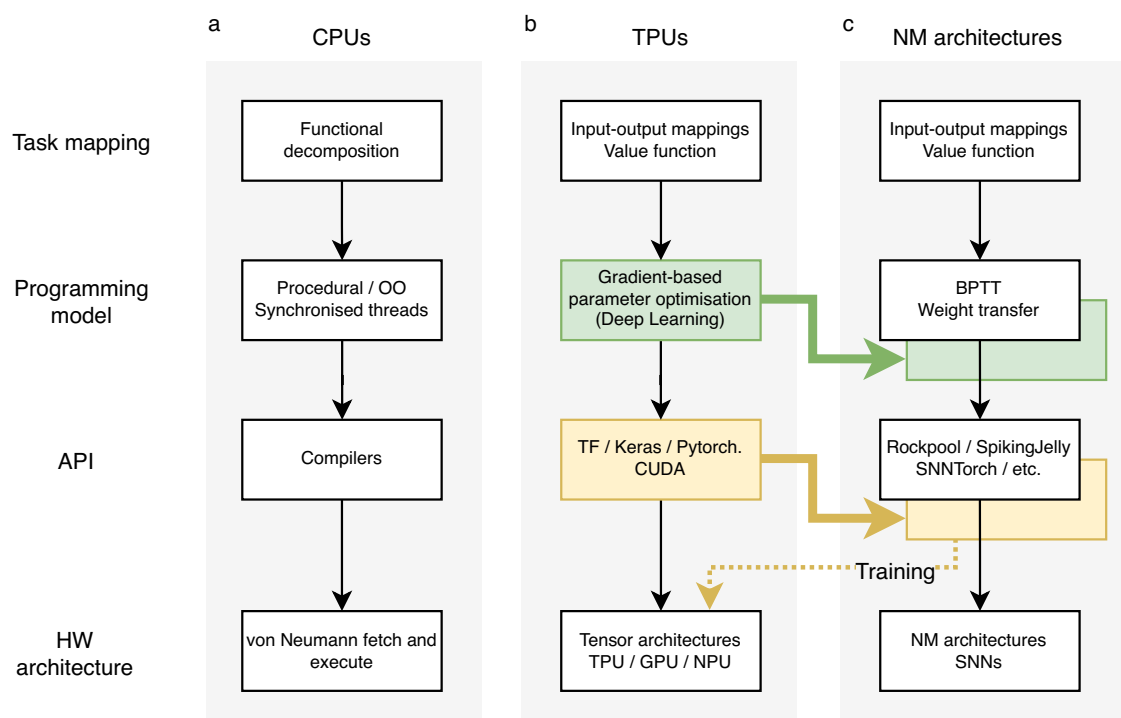


Fig. 1 | Programming models for von Neumann, tensor processor, and Neuromorphic hardware architectures. **a** von Neumann processors such as CPUs and MCUs use a number of programming models, all of which compile to a predominantly serial, fetch-and-execute approach (small-scale multi-processing and optimised pre-fetch notwithstanding). Programming these systems requires decomposing a desired task into a set of explicit procedures for a CPU to execute. **b** Tensor processors such as GPUs, TPUs and NPU have achieved great success for general-purpose applications, by adopting an example-driven, supervised machine-

learning (ML) programming model based on gradient-based parameter optimisation (green box). This is supported by APIs (yellow box) which efficiently implement the ML programming model on tensor processors. **c** Neuromorphic architectures are increasingly adopting a similar programming model, based on methods developed for deep learning (green arrow). At the same time, new software APIs for Neuromorphic application development are making efficient use of the SW tools for deep learning (yellow arrow), which permit rapid parallelised training of NM architectures on commodity tensor processors (dashed arrow).

approach crystallised into a direct examination of computation in neural-like substrates, following intense discussions between Mead, John Hopfield and Richard Feynmann¹⁰. The researchers who arranged themselves around that line of enquiry used the industrial tools of VLSI and CMOS fabrication to develop analog circuits for neurons¹¹, synapses¹², and various biological sensory organs^{13,14}. From the beginning, the commercial benefits to the NM approach were touted—to harness the extreme power efficiency of biological nervous systems, and usher in a new class of ultra-low-power computing and sensing systems¹⁵.

Early commercial success for Neuromorphic technologies came on the sensory side. Inspired by the vertebrate retina, graded analog circuits for signal processing lead to early capacitive touchpads, and movement sensors for optical mice^{16,17}. High-efficiency vision sensors were developed and commercialised into non-frame-based, low-latency cameras¹⁸, with several commercial entities investing in this approach for industrial and consumer machine vision applications as of 2024.

Similar commercial success on the computing front has taken longer. Several large industrial concerns have produced neurally-inspired event-driven processors—notably Qualcomm's Zeroth¹⁹, IBM's TrueNorth²⁰, and Intel's Loihi²¹. Samsung invested in event-based Neuromorphic vision sensing through their collaboration with startup iniVation²². However, with the exception of Intel's Loihi, and IBM's work on memristive elements for in-memory computing, these companies have moved away from Neuromorphic computational architectures in the direction of more traditional CPUs and tensor processors.

A key challenge for research into Neuromorphic computation has been uncertainty about precisely which aspects of biological neural computation are crucial. Early contributions from industry imported this uncertainty by offering high flexibility—and therefore complexity—in chip architecture.

Likewise, methods for configuring spiking neural network chips have been driven by academic research, and absent a powerful, general-purpose programming model.

Neuromorphic programming models

Until very recently, deploying an application to a spiking Neuromorphic processor required approximately one or more PhDs worth of effort. Designing an SNN application is not as simple as writing some lines of procedural code, as there is no straightforward way to compile a procedural language to the spiking neurons that comprise the computational units of a spiking Neuromorphic processor. This lack of an accessible programming model has stood in the way of NM processors as general-purpose computing solutions.

Consequently, a number of methods for configuring spiking neural networks were proposed to fill this gap²³.

Hand wiring and spike-based learning

For many applications, hand-wired networks have been used²⁴. Taking inspiration from synaptic plasticity in biological neural tissue, spike-based learning rules have been built into to neuromorphic hardware^{25–27}. However, STDP and other synapse- and neuron-local learning rules are not sufficient to produce arbitrary applications for Neuromorphic processors. Instead these learning rules are generally coupled with hand-designed networks, to solve particular restricted classes of problems.

Random architectures and dynamics

Several SNN configuration methods are based around random recurrent networks and random dynamics^{28,29}. These networks project a low-dimensional input into a high-dimensional space, and over a rich random temporal basis formed by complex dynamics of spiking networks. In theory, training these networks then becomes a linear

regression problem to tune a readout layer as desired, and these networks are universal function approximators^{28,30}. In practice, guaranteeing an expressive temporal basis, and maintaining stable-but-not-too-stable recurrent network dynamics, are extremely challenging. Due to their highly redundant architecture, random networks represent an inefficient use of resources for a given application. Computational efficiency can be improved by tuning network dynamics to follow a teacher dynamical system^{31–33}. These methods are effective, but apply best to smooth regression tasks. At the same time, the existence of a teacher dynamical system is required—which may not be feasible for some applications. Recurrent networks with engineered temporal basis functions, such as Legendre Memory Units, may offer a workable and more efficient alternative to random architectures^{34–36}.

Modular networks

Some researchers have proposed computational units, composed of spiking subnetworks, which can be combined to build larger computing systems. Such modules may be based on attractor dynamics³⁷; tuning curve based dynamical encoding³⁸; or on winner-take-all competitive networks^{39,40}. When these modules are combined in defined ways, they can be used to implement particular classes of applications such as state machines or dynamical systems. These methods could be considered as pre-cursors to large-scale programmable *functional* spiking neural networks^{41,42}.

Gradient-based machine learning optimisation

The methodological advance that is currently driving heightened interest in SNNs for computation is the realisation that data-centric, machine-learning (ML) approaches can be used to map tasks on to neuromorphic primitives. The modern approach to map a task onto a NM processor specifies an input-output mapping via a dataset or task simulation, combined with a value function that correlates with success in the desired task (Fig. 1c). Task definitions and datasets can be re-used from deep learning systems, as the steps are almost identical (Fig. 1b). Loss-gradient-based optimisation techniques can then be applied to deep SNNs, given some relatively simple adjustments. Since 2016, theoretical advances showed that surrogate gradients^{43–45}, back-and forward-propagation through time^{46,47} and spike-based direct gradient computation^{48,49} could open up SNNs to gradient-based methods. This new approach is highly similar to modern deep learning training methods, and permits SNNs to make direct use of software and hardware acceleration for training deep networks (Fig. 1).

The impact of gradient-based training for SNNs and Neuromorphic architectures has been enormous. It is now possible to train and deploy very large SNNs on tasks that have previously been in the realms of fantasy for spiking networks, for example large language models⁵⁰ and autonomous driving systems⁵¹. Gradient-based approaches can also be used to mitigate the effects of mismatch in mixed-signal and analog compute architectures^{52,53}. This application building approach is highly transferrable to even very complex HW architectures^{54,55}.

For the first time, NM compute architectures have a general-purpose programming model, permitting developers to map arbitrary tasks on to NM processors. This new approach makes deploying applications to NM hardware more straightforward, faster and more convenient than at any point in the past.

How to: popularise a new computing architecture

Achieving commercial success for NM processors requires satisfying a small constellation of conditions. The remaining barriers to commercialisation are not technical, but relate to softer aspects of widespread uptake. Here the main competitors are not alternative Neuromorphic processor designs, but are instead commodity devices that currently occupy the application niches NM companies would like to access. In the low-resource TinyML market, these are low-power

microcontrollers, low-power DSPs, FPGAs and other ASICs. In the high-resource datacenter ML market, these competitors are predominately tensor processors and other NN accelerators.

Market interest

For commercial uptake, market interests are primary—the direct benefits in terms of cost, power and capability that will justify a device manufacturer buying an NM processor over commodity devices. These business requirements should focus the attention of the Neuromorphic community towards applications where NM processors show a clear advantage.

Historically, some NM processor designs and systems have explicitly targeted large brain simulations, in the order of millions of neurons^{56–58}. As a potential avenue for commercialising NM technology, this use case suffers from several drawbacks. Firstly the global market size for brain simulation machines is currently very small, which is underlined by the necessity for research funding for these projects. Secondly, the paucity of Neuroscientific knowledge about precisely how organic brains are connected at a single-neuron and -synapse level points to uncertainty about how to usefully configure such large simulations. Although the sales margin for large-scale Neuromorphic supercomputers is high, these use-cases will not lead to widespread commercial adoption of Neuromorphic technologies.

In contrast, ML inference tasks are increasingly moving to portable consumer devices such as smartwatches, phones and headsets, as well as in-home devices such as smart speakers and cameras. Independence from cloud processing brings consumer advantages for privacy, data security, and robustness to network interference. Moving inference to the edge introduces application constraints for low-power, low-resource, real-time operation—known in the market as TinyML. TinyML is receiving significant attention from processor manufacturers, who are competing to obtain low-power operation at reasonable inference speeds. These constraints are ideal for NM processor technology, especially devices for which low-power operation is a key design goal (see Box). Concretely, this points to a niche for NM processors in low-power sensory processing and HCI, such as audio- and visual-wake phrase; and voice and gesture interaction for smart home consumer appliances. NM technologies can also provide general situational awareness for consumer devices, for example ambient audio analysis, presence detection, and vision- or motion-based human behaviour understanding.

Ease of use and interfacing

For engineers and application developers, questions of adoption come down to flexibility and ease of use: How straightforward is integration between a NM processor and the rest of a system? How steep is the learning curve for a new application developer? What restrictions exist in terms of applications which can be deployed? How reliable and robust will be the applications when deployed at scale? This last question is particularly pointed for mixed-signal and analog compute-in-memory architectures.

The NM community must ensure that answers to these questions lead commercial design engineers to favour NM technologies, in spite of the mature market dominance of von Neumann and tensor processor devices.

Since the design of NM processors has been primarily driven by the research community, relatively little attention has been paid to system-level design considerations. For commercial adoption, interfacing a NM processor with commodity hardware must be frictionless. This implies the NM community will need to adopt standard digital interfaces for configuration, data input and output; standard voltage rails used in consumer devices; compact packaging and pinouts; etc.

Adoption of the deep-learning inspired programming model and associated APIs solves the questions of flexibility and generality, while at the same time easing the learning curve for current ML engineers to

adopt NM processors. Familiarity is a valuable resource – when application design for an NM processor looks and feels very similar to that required for a tensor processor, it will be easier for engineers to advocate for NM processors in their systems. The NM community is already working to provide easy to use, easy to learn software toolchains, but a critical mass of NM application developers has yet to form. Community engagement in terms of tutorials, educational workshops and training will drive wider developer interest and adoption of NM programming.

Reliability and robustness

Neuromorphic hardware has had to work to overcome negative perceptions in terms of the expressive power of SNNs⁵⁹. At the same time, the reliability and robustness of mixed-signal NM devices, either compute-in-memory or traditional neuromorphic, have been impinged by mismatch and analog device non-idealities. These issues give rise to an image problem for Neuromorphic computation, which the field needs to overturn. Consumer device companies are used to a model where a single application is deployed at scale, identically, to a large number of devices. For the current breed of analog / compute-in-memory devices, non-idealities and mismatch impose a degree of calibration and per-device application fine-tuning which complicate mass deployment. However, the new breed of simpler, inference-only NM processors, based on standard spiking neuron modules and fabricated in standard digital CMOS, completely solve these questions of reliability and robustness.

Standardisation and benchmarks

Standardisation has come slowly to neuromorphic engineering, but a crop of new open-source software frameworks are already coalescing around the deep-learning programming model^{60–64}. These tools thematically extend earlier pushes for NM APIs, such as PyNN⁶⁵, which provided portability between NM simulators and some processors, but which did not address the absence of a programming model. Several projects for interoperability and benchmarking with broad community support are also underway^{66–68}, enabling application developers to easily pick and choose NM systems for deployment. Recent work has shown success in porting network architectures between seven software toolchains and four hardware platforms, requiring only few lines of code to deploy a single trained network to disparate NM processors⁶⁶.

While the motivation for the neuromorphic approach has always highlighted energy efficiency, it is only recently that head-to-head benchmarks between NM and other processing architectures have been available. For performing SNN inference, NM processors perhaps unsurprisingly show advantages over other computational architectures⁶⁹, with inference energy improvements of $280\text{--}21000 \times$ vs GPU on an SNN simulation task; $3\text{--}50 \times$ vs an edge ANN accelerator on the same task; and $75 \times$ vs Raspberry Pi on a Sudoku benchmark.

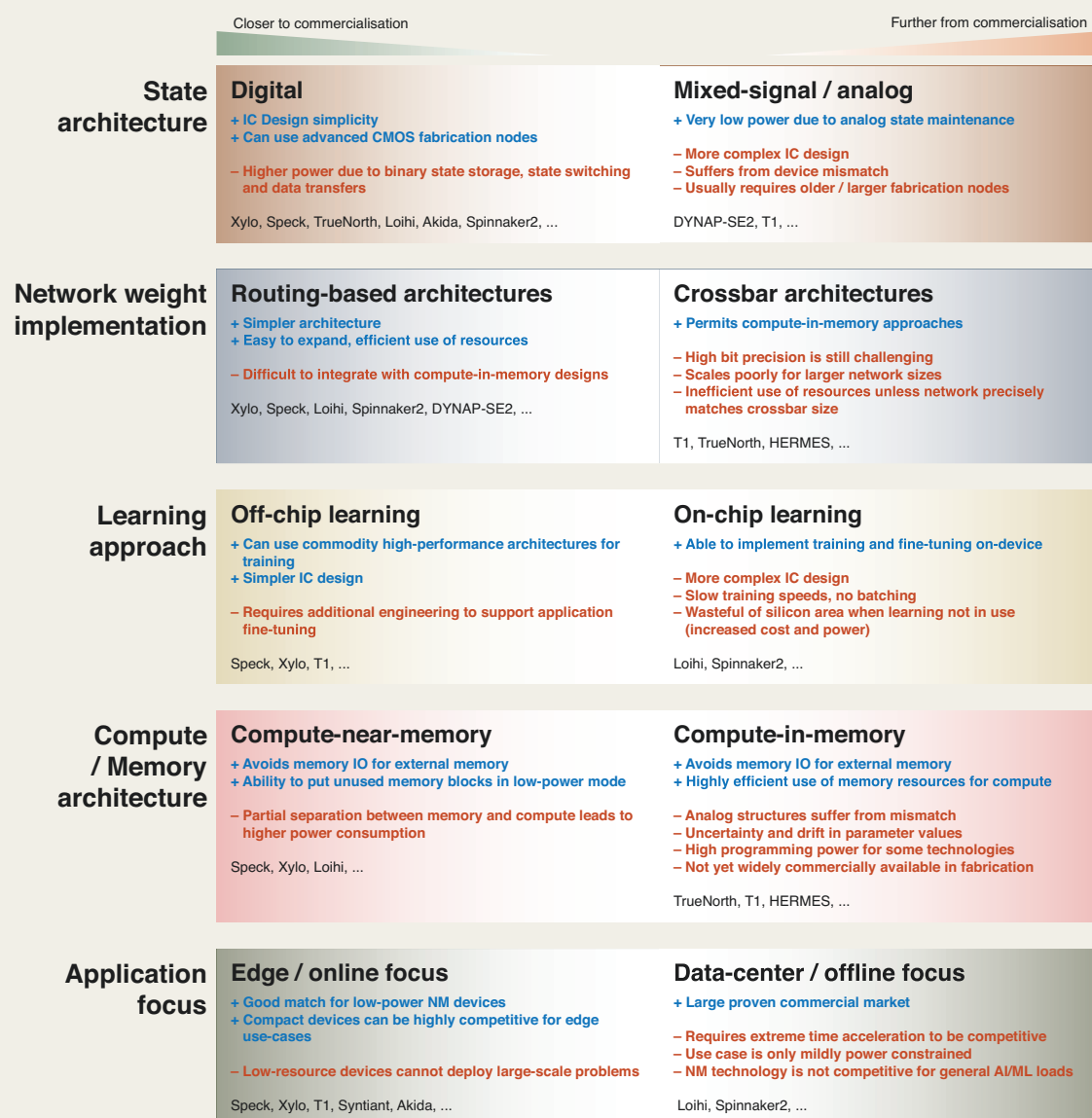
But on general physical simulation with neural networks⁷⁰, analog compute-in-memory (CIM) NM processors exhibit inference energy advantages over existing commercial devices, ranging $150\text{--}1300 \times$ improvement vs GPU and $1500\text{--}3500 \times$ vs CPU, on particle physics and condensed matter physics simulation tasks.

On dedicated deep learning tasks, CIM processors show distinct improvements over state-of-the-art NN accelerators with traditional architectures, in terms of energy efficiency ($10\text{--}1000 \times$ improvement) and compute density (around $10 \times$ depending on architecture), on ImageNet, CIFAR10 and LLM inference tasks^{71,72}.

When performing inference with neurally-inspired algorithms⁷³, SNN processors show dynamic energy improvements of $4.2\text{--}225 \times$ vs desktop GPU; $380 \times$ vs an edge mobile GPU; and $12 \times$ vs a desktop processor on an MNIST image reconstruction task.

On a range of sensory processing tasks (gesture recognition; audio keyword spotting; cue integration; and MNIST image

BOX



Current approaches to commercial NM hardware. Current Neuromorphic hardware offerings span a wide range of approaches in terms of hardware architecture, training approach and application focus. Here we provide a survey of the main options for commercialisation of Neuromorphic compute hardware in early 2025. These divide roughly in two families—on one side are elements which we believe can be commercialised quickly and successfully. On the other side are elements we believe can eventually show a significant commercial advantage, but which still require some research lead time before they reach commercial production.

State architecture. These describe design choices for maintaining the internal state of the computational units. Architectures using digital binary state maintenance can leverage standard memory cell designs^{20,21,115,118,119} on advanced fabrication nodes, with the drawback of potentially low-resolution state, and additional energy expenditure in switching state. Analog state architectures attain high energy efficiency and storage density by storing state as continuous analog values on individual circuit nodes^{120,121}, but with costs in terms of circuit complexity, device mismatch and reliance on older fabrication nodes.

Routing architecture. Choosing how computational units communicate within a Neuromorphic processor provides key constraints for IC design. Digital routing architectures can make highly efficient use of network resources, by power-cycling portions of parameter and state storage which are not needed for a given network^{21,115,118,120,122}. Crossbar routing architectures open the possibility of compute-in-memory implementations, but can be an inefficient use of resources unless precisely matched to network architecture size^{118,121}.

Learning approach. The essential choice for application building is where and how to train an SNN application. Off-chip learning brings the benefit of simpler IC design, by avoiding silicon expenditure on learning circuits, which can double the silicon requirements for inference alone, in the case of gradient-based training methods^{115,121,122}. Off-chip learning also enables full flexibility on learning methods and compute devices used for learning. On-chip learning devices require no additional resources for training and fine-tuning an application on-device^{21,118}. But learning circuits impose significant increases in silicon area—and therefore increased cost, energy

use and complexity. At the same time, end-to-end training of a NM application on a consumer device hugely complicates the application building process, since the developer has much less control over the data used in training.

Compute ↔ memory architecture. Neuromorphic processor designs often minimise memory IO bandwidth by moving memory either close to, or tightly coupled with computational elements. Compute-near-memory devices, which are often combined with digital routing architectures, can leverage standard memory cells for power-cycled operation when not needed^{21,115,122}. Compute-in-memory (CIM) architectures achieve extreme density of both memory and compute, by combining silicon resources for both in the same circuits^{20,95,121}. However, analog crossbar CIM architectures suffer from device non-idealities and value drift as well as slow and high-power write operations, and are not yet generally commercially available for fabrication.

Application focus. Choice of application focus provides key design constraints for NM processors. Consumers are increasingly expecting more intelligence and responsivity from their portable devices, along with increased battery life. Moving inference workloads from the cloud to the edge satisfies these demands, but creates tension between increased computational demands and need to for low-power operation. These Edge / IoT and online workloads are a good fit for small low-power NM devices, which can be highly competitive against von Neumann devices for these workloads^{115,121–123}. Larger devices aiming to address offline and data-center workloads have access to a proven commercial market^{21,118,119}, but stand in direct competition with existing processor manufacturers, and cannot easily leverage the low-power advantages of Neuromorphic technology.

Exemplar devices. SynSense Xylo¹¹⁵, SynSense Speck¹²², IBM TrueNorth²⁰, IBM HERMES⁹⁵, Intel Loihi²¹, BrainChip Akida¹¹⁹, Spinnaker2¹¹⁸, DYNAP-SE2¹²⁰, Innatera T1¹²¹, Syntiant NDP¹²³.

recognition), NM processors show power efficiency benefits over other low-power inference ASIC architectures, with continuous power improvements of $4\text{--}1700\times$, when normalised to processor node^{74,75}.

It is in real-time temporal signal analysis tasks that NM processors truly show off their energy benefits. On a spoken keyword spotting benchmark, where each architecture performs inference on the same benchmark dataset but using a HW-appropriate network, the neuromorphic approach brings power advantages of several orders of magnitude over GPUs and CPUs, and even over mobile GPU architectures^{76–78}.

Benchmarks that highlight the strengths of NM processors, on industry-standard tasks that can be compared against commodity devices, are particularly important to drive commercial uptake. We recommend that the NM community engages strongly with industry-standard benchmarks, ensuring that edge inference tasks which can show the benefits of the NM approach are included, and quantitatively illustrating those benefits. Similarly, establishing hardware standards, and adopting standard computational units such as the leaky-integrate-and-fire neuron (LIF), will signal to industry that the NM processor field is maturing commercially.

NVIDIA and CUDA

The crucial lesson for Neuromorphic hardware commercialisation is the conjunction of factors that led to the dramatic success of NVIDIA's tensor processors. Deep ANNs and CNNs have existed in a consistent form for decades, but training very large networks had been computationally infeasible. Likewise, gradient-based training methods have existed for considerable time, but without robust software toolchain support or hardware acceleration. GPUs from competitors to NVIDIA existed contemporaneously, but all were difficult to configure for strictly computational operation. In 2004, NVIDIA (along with DARPA, ATI, IBM and SONY) supported a prototype C language extension, Brook, for stream-based processing, with compilation support for several GPU architectures⁷⁹. In 2006 NVIDIA released the Tesla GPU architecture, explicitly tailored to support general-purpose computation in addition to graphical processing⁸⁰. Brook evolved into CUDA, which was officially released alongside the compute-tailored Tesla cores, and was promoted by NVIDIA as a general-purpose parallel computing API⁸. Soon after, CUDA and NVIDIA GPUs were adopted by researchers working to scale up neural network applications^{81,82}, paving the way for the CUDA-accelerated CNN architecture used by AlexNet in 2012⁸³, and the start of the Deep Learning boom.

With CUDA, NVIDIA provided an early, simple API for general-purpose computing on their tensor processors, which was taken up by deep learning toolchains and therefore by deep learning engineers.

Faster training powered by HW acceleration, and the ability to build larger models in a reasonable time, have delivered huge leaps in ML application performance. Increased interest in ML/AI applications has driven enormous concomitant interest and investment in training ML engineers, as well as enormous demand for ML training hardware supplied predominately by NVIDIA.

In other words, it was not until the advent of simple programming models and tools, simultaneous with straightforward API access to hardware, that tensor processors reached widespread adoption.

The NM community can actively drive this convergence for Neuromorphic processors. We recommend that the NM community, including funding bodies and researchers, support and consolidate the current crop of open-source tools so they reach industry-standard quality and usability.

Commercial concepts for NM computing devices

NM computing hardware is undergoing a commercial renaissance, with numerous companies fabricating processors that can reasonably claim the title of Neuromorphic (see Fig. 2). These offerings vary in their approach to hardware design and application targeting^{84,85} (see Box).

There has been significant overlap between research into analog compute-in-memory (CIM) architectures and event-driven computation, as the analog computing approach of mixed-signal spiking neuromorphic processors complements the analog signals generated by CIM crossbars. Combining these technologies can avoid the need for analog-digital converters on the output of a crossbar.

Analog CIM crossbars have been proposed using a range of technologies which are compatible with CMOS logic fabrication, encompassing flash-based memory⁸⁶, capacitor-based approaches⁷¹, floating-gate transistors⁸⁷, ferro-electric RAM⁸⁸, resistive RAM^{89–93}, Phase-Change Memory^{52,94,95}, as well as other approaches which are more difficult to integrate into the VLSI fabrication process.

CIM approaches show significant benefits over conventional non-CIM designs in terms of energy efficiency^{71,72} and computational density⁷¹. Although they do not show advances over existing conventional state-of-the-art tensor processors in terms of throughput^{96,97}, CIM architectures have considerably lower power consumption in real terms.

Crossbar designs suffer from issues of utilisation density and scaling. When scaling up a crossbar to support larger neural network layer widths, the crossbar size required to implement the corresponding linear weight matrices increases with the square of the layer width. At the same time, if a weight matrix is smaller than the available crossbar size, a portion of the crossbar resources will remain unused.

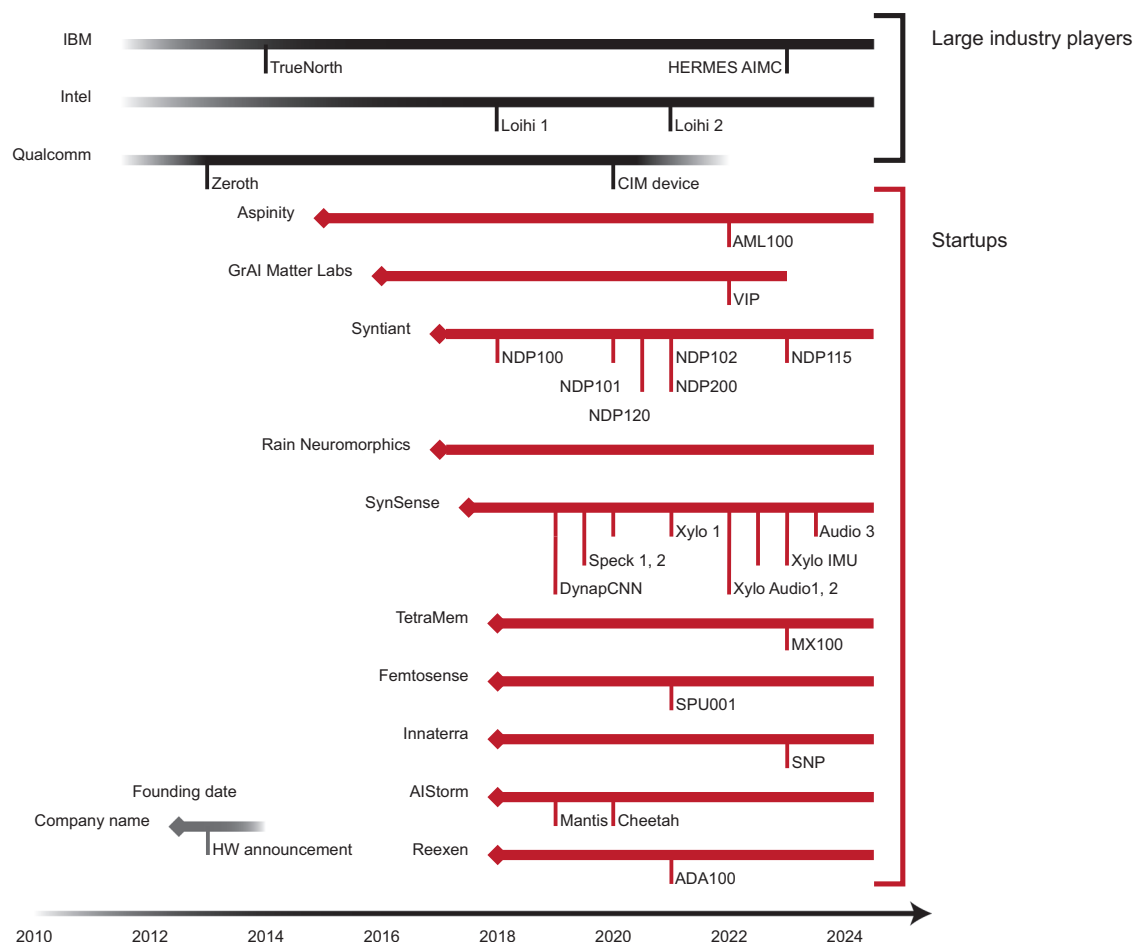


Fig. 2 | Timeline of recent commercial NM compute firms and new hardware. Large industry players (top) have committed efforts to NM compute hardware for some time. From around 2015, increased interest in NM compute as a commercial prospect has seen a cluster of new commercial startups and spinoffs emerge.

Shown here are commercially available announced hardware, focussing on NM compute. University research chips are not shown. Note that IBM's NorthPole is a near-memory efficient sparse tensor processor, rather than an SNN accelerator or compute-in-memory (CIM) processor¹⁷.

Since crossbars physically instantiate each weight or synapse, they cannot gain efficiency benefits from sparse inter-layer connections. Convolutional architectures are not easy to support on crossbar architectures, as shared weights must be replicated out across the physical crossbar array, or else temporally multiplexed across the input space. Several labs are exploring tiled configurations for in-memory computing crossbar arrays, which permit greater flexibility in network architecture while ensuring better utilisation density of crossbar resources^{71,93,98,99}.

Arrays of analog devices exhibit mismatch between ostensibly identical devices, due to imperfections in the silicon fabrication process^{100,101}. When used for computation, mismatch gives rise to variation in the behaviour of computing elements across an array and between chips, which can reduce computational accuracy if not mitigated through per-device calibration¹⁰², iterative parameter storage⁹⁹, chip-in-loop training⁹³ or by including mismatch during application training^{52,53,93}.

Digital devices (whether synchronous or asynchronous) have the advantage of being easy to synthesise at a range of advanced silicon fabrication nodes, and avoid issues of device mismatch in analog and mixed-signal devices. At the same time, digital routing architectures make efficient use of memory resources, especially in compute-near-memory devices.

Inference-only architectures are a good fit for train-once-deploy-at-scale commercial Neuromorphic applications, while still

permitting application fine-tuning with the cooperation of system application processors. The additional silicon area required for on-chip learning circuits implies additional cost, additional leakage power, and additional application complexity to control the learning problem in a deployed consumer device. For analog CIM architectures, the longer write latency and high write voltages for some technologies make them more appealing for inference-only work-loads, to avoid these costs incurred during training¹⁰³. Fixed-weight inference loads, as opposed to dynamic-weight inference, provide additional benefits in throughput, energy efficiency and latency¹⁰⁴.

Edge/IoT and online workloads are a good fit for compact, low-resource, low-cost and low-power NM devices. Compact NM processor designs can therefore be highly competitive in terms of power and performance against von Neumann architecture devices, for these use cases. In contrast, larger NM processor designs as well as multi-processor NM systems can have a proven market in the form of offline and data-centre workloads. However, in doing so they stand in direct competition with existing processor manufacturers. In addition, off-line workloads do not benefit as strongly from the low-power advantages of NM technology.

Commercial deployment at scale

With the question of how to program Neuromorphic computing applications more or less solved by gradient-based training, the focus

shifts to integration between NM compute and existing product ecosystems.

Only rarely will a NM processor exist in isolation in a consumer device. Usually it will interface with the other components of a system design. The choice of interfaces for a commercial NM processor should be guided by the need to communicate with standard commodity compute, standard memory technologies, and standard peripherals. For example, standard digital busses such as SPI and I²C should be preferred over custom parallel busses.

Use of standards extends also to power supply requirements. NM processors should prefer standard voltage supply levels, to be able to share rails with other system components. For watchdog use cases in edge devices, an NM processor should expect to be the only compute device active in a system. It should therefore be able to operate independently, then wake up and communicate with an application processor operating in standby or deep sleep mode. For some edge use cases the ability to directly communicate with a low-power networking peripheral such as Bluetooth low-energy (BLE) or Zigbee modules might enable simpler and lower-cost system designs.

In many edge systems physical space is at a premium. This suggests that NM processors should minimise silicon and packaging area, preferring low pin counts, serial interfaces and surface-mount packaging.

The constraints of mass production and mass deployment suggest that the ability to program a large number of NM processors with identical configurations, analogously to coding applications for commodity processors, will be beneficial. System designers are used to a standard application development and deployment flow, where a single application binary is copied identically to a large number of devices. To support this scheme, NM processors should present identical behaviour for identical configurations from first design principles. If this is not possible, devices should either be pre-calibrated before sale, such that all devices meet a common behaviour specification, or else be self-calibrating to achieve the same effect. Even in the case of NM devices that include online learning components, a basic network architecture configuration will be required. For almost all use cases, the ability to update the application binary with over-the-air (OTA) firmware updates will be beneficial. Economic requirements imply that high fabrication yield will be preferred; this suggests that use of commercially-available and well-validated fabrication nodes will be preferable, and will also help to reduce fabrication costs. Commercial constraints therefore suggest that NM processors should be reconfigurable, able to be fabricated at scale, and not single-application-specific devices.

NM processors for edge applications are frequently compute-near-memory devices, making use of local parameter and state storage to avoid the energy cost of off-chip data transfer. Most current commercial NM processors adopt volatile storage for this, often standard SRAM cells, implying that off-chip storage and associated interfacing is required for initial configuration on power-on. Again in the interests of system design simplicity, the ability for an NM processor to self-configure from external non-volatile storage on power-on, without requiring an intervening application processor, may be beneficial. On-chip FLASH storage or similar might be included in future devices for greater independence. Once novel non-volatile memory technologies are available at commercial scale, these may replace parameter storage in NM processors.

NM processors for compute-near-sensor use cases will benefit from customised low-power sensory interfaces for direct sensor connection. These devices may then be considered for sensor co-packaging, enabling extreme edge, compute-in-sensor products. In contrast, NM processors for multi-sensor and sensory integration use cases should include more extensive sensory interface options.

Data-center (DC) use cases will have a different set of constraints for NM processors. DC placement implies interfacing of NM

compute with standard high-performance computing (HPC) busses, and with HPC scheduling systems. Temporal multiplexing of applications, and multiplexing of layers in deep networks, implies frequent reconfiguration and parameter changes for DC-NM processors. Streaming of data through a static network configuration is also possible, but in either case significant off-chip memory bandwidth will be required. DC-centric NM processors will benefit from operating in accelerated time, with simulated or emulated neural dynamics operating much faster than required for streaming edge use cases.

One aspect not yet considered for any commercial NM processor is low-level data security. Binary application configurations may represent significant investments in data collection, engineering R&D effort, and IP value. This suggests that NM processors should support protection of configuration bitstreams, for example through encryption or write-only storage of application parameters.

While many of these requirements are addressed by the range of current NM processors, no current device fulfils all of them.

The final straight

There has historically been considerable hand-wringing in the Neuromorphic engineering community about the need for a so-called killer app. This would comprise a groundbreaking application that, standing alone, showcases the superiority of Neuromorphic engineering. The field has focussed on two formulations of this need: Which existing AI or application processors can be replaced by NM processors? And what fundamentally new application domains can be addressed by NM engineering?

An arguably more pertinent query is: Which applications and processors can be enhanced by NM compute?

The IoT and edge sensing market is expanding rapidly, with MEMS pressure sensors alone comprising more than USD\$2B in 2023¹⁰⁵. At the same time, considerable compute is moving to IoT sensory nodes to enable ML/AI analysis of sensory data in real time, independent of cloud compute^{106,107}. NM compute architectures are perfectly placed to enable ultra-low-power sensor-adjacent processing and condition detection for edge devices, and can benefit directly from this growing market. In-microphone voice activity detection (VAD) is one example use-case, along with general vibration-based condition and anomaly detection.

Battery-powered portable systems and consumer wearables in particular can benefit from the low-power edge inference capabilities of commercial NM processors. Consumer wearables shipments grew dramatically in 2019, reaching more than 500 million devices in 2022 for a global market value of USD\$61 billion^{108,109}. NM compute architectures can be used to offload sensory processing inference tasks from system processors and DNN tensor processors, particularly when delivered as IP for inclusion into the highly integrated SOCs which often power these compact devices. For example, low-power continuous bio-signal monitoring can be applied to ECG-based pathological heartbeat detection¹¹⁰ or EEG-based seizure detection¹¹¹. Edge processing for continuous audio monitoring is already used for spoken wake-phrase detection^{112,113}, and NM compute and processors can provide ultra-low-power wake-phrase¹¹⁴ and keyword spotting^{76,78}, as well as acoustic scene classification¹¹⁵.

In-camera processing is a nascent commercial market where NM compute architectures can compete effectively. Embedding NM processing into low-power machine vision sensors will enable visual wake-word, low-latency industrial monitoring, and safety monitoring applications.

In all these examples, providing simple and standardised interfaces to system processors is a must to facilitate adoption of NM compute devices. NM processing can be applied where it shows advantage (e.g. extracting and recognising visual and auditory features), and combined with conventional computing where NM

computing struggles (e.g. map building algorithms; controlling a wireless interface to transmit data).

In our view, the new example-based, ML-inspired Neuromorphic programming model and associated APIs are themselves the killer apps that will enable commercial uptake of Neuromorphic technologies. These allow almost universal application targeting, permit larger and more complex Neuromorphic applications, and have dramatically accelerated application development for Neuromorphic processors. Coupled with simple inference-only NM processors, the new programming model gives NM technology the look and feel of well-understood systems, while maintaining the significant benefits of the Neuromorphic approach. Taken together, these make NM technology commercially appealing as an alternative computing architecture, for the first time. The significant advantages of NM technology in terms of energy consumption and computational power for IoT, wearable and edge devices, are newly commercially viable as a result of the accuracy and flexibility granted by the ML programming model.

Interestingly, the new approach offers to pull in its wake the older analog and mixed-signal Neuromorphic designs relying on physics-based computing. The flexibility and power of differentiable programming permits building applications targeting almost any hardware model, as long as that model can be defined as a differentiable function⁵⁴. This approach can take non-idealities into account, solving problems of analog mismatch for both analog SNN and memristor compute-in-memory architectures^{52,53,116}. This opens a path towards commercialisation for ultra-low-power analog compute-in-memory designs, once they can demonstrate robust fabrication at scale.

The Neuromorphic community has the opportunity to replicate the success of tensor processors, by following a similar path towards commercialisation.

This implies the community should consolidate their efforts around the current crop of open-source tools to avoid further fragmentation. It also means standardisation of benchmarks, and a shift from niche, field-specific benchmarks towards industry-relevant applications where NM processors can be compared directly against commodity hardware. The NM community—both academic and commercial—should integrate with existing ML industry benchmark organisations and communities, to ensure that future benchmarks are suitable to show the benefits of NM processors. For commercial entities, we suggest the optimal market entry for NM processors will likely be on wearable, edge / IoT and sensor workloads, and recommend to focus product and business development efforts there. Research groups hoping to make commercial impact would do well to address the pain-points and difficulties that have prevented commercialisation until now. Specifically, exploring architectures and approaches that lend themselves to mass production; contributing to benchmarks and tools that are relevant to industry; and offering improvements in application building that are adaptable to commercial use. Graduate and undergraduate training courses can also be tailored towards commercial impact; not only in research programmes focussed specifically on Neuromorphic Engineering, but more broadly to bring NM approaches into Edge ML/AI courses. The next generation of engineers can be trained using the early commercial NM processors which are now available, improving their exposure and familiarity with NM compute.

The intriguing combination of neurobiological inspiration for hardware, and engineered optimisation for application building methods, promises to overcome the final hurdles that have held Neuromorphic processors back from widespread commercial success. It is up to the field to capitalise on the convergence of new hardware and new approaches—as in Neuromorphic processors themselves, time (and timing) is a key factor.

Data availability

No datasets were generated or analysed during the current study.

References

- McCulloch, W. S. & Pitts, W. A logical calculus of the ideas immanent in nervous activity. *Bull. Math. Biophysics* **5**, 115–133 (1943).
- Turing, A. et al. *Intelligent Machinery*. Tech. Rep. (1948).
- von Neumann, J. et al. *The Computer and the Brain* (Yale University Press, 1958).
- von Neumann, J. et al. The general and logical theory of automata. In *Cerebral Mechanisms in Behavior. The Hixon Symposium*, 1–31 (John Wiley & Sons, Inc., 1948).
- Taubes, G. A. et al. *The Rise and Fall of Thinking Machines*. Inc. 61–65 (1995).
- Schatz, W. et al. *Laying a Bet To Rest*. Upside 39–45 (1996).
- LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521**, 436–444 (2015).
- Nickolls, J., Buck, I., Garland, M. & Skadron, K. Scalable parallel programming with CUDA: Is CUDA the parallel programming model that application developers have been waiting for? *Queue* **6**, 40–53 (2008).
- Gilder, G. et al. *The Silicon Eye*. Enterprise (WW Norton, New York, NY, 2005).
- Mead, C. How we created neuromorphic engineering. *Nat. Electron.* **3**, 434–435 (2020).
- Mahowald, M. & Douglas, R. A silicon neuron. *Nature* **354**, 515–518 (1991).
- Maher, M., Deweerth, S., Mahowald, M. & Mead, C. Implementing neural architectures using analog vlsi circuits. *IEEE Trans. Circuits Syst.* **36**, 643–652 (1989).
- Mead, C. A. & Mahowald, M. A silicon model of early visual processing. *Neural Netw.* **1**, 91–97 (1988).
- Lyon, R. & Mead, C. An analog electronic cochlea. *IEEE Trans. Acoust., Speech, Signal Process.* **36**, 1119–1134 (1988).
- Indiveri, G. & Horiuchi, T. K. Frontiers in neuromorphic engineering. *Frontiers in Neuroscience*. <https://doi.org/10.3389/fnins.2011.00118> (2011).
- Lyon, R. F. et al. US4521772A: Cursor control device. <https://patents.google.com/patent/US4521772A> (1983).
- Bidiville, M. et al. US5288993A: Cursor pointing device utilizing a photodetector array with target ball having randomly distributed speckles. <https://patents.google.com/patent/US5288993A> (1994).
- Lichtsteiner, P., Delbruck, T. & Kramer, J. Improved on/off temporally differentiating address-event imager. In *Proceedings of the 2004 11th IEEE International Conference on Electronics, Circuits and Systems, 2004. ICECS 2004.*, 211–214 (2004).
- Gehlhaar, J. et al. Neuromorphic processing: a new frontier in scaling computer architecture. In *ASPLOS*, 317–318 (2014).
- DeBole, M. V. et al. TrueNorth: accelerating from zero to 64 million neurons in 10 years. *Computer* **52**, 20–29 (2019).
- Davies, M. et al. Advancing neuromorphic computing with Loihi: a survey of results and outlook. *Proc. IEEE* **109**, 911–934 (2021).
- Ryu, H. E. et al. *Industrial DVS Design: Key Features and Applications*. (2019).
- Schuman, C. D. et al. Opportunities for neuromorphic computing algorithms and applications. *Nat. Computational Sci.* **2**, 10–19 (2022).
- Burelo, K., Ramantani, G., Indiveri, G. & Sarnthein, J. A neuromorphic spiking neural network detects epileptic high frequency oscillations in the scalp EEG. *Sci. Rep.* **12**, 1798 (2022).
- Mitra, S., Fusi, S. & Indiveri, G. Real-time classification of complex patterns using spike-based learning in neuromorphic VLSI. *IEEE Trans. Biomed. circuits Syst.* **3**, 32–42 (2008).
- Mayr, C. G., Sheik, S., Bartolozzi, C. & Chicca, E. Editorial: Synaptic plasticity for neuromorphic systems. *Front. Neurosci.* **10**, 214 (2016).

27. Khacef, L. et al. Spike-based local synaptic plasticity: A survey of computational models and neuromorphic circuits. *Neuromorph. Comput. Eng.* **3**, 042001 (2023).
28. Maass, W. & Markram, H. On the computational power of circuits of spiking neurons. *J. Computer Syst. Sci.* **69**, 593–616 (2004).
29. Rasmussen, D. NengoDL: Combining deep learning and neuromorphic modelling methods. *Neuroinformatics* **17**, 611–628 (2019).
30. Gonon, L. & Ortega, J.-P. Reservoir computing universality with stochastic inputs. *IEEE Trans. Neural Netw. Learn. Syst.* **31**, 100–112 (2020).
31. Nicola, W. & Clopath, C. Supervised learning in spiking neural networks with FORCE training. *Nat. Commun.* **8**, 2208 (2017).
32. Brendel, W., Bourdoukan, R., Vertechi, P., Machens, C. K. & Denève, S. Learning to represent signals spike by spike. *PLOS Computational Biol.* **16**, e1007692 (2020).
33. Büchel, J., Zendrikov, D., Solinas, S., Indiveri, G. & Muir, D. R. Supervised training of spiking neural networks for robust deployment on mixed-signal neuromorphic processors. *Sci. Rep.* **11**, <https://doi.org/10.1038/s41598-021-02779-x> (2021).
34. Voelker, A., Kajić, I. & Eliasmith, C. Legendre Memory Units: Continuous-Time Representation in Recurrent Neural Networks. In: *Advances in Neural Information Processing Systems* 32 (NeurIPS, 2019).
35. Stoffel, M. & Tandale, S. B. Spiking neural networks for nonlinear regression of complex transient signals on sustainable neuromorphic processors. *npj Unconv. Comput.* **1**, 2 (2024).
36. Stan, M. I. & Rhodes, O. Learning long sequences in spiking neural networks. *Sci. Rep.* **14**, 21957 (2024).
37. Sandamirskaya, Y. et al. Dynamic neural fields as a step toward cognitive neuromorphic architectures. *Frontiers in Neuroscience* **7**, <https://doi.org/10.3389/fnins.2013.00276> (2014).
38. Stewart, T. C. & Eliasmith, C. Large-scale synthesis of functional spiking neural circuits. *Proc. IEEE* **102**, 881–898 (2014).
39. Massoud, T. M. & Horiuchi, T. K. A neuromorphic VLSI head direction cell system. *IEEE Trans. Circuits Syst. I: Regul. Pap.* **58**, 150–163 (2010).
40. Neftci, E. et al. Synthesizing cognition in neuromorphic electronic systems. *Proc. Natl Acad. Sci.* **110**, E3468–E3476 (2013).
41. Stewart, T., Choo, F.-X. & Eliasmith, C. Spaun: A perception-cognition-action model using spiking neurons. In *Proc. Annual Meeting of the Cog. Sci. Society*, **34** (2012).
42. Kreiser, R., Renner, A., Sandamirskaya, Y. & Pienroj, P. Pose estimation and map formation with spiking neural networks: towards neuromorphic slam. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2159–2166 (IEEE, 2018).
43. Lee, J. H., Delbruck, T. & Pfeiffer, M. Training deep spiking neural networks using backpropagation. *Frontiers in Neuroscience* **10**, <https://doi.org/10.3389/fnins.2016.00508> (2016).
44. Neftci, E. O., Mostafa, H. & Zenke, F. Surrogate gradient learning in spiking neural networks: bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Process. Mag.* **36**, 51–63 (2019).
45. Cramer, B. et al. Surrogate gradients for analog neuromorphic computing. *Proc. Natl Acad. Sci.* **119**, e2109194119 (2022).
46. Yin, B., Corradi, F. & Bohtë, S. M. Accurate online training of dynamical spiking neural networks through forward propagation through time. *Nat. Mach. Intell.* **5**, 518–527 (2023).
47. Lee, J. H., Haghighatshoar, S. & Karbasi, A. *Exact Gradient Computation for Spiking Neural Networks via Forward Propagation*. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, 1812–1831 (PMLR, 2023).
48. Göltz, J. et al. Fast and energy-efficient neuromorphic deep learning with first-spike times. *Nat. Mach. Intell.* **3**, 823–835 (2021).
49. Wunderlich, T. C. & Pehle, C. Event-based backpropagation can compute exact gradients for spiking neural networks. *Sci. Rep.* **11**, 12829 (2021).
50. Zhu, R.-J., Zhao, Q. Li, G. & Eshraghian, J. K. SpikeGPT: Generative Pre-trained Language Model with Spiking Neural Networks. *Trans. ML Research* <https://openreview.net/forum?id=gcf1anBL9e> (2024).
51. Zhu, R.-J., Wang, Z., Gilpin, L. & Eshraghian, J. K. Autonomous driving with spiking neural networks. *Trans. ML Research* <https://openreview.net/forum?id=95VyH4VxN9> (2024).
52. Joshi, V. et al. Accurate deep neural network inference using computational phase-change memory. *Nature Commun.* **11**, 2473 (2020).
53. Büchel, J., Faber, F. F. & Muir, D. R. Network insensitivity to parameter noise via parameter attack during training. In *International Conference on Learning Representations* (2022).
54. Wright, L. G. et al. Deep physical neural networks trained with backpropagation. *Nature* **601**, 549–555 (2022).
55. Kazemi, A. et al. Achieving software-equivalent accuracy for hyperdimensional computing with ferroelectric-based in-memory computing. *Sci. Rep.* **12**, 19201 (2022).
56. Briiderle, D. et al. Simulator-like exploration of cortical network architectures with a mixed-signal VLSI system. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems* (IEEE, 2010).
57. Furber, S. & Bogdan, P. (eds.) *SpiNNaker: A Spiking Neural Network Architecture* (Now Publishers, 2020).
58. Brüderle, D. et al. A comprehensive workflow for general-purpose neural modeling with highly configurable neuromorphic hardware systems. *Biol. Cybern.* **104**, 263–296 (2011).
59. LeCun, Y. 1.1 deep learning hardware: Past, present, and future. In *2019 IEEE International Solid-State Circuits Conference - (ISSCC)* (IEEE, 2019).
60. Muir, D., Bauer, F. & Weidel, P. Rockpool Documentation, <https://doi.org/10.5281/zenodo.3773845> (2019).
61. Sheik, S., Lenz, G., Bauer, F. & Kuepelioglu, N. SINABS: a simple Pytorch based SNN library specialised for speck. <https://github.com/synsense/sinabs>.
62. Fang, W. et al. Spikingjelly. <https://github.com/fangwei123456/spikingjelly> (2020).
63. Pehle, C. & Pedersen, J. E. Norse — a deep learning library for spiking neural networks. <https://doi.org/10.5281/zenodo.4422025> (2021).
64. Eshraghian, J. K. et al. Training spiking neural networks using lessons from deep learning. *Proc. IEEE* **111**, 1016–1054 (2023).
65. Davison, A. P. et al. PyNN: a common interface for neuronal network simulators. *Frontiers in Neuroinformatics* **2**, <https://www.frontiersin.org/journals/neuroinformatics/articles/10.3389/neuro.11.011.2008> (2009).
66. Pedersen, J. E. et al. Neuromorphic intermediate representation: a unified instruction set for interoperable brain-inspired computing. *Nat. Commun.* **15**, 8122 (2024).
67. Yik, J. et al. The neurobench framework for benchmarking neuromorphic computing algorithms and systems. *Nat. Commun.* **16**, 1545 (2025).
68. Banbury, C. et al. MLPerf Tiny Benchmark. *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks* (2021).
69. Ostrau, C., Klarhorst, C., Thies, M. & Rückert, U. Benchmarking neuromorphic hardware and its energy expenditure. *Front. Neurosci.* **16**, 873935 (2022).
70. Kösters, D. J. et al. Benchmarking energy consumption and latency for neuromorphic computing in condensed matter and particle physics. *APL Mach. Learn.* **1**, 016101 (2023).

71. Jia, H. et al. Scalable and programmable neural network inference accelerator based on in-memory computing. *IEEE J. Solid-State Circuits* **57**, 198–211 (2022).
72. Wolters, C., Yang, X., Schlichtmann, U. & Suzumura, T. Memory is all you need: an overview of compute-in-memory architectures for accelerating large language model inference. ArXiv <https://doi.org/10.48550/arXiv.2406.08413> (2024).
73. Parpart, G., Risbud, S. R., Kenyon, G. T. & Watkins, Y. Implementing and benchmarking the locally competitive algorithm on the loihi 2 neuromorphic processor. In: *ICONS '23: Proceedings of the 2023 International Conference on Neuromorphic Systems*, vol. 15, 1–6 <https://doi.org/10.1145/3589737.3605973> (2023).
74. Frenkel, C. & Indiveri, G. ReckOn: A 28nm sub-mm² task-agnostic spiking recurrent neural network processor enabling on-chip learning over second-long timescales. In *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, **65**, 1–3 (IEEE, 2022).
75. Ottati, F. et al. To Spike or Not To Spike: A digital hardware perspective on deep learning acceleration. *IEEE J. Emerg. Sel. Top. Circuits Syst.* **13**, 1015–1025 (2023).
76. Blouw, P., Choo, X., Hunsberger, E. & Eliasmith, C. Benchmarking keyword spotting efficiency on neuromorphic hardware. <http://arxiv.org/abs/1812.01739> (2018).
77. Yan, Y. et al. Comparing Loihi with a SpiNNaker 2 prototype on low-latency keyword spotting and adaptive robotic control. *Neuromorphic Comput. Eng.* **1**, 014002 (2021).
78. Bos, H. & Muir, D. R. Micro-power spoken keyword spotting on xylo audio 2. Preprint at <https://arxiv.org/abs/2406.15112> (2024).
79. Buck, I. et al. Brook for GPUs: Stream computing on graphics hardware. *ACM Trans. Graph.* **23**, 777–786 (2004).
80. Lindholm, E., Nickolls, J., Oberman, S. & Montrym, J. NVIDIA Tesla: a unified graphics and computing architecture. *IEEE Micro* **28**, 39–55 (2008).
81. Mohamed, A.-r., Dahl, G. & Hinton, G. *Deep Belief Networks For Phone Recognition*. In *Neural Information Processing Systems* (Vancouver, Canada, 2009).
82. Witt, S. How Jensen Huang's Nvidia Is Powering the A.I. Revolution. *The New Yorker*. <https://www.newyorker.com/magazine/2023/12/04/how-jensen-huang-nvidia-is-powering-the-ai-revolution> (2023).
83. Krizhevsky, A., Sutskever, I. & Hinton, G. E. ImageNet classification with deep convolutional neural networks. *Commun. ACM* **60**, 84–90 (2017).
84. Christensen, D. V. et al. 2022 roadmap on neuromorphic computing and engineering. *Neuromorph. Comput. Eng.* **2**, 022501 (2022).
85. Frenkel, C., Bol, D. & Indiveri, G. Bottom-up and top-down approaches for the design of neuromorphic processing systems: Tradeoffs and synergies between natural and artificial intelligence. *Proceedings of the IEEE* (2023).
86. Fick, L., Skrzyniarz, S., Parikh, M., Henry, M. B. & Fick, D. Analog matrix processor for edge AI real-time video analytics. In *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, **65**, 260–262 (2022).
87. Merrikh-Bayat, F. et al. High-performance mixed-signal neuro-computing with nanoscale floating-gate memory cell arrays. *IEEE Trans. Neural Netw. Learn. Syst.* **29**, 4782–4790 (2018).
88. Lederer, M. et al. Ferroelectric field effect transistors as a synapse for neuromorphic application. *IEEE Trans. Electron Devices* **68**, 2295–2300 (2021).
89. Chen, Y. ReRAM: history, status, and future. *IEEE Trans. Electron Devices* **67**, 1420–1433 (2020).
90. Liu, Q. et al. 33.2 A fully integrated analog ReRAM based 78.4TOPS/W Compute-in-memory chip with fully parallel MAC computing. In *2020 IEEE International Solid-State Circuits Conference - (ISSCC)*, 500–502 (2020).
91. Xue, C.-X. et al. 15.4 A 22nm 2Mb ReRAM Compute-in-Memory Macro with 121-28TOPS/W for Multibit MAC Computing for Tiny AI Edge Devices. In *2020 IEEE International Solid-State Circuits Conference - (ISSCC)*, 244–246. <https://ieeexplore.ieee.org/abstract/document/9063078> (2020).
92. Hung, J.-M. et al. A four-megabit compute-in-memory macro with eight-bit precision based on CMOS and resistive random-access memory for AI edge devices. *Nat. Electron.* **4**, 921–930 (2021).
93. Wan, W. et al. A compute-in-memory chip based on resistive random-access memory. *Nature* **608**, 504–512 (2022).
94. Khwa, W.-S. et al. A 40-nm, 2M-Cell, 8b-Precision, Hybrid SLC-MLC PCM Computing-in-Memory Macro with 20.5–65.0TOPS/W for Tiny-AI Edge Devices. In *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, **65**, 1–3 (2022).
95. Le Gallo, M. et al. A 64-core mixed-signal in-memory compute chip based on phase-change memory for deep neural network inference. *Nat. Electron.* **6**, 680–693 (2023).
96. NVIDIA A100 GPUs Power the Modern Data Center. <https://www.nvidia.com/en-au/data-center/a100/>.
97. NVIDIA H100 Tensor Core GPU. <https://www.nvidia.com/en-au/data-center/h100/>.
98. Klein, J. et al. ALPINE: Analog In-Memory Acceleration With Tight Processor Integration for Deep Learning. *IEEE Transactions on Computers* **72**. <https://ieeexplore.ieee.org/document/9991844> (2023).
99. Dalgaty, T. et al. Mosaic: in-memory computing and routing for small-world spike-based neuromorphic systems. *Nat. Commun.* **15**, 142 (2024).
100. Pelgrom, M., Duinmaijer, A. & Welbers, A. Matching properties of MOS transistors. *IEEE J. Solid-State Circuits* **24**, 1433–1439 (1989).
101. Tuinhout, H., Wils, N. & Andricciola, P. Parametric mismatch characterization for mixed-signal technologies. *IEEE J. Solid-State Circuits* **45**, 1687–1696 (2010).
102. Neftci, E. & Indiveri, G. A device mismatch compensation method for VLSI neural networks. In *2010 Biomedical Circuits and Systems Conference (BioCAS)*, 262–265. <https://ieeexplore.ieee.org/abstract/document/5709621> (2010).
103. Yu, S., Sun, X., Peng, X. & Huang, S. Compute-in-memory with emerging nonvolatile-memories: challenges and prospects. In *2020 IEEE Custom Integrated Circuits Conference (CICC)*, 1–4 (2020).
104. Burr, G. W. et al. Design of analog-AI hardware accelerators for transformer-based language models (Invited). In *2023 International Electron Devices Meeting (IEDM)*, 1–4 (2023).
105. Clarke, P. Bosch tops two of five pressure sensor rankings. <https://www.eenewseurope.com/en/bosch-tops-two-of-five-pressure-sensor-rankings/> (2018).
106. Ahmed, M. Where sensor fusion and sensor processors stand in IoT. <https://www.eetimes.com/where-sensor-fusion-and-sensor-processors-stand-in-iot/> (2023).
107. Ahmed, M. Sensor fusion with AI transforms the smart manufacturing Era. <https://www.eetimes.com/sensor-fusion-with-ai-transforms-the-smart-manufacturing-era/> (2023).
108. Wearables unit shipments worldwide from 2014 to 2022 (in millions), by vendor. Tech. Rep., Statista <https://www.statista.com/statistics/515634/wearables-shipments-worldwide-by-vendor/> (2023).
109. Wearable Technology Market Size, Share & Trends Analysis Report By Product (Head & Eyewear, Wristwear), By application (Consumer Electronics, Healthcare), by region (Asia Pacific, Europe), And Segment Forecasts, 2023 - 2030. Tech. Rep. 978-1-68038-165-8, Grand View Research <https://www.grandviewresearch.com/industry-analysis/wearable-technology-market> (2022).

110. Bauer, F. C., Muir, D. R. & Indiveri, G. "real-time ultra-low power ECG anomaly detection using an event-driven neuromorphic processor. *IEEE Transactions on Biomedical Circuits and Systems*.
111. Li, R. et al. Real-time sub-milliwatt epilepsy detection implemented on a spiking neural network edge inference processor. *Computers Biol. Med.* **183**, 109225 (2024).
112. Team, S. Hey Siri: An on-device DNN-powered voice trigger for apple's personal assistant. <https://machinelearning.apple.com/research/hey-siri> (2017).
113. Coucke, A. et al. *Efficient Keyword Spotting Using Dilated Convolutions and Gating*. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6351–6355 (2019).
114. Weidel, P. & Sheik, S. WaveSense: efficient temporal convolutions with spiking neural networks for keyword spotting. <http://arxiv.org/abs/2111.01456> (2021).
115. Bos, H. & Muir, D. *Sub-mW Neuromorphic SNN Audio Processing Applications with Rockpool and Xylo*, 69–78 (CRC Press, 2022).
116. Cakal, U., Wu, C., Ulusoy, I. & Muir, D. R. Gradient-descent hardware-aware training and deployment for mixed-signal neuromorphic processors. *Neuromorphic Comput. Eng.* **4**, 014011 (2024).
117. Modha, D. S. et al. Neural inference at the frontier of energy, space, and time. *Science* **382**, 329–335 (2023).
118. Mayr, C., Höppner, S. & Furber, S. B. Spinnaker 2: a 10 million core processor system for brain simulation and machine learning. Preprint at <http://arxiv.org/abs/1911.02385> (2019).
119. Vanarse, A., Osseiran, A., Rassau, A. & van der Made, P. A hardware-deployable neuromorphic solution for encoding and classification of electronic nose data. *Sensors* **19**, 4831 (2019).
120. Richter, O. et al. DYNAP-SE2: A scalable multi-core dynamic neuromorphic asynchronous spiking neural network processor. *Neuromorph. Comput. Eng.* **4**, 014003 (2024).
121. Innatera. <https://www.innatera.com/> (2023).
122. Speck: A Smart Event-based Vision Sensor With A Low Latency 327k Neuron Convolutional Neuronal Network Processing Pipeline. <https://speck.ai>.
123. Syntiant Neural Decision Processors. <https://www.syntiant.com/neural-decision-processors> (2023).

Acknowledgements

This work was partially funded by the ECSEL Joint Undertaking (JU) under grant agreement number 876925, "ANDANTE" (D.R.M.). The JU receives support from the European Union Horizon 2020 research and innovation program and France, Belgium, Germany, Netherlands, Portugal, Spain and Switzerland. This work was partially funded by the KDT JU under grant agreement number 101097300, "EdgeAI" (D.R.M.). The authors

gratefully acknowledge discussions at the Telluride Neuromorphic workshop.

Author contributions

D.R.M. and S.S. conceived the perspective piece, discussed the concepts and argumentation, and approved the final text. D.R.M. wrote and edited the text.

Competing interests

Both D.R.M. and S.S. are associated with SynSense, a commercial Neuromorphic Technology company. The opinions contained in this article are the personal opinions of the authors, and do not represent the views of their employers.

Additional information

Correspondence and requests for materials should be addressed to Dylan Richard Muir.

Peer review information *Nature Communications* thanks Matthew Marinella and Andrew Nere for their contribution to the peer review of this work.

Reprints and permissions information is available at <http://www.nature.com/reprints>

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

© The Author(s) 2025